D3.8 Final version of the personalization and positioning software tool with documentation. PIPER EU Project

Philippe Beillas, Yoann Lafon, Bertrand Frechede, Tomas Janak, Thomas Dupeux, Matthieu Mear, Svein Kleiven, Chiara Giordano, Victor Alvarez, Xiaogai Li, et al.

Project funded by the European Commission
within the 7th Framework programme

[D3.8] Final version of the personalization and positioning software tool with documentation

PIPER			
Position and Personalize Advanced Human Body Models for Injury Prediction			
Project nr:	605544	Call reference:	SST.2013.4-3.
Start date:	1st November 2013	Duration:	42 months

Deliverable identification			
Leading beneficiary:	1	Planned delivery date:	M36
Related WP:	4	Actual delivery date:	M42
Dissemination level:	PU	Delivery status:	

Contributors	
Beneficiary name	Contributor(s)' name(s)
Université Claude Bernard Lyon 1	P Beillas, Y Lafon, B Fréchède, T Janak, T Dupeux, M Mear, A Le Ruyet, M Gardemont, I Theodorakos,
GIE de Recherches et d'Etudes PSA Renault	
Kungliga Tekniska Hoegskolan	S Kleiven, C Giordano, V Alvarez, X Li
Technische Universitat Berlin	
Foundation for Innovation and Technology Transfer (FITT)	A Chawla, A Chhabra, S Paruchuri, S Singh, D Kaushik, S Mukherjee, S Kumar, K Devane, K Mishra, G Machina
Centre Européen d'Etudes de Sécurité et d'Analyse des Risques	E. Jolivet
Institut National de Recherche en Informatique et en Automatique	T Lemaire, F Faure, B Gilles, U. Vimont
Partnership for Dummy Technology and Biomechanics	
University of Southampton	C Lecomte
Lyon Ingénierie Projets	

Reviewers		
Version	Reviewer	Date

Table of content

1. Foreword: report contents

The aim of this report is to provide an overview of the final version of the PIPER framework and application. The software, along with its documentation, and not the report, constitutes the main part of the deliverable.

The software and documentation were already distributed at the Final Workshop and online (under the Open Source license GPLv2 or later for the software, and the GNU FDL 1.3 license for the documentation). The documentation includes detailed descriptions of the framework principles, user interface, metadata, along with the modules and their parameters. It also includes application scenarios (called workflows). Information about the use of the modules is complemented by Tutorials that were developed as part of WP1 (online on the wiki) and explanatory videos were developed as part of WP4 (videos of the final workshop, now available on YouTube). The headers in the source code files (also available online) list the main contributors to the software.

The report will therefore not provide details about information that is already available elsewhere but will only provide a very brief summary of the functionalities available. Some of the descriptions are excerpts of the manual.

Links to access the software, documentation, tutorials, etc. can be found at http://piper-project.org or http://piper-project.eu. The user manual is provided in appendix for reference.

2. PIPER Framework

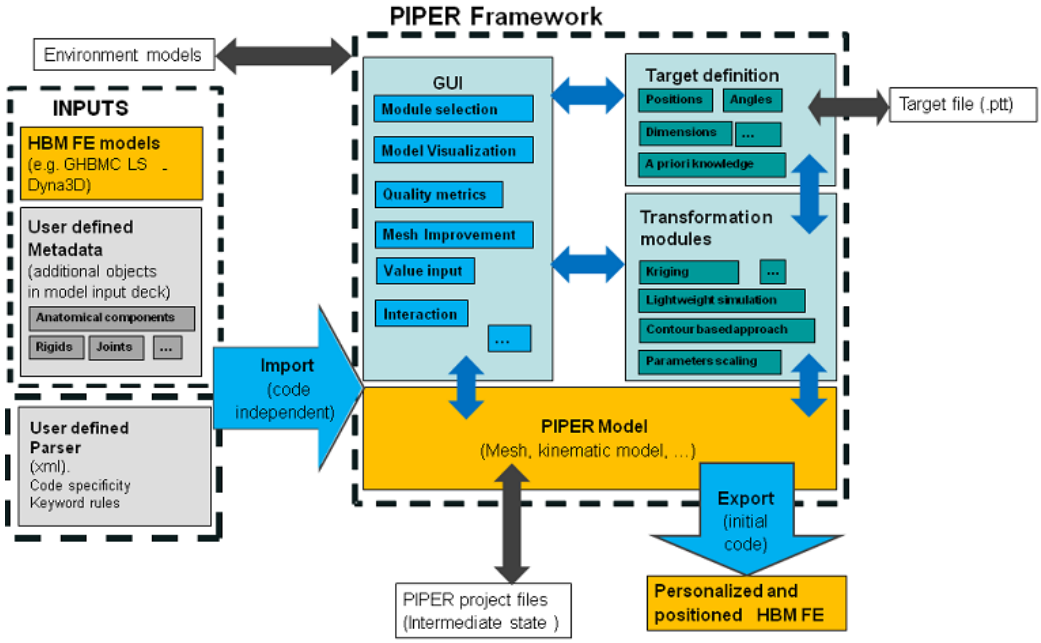
2.1 Overview and main concepts

The PIPER software framework was developed to help with the positioning and the personalization of Human Body Models (HBM) for injury prediction to be used in road safety. These HBM are typically available in one size and one posture (which can be difficult or time consuming to change), and they are implemented in commercial explicit Finite Element (FE) codes such as Ld-Dyna3D (LSTC), Pamcrash (ESI), Radioss (Altair) or Abaqus.

The framework aims to be modular, model and code agnostic. More specifically, the idea is to be able to apply the same positioning, scaling or personalizing methodologies to several models in different codes, resulting in models that can be used in simulations with little or no correction. For this, the framework handles the import and export of the model, and model transformation methodologies are implemented in reusable modules. In order to facilitate the real time user interactions, the PIPER framework only uses geometric or lightweight physics approaches to transform the FE model.

In practice, the import, export, and most modules developed up to now are included in a main application that also provides a GUI, a 3D display of the model and a Python scripting interface. As it is Open Source, the framework and application use many other open source libraries. The framework can easily be extended by adding modules or through scripting.

A schematic overview of the software structure is provided below.



2.2 Framework concepts

A number of concepts were developed for the framework to support the required functionalities. All concepts are described in detail in the user manual, and a brief list is provided below:

- Rule files (.pfr): in order to allow working with various solvers, the specification of the FE format (e.g. geometry etc.) is expressed using a simple rule language that was developed and implemented using an xml syntax. These files can be created and edited by users to extend the support for a given solver or add support for a new solver. Sample files are provided for ls-dyna.
- Metadata: in order to work with various HBM, a system of metadata was created to put in correspondence the internal PIPER concepts needed for scaling and positioning with the FE entities. It uses groups (written in the FE format) and a metadata file (pmr, xml syntax). Files were created for leading HBMs including the GHBMC mid male M50 occupant detailed, M50 pedestrian simplified, Thums AM50 occupant versions 3 and 4, and of course the PIPER scalable child model. While the metadata developed up to now do not support all modules for all HBM, metadata can be developed from scratch or expanded by users without recompiling the software.
- A common vocabulary is required to be able to work with different HBM which all use their own conventions. AnatomyDB (anatomy database with definitions, vocabulary and synonyms,

relationships, etc.) was developed and is provided with PIPER. It covers definition of bones, joints, landmarks among others and can be updated by the user. It is released under the LGPL license for the software and Creative common to facilitate re-use in other software.

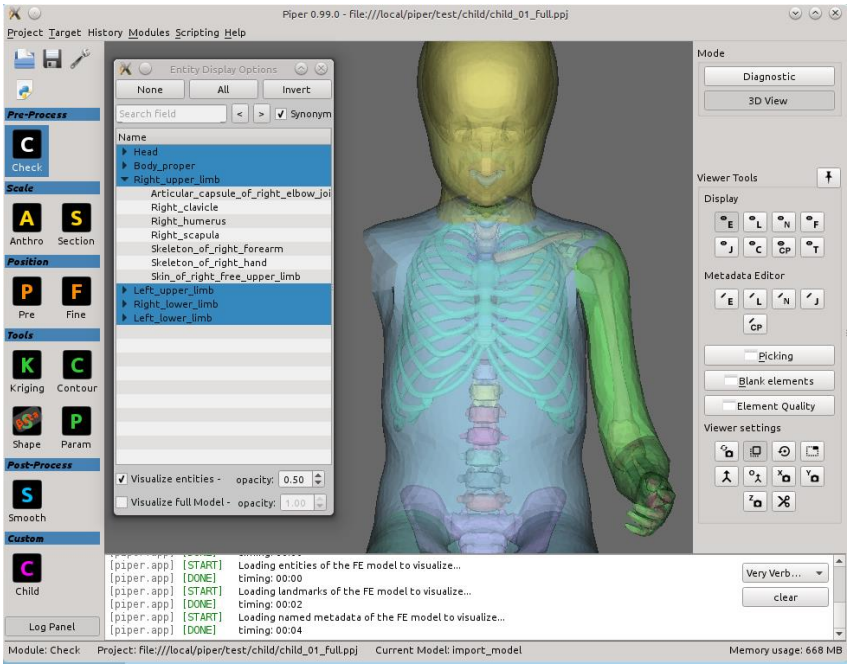
- Once imported, the HBM is stored in a structure that is called the PIPER model. It can be updated by any of the modules, saved (in the PIPER format) and exported back to the FE format (update).
- The target, which is what the user would like to achieve in terms of scaling or positioning, can also be updated by the modules, for example to add more details. The target can be shared between modules, or exported in xml (.ptt file) and reused later.

3. PIPER application

The PIPER application allows executing scaling and positioning workflows in an interactive manner. It provides among others:

- A GUI (utilizing the Qt library) with menus to select the modules, options, select the input/output.
- A history of the model, with all intermediate versions of the PIPER model (allowing to undo a transformation, make comparisons etc.).
- A 3D display interface (using the VTK library) to show the HBM, the PIPER model, preview of the transformation results, etc. It includes functions to select interactively parts of the model, mask them, display various PIPER entities (bones, landmarks, frames), display element quality or variations of element quality between models (VTK definition). It is shared by all modules except the ones based on lightweight physics simulation as these have specific requirements and have their own display interface (based on the SOFA framework).
- A partial batch functionality and scripting ability through a python interpreter (which can access the PIPER model and update it) and an octave interpreter (which can only communicate through files with the application). Example scripts are provided with the application.

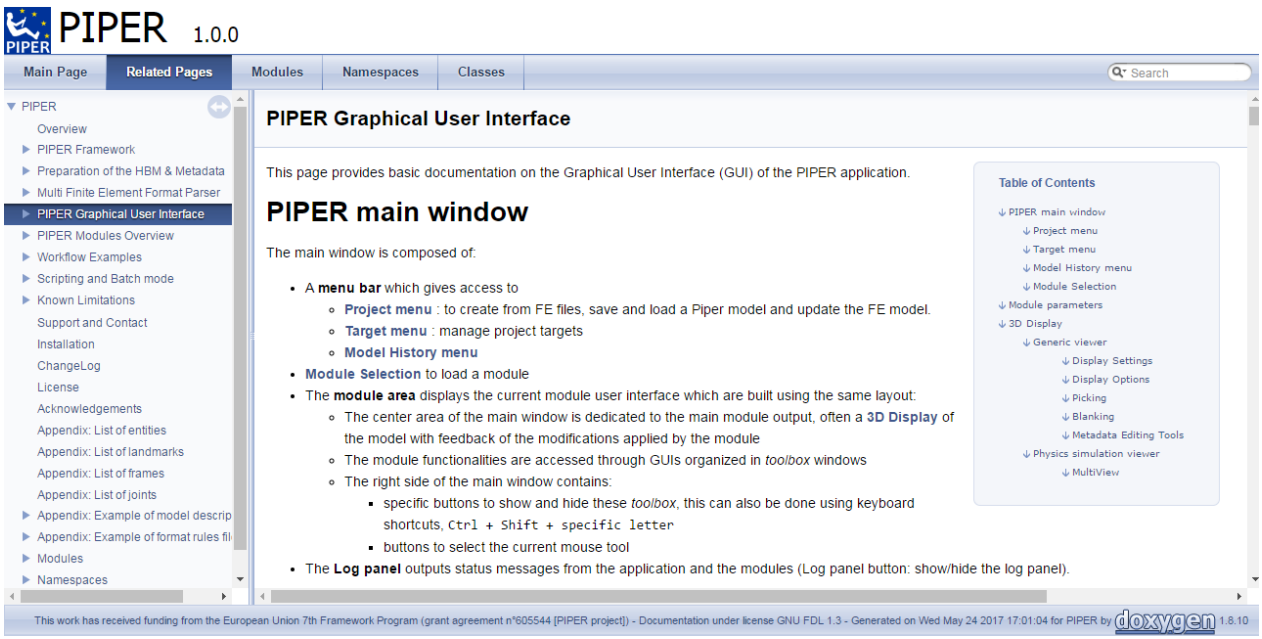
The application is available for Windows and Linux. Both executables and source code are provided. It is programmed mostly in the C++ language (according to ISO/IEC 14882:2011, i.e. the “C++11” standard) and uses many Open Source libraries. An illustration is provided below, but there are many others in the manual.



4. Documentation and resources

4.1 User manual

The user manual (Gnu documentation license) is built using Doxygen. It is provided with the application in an html format (illustration below) and also as a pdf document. In its current form, the pdf manual include 330 pages. Sections include a general description followed by more detailed explanations about the choices. Appendices include definitions of terms from anatomyDB.



4.2 Other resources

Information which is complementary to the manual is provided on the PIPER wiki page (https://gitlab.inria.fr/piper/application/) including:

- Tutorials written by WP1 users for various modules, adult and child models…
- Guidelines / FAQ, with advices on how to perform specific tasks or avoid some errors (written by WP1 users)
- Compilation guidelines
- …

Some of these materials are likely to move to the manual in the future.

All presentations made during the final PIPER workshop were recorded, edited and are available online on the PIPER project YouTube channel (https://www.youtube.com/channel/UC7cpPUIiJM5D-UZ7G3PNzcA). They include explanations about the framework philosophy, metadata, implementation, most modules, licensing scheme, as well as live demos.

5. Appendix: user manual

PIPER

1.0.0

Generated by Doxygen 1.8.10

Wed May 24 2017 16:59:59

Contents

1 Overview

The PIPER software framework was developed to help with the positioning and the personalization of Human Body Models (HBM) for injury prediction to be used in road safety. These HBM are typically available in one size and one posture (which can be difficult or time consuming to change), and they are implemented in commercial explicit Finite Element (FE) codes such as Ld-Dyna3D (LSTC), Pamcrash (ESI), Radioss (Altair) or Abaqus.

The framework aims to be modular, and model and code agnostic. More specifically, the idea is to be able to apply the same positioning, scaling or personalizing methodologies to several models in different codes, resulting in models that can be used in simulations with little or no correction. For this, the framework handles the import and export of the model, and model transformation methodologies are implemented in reusable modules. In order

to facilitate the real time user interactions, the PIPER framework uses only use geometric or lightweight physics approaches for the modules transforming the FE model.

In practice, the import, export, and most modules developed up to now are included in a main application that also provides a GUI, a 3D display of the model and a Python scripting interface. As it is Open Source, the framework and application uses many other open source libraries. The framework can easily be extended by adding modules or through scripting. The software was developed as part of the PIPER European Project.

A schematic overview of the software structure is provided below.

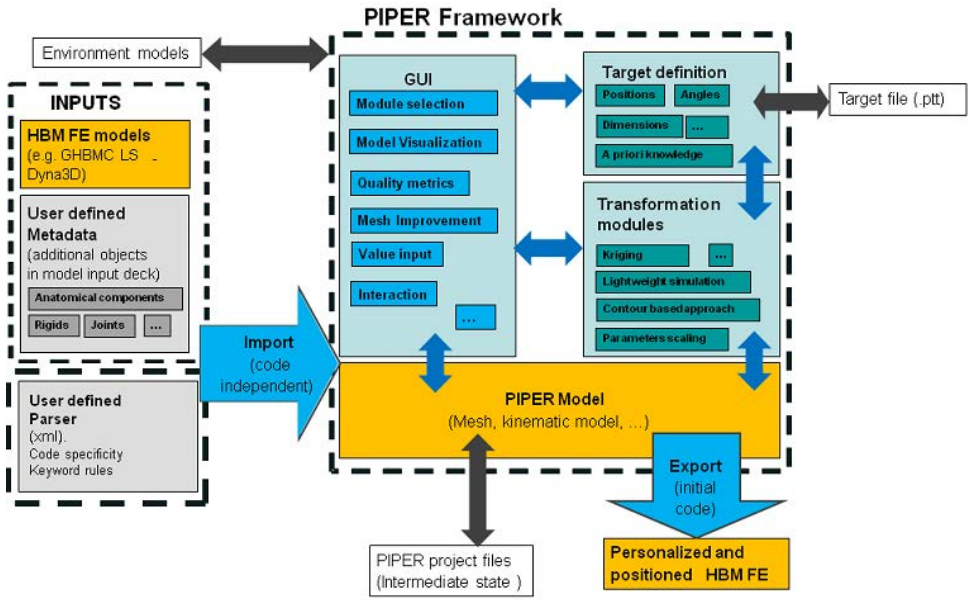


Figure 1: Overview of the software structure

The user manual is divided in the following sections:

- PIPER Framework
- Preparation of the HBM & Metadata
- Multi Finite Element Format Parser
- PIPER Graphical User Interface
- PIPER Modules Overview
- Known Limitations
- Support and Contact
- ChangeLog
- License

2 PIPER Framework

2.1 Introduction and basic concepts

In order to be able to transform different finite element Human Body Model (HBM) using the same methodology, different types of concepts were formulated and corresponding files format created. While this can be intimidating for new users, understanding at least the basic concepts is really beneficial to use the application. The basic concepts are not specific to the PIPER framework and are needed in any scaling or positioning process.

- **HBM interpretation and description:** there is typically nothing in the FE format linking an entity to an anatomical structure. For example, while there are keywords indicating what an airbag or a spotwelt is, there is no keyword to indicate that something is a bone, a femur, or the head. Model authors make their best to use descriptive names but there is no standard for that either. It is therefore desirable to associate some of the FE entities to anatomical concepts as these are useful for scaling or positioning (e.g. a bone is not expected to deform during positioning to the contrary of the skin). This association between the FE entities and anatomical entities is made through Metadata and only needs to be done once for a given model. To try to standardize a little bit the names of the anatomical concepts, a database containing vocabulary and relationships between anatomical entities is also provided with PIPER (AnatomyDB).

- **what needs to be described and imported by PIPER?** It depends on what you are trying to do but typically, not everything. A few pointers:

  - For a simple geometrical transformation, only the FE node positions need to be updated. These can be easily read once PIPER knows how they are stored in the FE input format (typically a node number and coordinates). This is achieved using parsing rules that can be created and edited by the user to add functionality (see Multi Finite Element Format Parser). Knowing where the nodes are in the FE input is also used to update their position at the end of the process. Basic geometry read for FE input files typically includes nodes and elements (to be able to display the model). This description only needs to be done once per FE solver.
  - For positioning, it seems desirable to describe the bones (so that they are not deformed) and their relationships (skeletal structure, joints). For scaling, it also seems desirable to describe the body structure such that the various body segments characteristics (length, section) can be updated separately.
  - Overall, what needs to be described depends on each transformation module. Minimum requirements to be able to use a module are provided in the documentation of each module. The information is only defined once and shared between modules.

- **where is the data that is imported stored?** What is imported is an anatomical interpretation of a Human Body Model including information required for positioning or scaling. As it is a model of its own, we called it a PIPER model (or PIPER HBM, PIPER Model). It is build dynamically every time a FE HBM is imported. The PIPER model can be accessed and updated by all modules and also by the scripting interface. It is not needed to export the FE model after each transformation (the PIPER model just gets updated), and the PIPER model can even be saved.

- **how is the transformation specified?** For any transformation, it is needed to define a source and target. In broad terms, the target is what the user would like to obtain (a position, a stature, a specific dimension, ...). However broad terms are typically not sufficient to fully define the transformation. In PIPER, the target is a collection of various types of information (e.g. age, stature, anthropometric dimension, joint angle, bone position, control point position) that are used to specify and drive the model transformation. The target information is accessible by all modules and it can be saved to be reused later. The PIPER framework and modules can help the user define the target with more specificity:

  - the target can be augmented by using *a priori* knowledge such as anthropometric regressions, or spinal postural preferences
  - a target can be adjusted interactively by the user (e.g. visual positioning or visual section adjustment)

It must be noted that these various concepts are also present in previous scaling or positioning approaches in the literature but that they are not formulated explicitly (e.g. a morphing Matlab script reads a specific model and scale it to a target). By formulating them, it is hoped that the PIPER framework can help using the same methods for different models, and help with the reproducibility of the transformation.

2.2 PIPER Files

This section provides information on the files used in PIPER:

- **Finite Element Model:** The Finite Element Model is the Human Body Model (HBM) in its original vendor format (e.g. : LS-DYNA ©, Pam-Crash ©, RADIOSS ©).

- **PIPER Model:** The PIPER model is the model saved by the PIPER application in its own xml format. Two associated files constitute the PIPER model:
  1. `.vtu` file: stores nodes and elements of the model in VTK Unstructured grid format.
  2. `.pmd` file: stores other components of the PIPER model in xml-based format.

- **PIPER Project:** when a project is saved, the followings files are created:
  1. `.ppj` file: the piper project file. This xml-based file contains:
     - Reference to the PIPER model saved with the project
     - The description of target(s) for positioning if exists
     - The description of body dimensions(s) for personalizing if exists
     - Reference to environments files if exists
  2. the PIPER model (`.vtu` and `.pmd`)

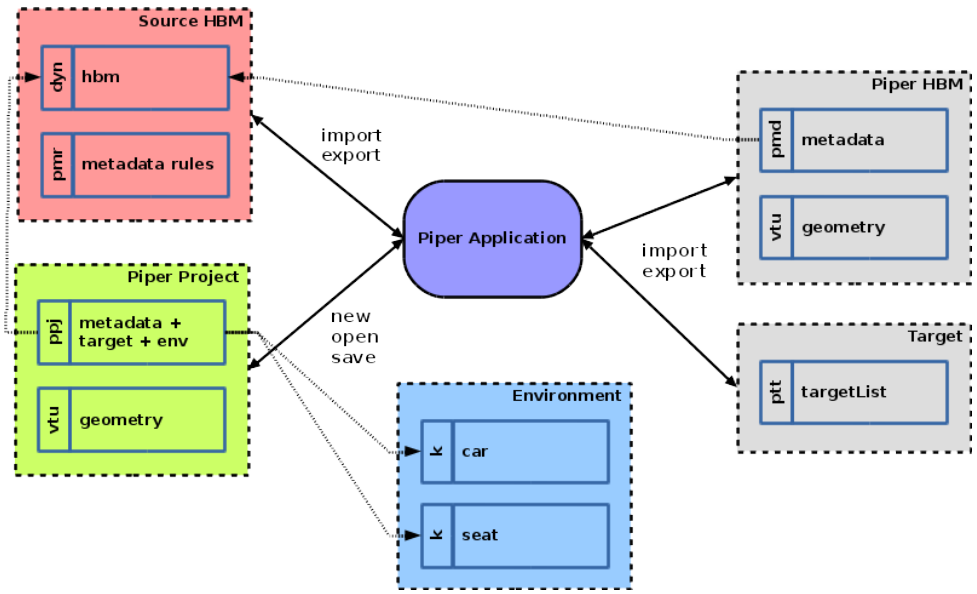


Figure 2: Overview of files manipulated by the PIPER application

2.3 PIPER Model

The PIPER model is composed of various components defined for their relevance in the positioning or scaling process. They do not aim to reproduce exactly the structure of the FE model but only to capture the different types and anatomical entities that need to be transformed in the process. A PIPER model contains both Geometry and specific Metadata.

2.3.1 Geometry

The following components are defined to describe the geometry and the structure of the PIPER Model:

- node components: correspond to nodes in the Finite Element Model.
- element components: correspond to elements in the Finite Element Model with distinction between one-dimensional, two-dimensional and three-dimensional element type.
- coordinate system: A local coordinate system defined in the PIPER model.
- group components: correspond to groups of nodes, groups of elements by type (1D/2D/3D) or groups of groups.

2.3.2 Metadata

The metadata are used to describe anatomical structure and functional constraints of the finite element model.

2.3.2.1 Entity

An *entity* defines an anatomical structure of the Human Body Model (bones, organs....). An entity corresponds to a set of group of elements (or group of groups) defined in the Finite Element Model. Naming of the entities must respect the PIPER anatomy database of entities (Appendix: List of entities), and the nature of the entity (skin, bone, ligament,...) and other information are deduced from this database. This dictionary is partially based on the FMA ontology (http://sig.biostr.washington.edu/projects/fm).

2.3.2.2 Joint

A *joint* can be used to describe the articulation between two entities. It is mainly used within the concept of positioning. An entity joint is defined by:

- names of entities involved in the joint.
- a coordinate system associated with each entity involved in the joint.
- the definition of degree of freedom of the joint.

Notes: When joints between bones are defined as generalized joints in the HBM, it could be desirable to use the same definition in the metadata (same frames, etc).

2.3.2.3 Anatomical Joint

An *anatomical joint* is similar to a *joint*. It is defined by its name which must be in Appendix: List of joints and its degree of freedom. From the name the following anatomical knowledge is deduced :

- the entities involved in the joint
- the coordinate system associated with each entity are computed from landmarks (see Appendix: List of frames)

2.3.2.4 Landmark

A *landmark* is used to describe a specific anatomical location (that could be found in a reproducible manner by someone with anatomical knowledge). Three types of landmark can be defined:

- type point: defined by 3D coordinates.
- type sphere: defined by the center of the least squares sphere deduced from a group of nodes.
- type barycenter: defined by the barycenter of a group of nodes.

The naming of the landmark must respect the PIPER anatomy Database of Landmark (Appendix: List of landmarks). Among others, the anatomy database is usefull:

- to know which entity (e.g. bone) the landmark should be associated with,
- to compute anatomical frames (Appendix: List of frames),
- to compute Anatomical Joint,
- etc... If needed, the database could be extended (see Anatomy Database).

2.3.2.5 Contact

A *contact* is defined between two entities. the contact can be defined as a "sliding" contact or an "attached" contact.

Note: corresponding FE concepts would be sliding without separation and tied.

2.3.2.6 Control Points

It defines a set of control points to be used in the geometrical interpolation/morphing module (Kriging Module).

2.3.2.7 Model Parameter

A *parameter* is a scalar value that corresponds to a property in the Finite Element model. It can be a material law parameter, a part property, a shell thickness, ...

2.3.2.8 Named Metadata

A *named metadata* defines a set of model geometrical elements (nodes, 2d, 3d elements) and associates a *name* to it. It is used in specific modules such as Pre-Positioning Module or Contour Deformation Module . This approach allows defining metadata that does not fit the definition of the previous subsections (e.g. not an anatomical landmark) but that a module can recognize through its name.

2.4 Module

2.4.1 Inputs/Outputs

A module provides a functionality in the PIPER application. It has a well defined interface:

- the **parameters** of its numerical methods
- the **metadata** it needs
- the **input targets** it can load
- the **output targets** it can produce
- whether it can modify the model, only the nodes coordinates or the full geometry
- the other **input**/**output** it can deal with

2.4.2 Workflow

To perform a complete task the user can run several modules in sequence, save intermediate results (targets or model) to work with later in the application or using the Batch mode

A typical workflow is:

1. import a HBM from its vendor format files (see Project menu)
2. check and explore the imported model with the Check Module
3. produce some anthropometric targets using the Anthropo Module
4. scale the HBM according to these targets using the Kriging Module
5. position the HBM in an environment using the Pre-Positioning Module and produce bone positions and landmarks targets
6. deform the HBM using the Fine-Positioning Module using the bone positions targets and update the model nodes position
7. or deform the HBM using the Contour Deformation Module using the landmarks targets and update the model nodes position
8. check the element quality, and improve it if needed using the Smoothing module
9. export the HBM to its original vendor format files (see Project menu)

At any time in his workflow, the user can run custom operations by Writing a script

2.5 Model history

In the "Model History" menu:

- the current model can be renamed
- the baseline model ("the reference") can be imported as the root of the history. This re-imports the FE model in the PIPER application
- the list of history content is provided with the current model in bold. The selection of another model in history makes it current.

Each time the human body model is transformed (for positioning or personalizing), the transformed model is added in the history and is set as the current model.
Be aware that when a new model is loaded (either by loading a new PIPER project or by importing a new model), all the current history entries are deleted. Also, models cannot currently be deleted from the history.

2.6 Target

3 Preparation of the HBM & Metadata

3.1 Introduction

The followings steps are typically required to import a Finite Element Human Body model (HBM) in the PIPER application:

- Step 1: check that a format rules file (.pfr) exists for the considered finite element format (if not, see Multi Finite Element Format Parser to define a new one). This is required to parse the finite element files.
- Step 2: prepare the Finite Element Human Body Model. Typically, the preparation consists in creating groups of finite element entities corresponding to anatomical or PIPER components. For example, the user could create a group of all FE parts constituting the femur. These are stored in the native finite element format.
- Step 3: define model metadata in the description model file (Preparing the model and metadata). Typically, groups prepared in the previous step are associated to PIPER entities (.pmr file).

Step 2 and Step 3 are detailed after the summary table below.

Summary of files needed to import a FE HBM and build the internal representation of the HBM (PIPER model)

Contents	Format	Defined
Rules to parse the FE format (1)	.pfr (XML based)	once per FE format
Groups corresponding to anatomical or PIPER components	native FE format	once per HBM
Association between PIPER components and FE groups/entities	.pmr (XML based)	once per HBM

(1) available pfr files are stored in the Piper/bin directory

3.2 Preparing the model and metadata

The Model description file (xml based format with extension .pmr) is defined to:

- provide information about model units, location of file to be parsed and identification of Format rule file to be used for parsing
- describe the structure of the Human Body Model according to the description of the Metadata of the model. The following PIPER components can be described:

- "entity" element
- "joint" element
- "landmark" element
- "Control Points" element
- "parameter" element

If groups or FE entity are created specifically to be associated with these PIPER components, it is suggested to store them in separate include files of the FE format (see the example of the child model).

3.2.1 Structure of the model description file

A typical overall structure of a model description file (.pmr) is shown below, with detailed descriptions of the components in the following subsections. The pmr file of the PIPER child model can also be used as an example.

```
1 <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2 <!DOCTYPE format_description SYSTEM "ModelRules.dtd">
3
4 <-- Root element of the description document -->
5 <model_description>
6
7     <-- model units (required) -->
8     <units length="mm" mass="kg" age="year" />
9     <-- source file or source directory (required) -->
10     <source>model_01_separator.dyn</source>
11     <-- source format rule file (required) -->
12     <format_rules format="LSDyna_separator">../formatrules/Formatrules_LSDyna_Separator.pfr</format_rules>
13
14
15     <-- define entity: repeat for each entity in the model -->
16     <entity name="entityname">
17     ...
18     </entity>
19
20     <-- define landmark: repeat for each landmark in the model -->
21     <landmarks name="landmarkname" type="landmark type">
22     ...
23     </landmarks>
24
25     <-- define a set control point: repeat for each joint in the model -->
26     <controlPoint name="control_point_0" role="source">
27     ...
28     </controlPoint>
29
30     <-- define joint: repeat for each joint in the model -->
31     <joint name="jointname">
32     ...
33     </joint>
34
35     <contact name="contactname" type="sliding">
36     ...
37     </contact>
38
39     <-- define parameter: repeat for each parameter in the model -->
40     <parameter name="parametername">
41     ...
42     </parameter>
43
44 </model_description>
```

3.2.2 "model units" element

Length units: m, dm, cm, mm
Mass unit: kg, dg, cg, mg
Age unit: year, month

```
1 <-- model units (required) -->
2 <units length="mm" mass="kg" age="year" />
```

3.2.3 "source" element

The path of the main file to be imported to create the piper HBM need to be specified as follow. The path can be absolute or relative:

```
1 <-- source file or source directory (required) -->
2 <source>model_01_separator.dyn</source>
```

3.2.4 "Format" element

For the LSDyna and pamCrash FE formats, format rule files are provided with PIPER application, so only format name is required. The following format names are available;

- LSDyna_fix: for FE files in LSDyna fix (or block) format

```
1 <-- source format rule file (required) -->
2 <format_rules format="LSDyna_fix"/>
```

- LSDyna_separator: for FE files in LSDyna format with comma separator

```
1 <-- source format rule file (required) -->
2 <format_rules format="LSDyna_separator"/>
```

- PamCrash: for FE files in pamcrash format

```
1 <-- source format rule file (required) -->
2 <format_rules format="PamCrash"/>
```

If format rules required for parsing is not one of those provided with PIPER application, the format rule path (relative or absolute) can be specified:

```
1 <-- source format rule file (required) -->
2 <format_rules format="myFormatName">../myformatrules/Formatrules_myformat.pfr</format_rules>
```

3.2.5 "FE component identification" element

Depending on the FE format of the input Human Body model, a FE component (nodes, groups, material...) is associated with an unique identifier which can be an integer or a name. To specify which FE component is involved in a association, the "keyword" element is used as follows (example considering LS-Dyna format and Pamcrash format):

```
1 <-- FE component identified by an integer identifier (example for LS Dyna format):  -->
2 <keyword kw="*SET_SOLID">
3     <id>10</id>
4 </keyword>
5 <-- FE component identified by a name (example for Pam-Crash format) -->
6 <keyword kw="GROUP">
7     <name>femur</name>
8 </keyword>
```

In the first example, the FE component corresponding to the keyword ∗SET_SOLID (a group of solid elements in LS-Dyna) with an identifier equal to 10 is selected and used for the association rule. In the second one, the FE component corresponding to the keyword GROUP (a group in Pam-Crash) with an identifier name equal to groupname is selected and used for the association rule.

3.2.6 "entity" element

Entities are typically used to describe anatomical entities such as bones, skin, ligaments, etc. The list of available anatomical entities is provided in Appendix: List of entities .

The "entity" element defines association between group(s) of elements (or group(s) of part) in the FE Human Body Model and a Entity identified by `name` .

Example of "entity" element, the "entity" named Entity_1 is associated with FE components identified by:

- the keyword "kw_for_group_of_element3d" and Id 10 and 101
- the keyword "kw_for_group_of_element2D" and Id 5

```
1 <entity name="Entity_1">
2     <keyword kw="kw_for_group_of_element3d">
3         <id>10 101</id>
4     </keyword>
5     <keyword kw="kw_for_group_of_element2D">
6         <id>5</id>
7     </keyword>
8 </entity>
```

3.2.6.1 Model preparation for entities

The user has to first define group(s) of elements or a group(s) of parts in the Finite Element Model using their finite element pre-processor. All elements composing the anatomical entity must be included. For example, if the femur is modelled with 3d element for trabecular bone with an overlay of 2D elements for cortical bone, group(s) must include all 3D and 2D elements.

If 2D elements are included in the entity definition, normal orientation and topology must be carefully checked (the envelope should be closed, manifold, oriented coherently with normals pointing out). If this is not the case, the user can either modify their HBM or, as a workaround, add a part composed with null elements respecting these criteria to describe the outside boundary of the entity. The PIPER child model is an example of model with properly oriented normals.

If the original Finite Element format allows defining a group name, association between group and entity can be made in "auto" mode if the group name correspond to an entity name following PIPER dictionary.

To be able to use the auto mode, the group name must be parsed in the description of the rule to parse keyword corresponded to group in the considered FE format (Multi Finite Element Format Parser).

Example of "entity" element with "auto" mode: In this example, two FE group components identified by the keyword "kw_for_group_of_element3d" and ids 10 and 101 are considered and the name defined for each group is used to identified Entity to be associated with. If the group name identified by keyword "kw_for_group_of_element3d" and ids 10 is Left_femur, then this group is associated with Entity left femur.

```
1 <entity name="auto">
2     <keyword kw="kw_for_group_of_element3d">
3         <id>10 101</id>
4     </keyword>
5 </entity>
```

3.2.7 "joint" element

Joints are typically used to articulate two bones with a robotic joint (which can match an anatomical convention or not).

To define a joint between two piper components, a frame is associated with each entity involved. This can be achieved using the "setFrame" element that identify the proper FE frame component using the "keyword" element. The "setDof" element is defined to specify degrees of freedom of the joint (relatively to frame defined for entity_1 of the frame).

Example 1:

```
1 <joint name="Joint_1">
2     <entity_1 name="Entiyname1">
3         <setFrame>
4             <keyword kw="kw_frame">
5                 <id>60010</id>
6             </keyword>
7         </setFrame>
8     </entity_1>
9     <entity_2 name="Entiyname2">
10         <setFrame>
11             <keyword kw="kw_frame">
12                 <id>5050</id>
13             </keyword>
14         </setFrame>
15     </entity_2>
```

```
16     <setDof>0 0 0 1 0 1</setDof>
17 </joint>
```

In the example, a Joint named `Joint_1` is defined between entity `Entiyname1` and `Entiyname2`. Frame defined in the keyword "kw_frame" with id equal to 60010 is associated with `Entiyname1` and Frame defined in the keyword "kw_frame" with id equal to 5050 is associated with `Entiyname2`. The degree of freedom of the joint are rotation relative to the X axis of the `Entiyname1` frame and rotation relative to the Z axis of the `Entiyname1` frame.

The global frame axis of the FE model can be used to define a joint frame (see example 2)
. **Example 2:**

```
1 <joint name="Joint_1">
2     <entity_1 name="Entiyname1">
3         <setFrame type="global">
4             <keyword kw="*NODE">
5                 <id>2056046</id>
6             </keyword>
7         </setFrame>
8     </entity_1>
9     <entity_2 name="Entiyname2">
10         <setFrame type="global">
11             <keyword kw="*NODE">
12                 <id>2056046</id>
13             </keyword>
14         </setFrame>
15     </entity_2>
16     <setDof>0 0 0 1 0 1</setDof>
17 </joint>
```

In example, a Joint named `Joint_1` is defined between entity `Entiyname1` and `Entiyname2`. Frame defined in the keyword "kw_frame" and its orientation are identical to the global model frame (`type="global"`). The origin of the joint frame is defined by setting a node id. The degree of freedom of the joint are rotation relative to the X axis of the `Entiyname1` frame and rotation relative to the Z axis of the `Entiyname1` frame.

3.2.7.1 Model preparation for joint

When springs or 6 d.o.f. joints are already used in the FE model to articulate bones, it is suggested to use that definition as the basis for the joint entity.

3.2.8 "landmark" element

Landmarks are typically

The "landmark" element is used to associate FE groups of nodes and a Landmark identified by its `name` and `type` (point, barycenter, sphere). In the following example, a landmark with `name` Landmark_2 is defined by the group of node identified by the "keyword" element.

```
1 <landmarks name="Landmark_2" type="sphere">
2     <keyword kw="ground_node_keyword">
3         <id>10002</id>
4     </keyword>
5 </landmarks>
```

3.2.8.1 Prepare HBM Landmarks

To define a landmark, the user has to define a group of nodes in the Finite Element Human Body Model in the Finite Element Model using their finite element pre-processor. The node number can also be used directly if only one node is needed.

If the original Finite Element format allows to define a group name, association between group and landmark can be made in "auto" mode if the group name correspond to an landmark name following the AnatomyDB definition for landmarks. To be able to use the auto mode, the group name must be parsed in the description of the rule to parse keyword corresponded to group in the considered FE format (see Multi Finite Element Format Parser).

3.2.9 Contact element

In order to describe a contact between *entity1* and *entity2*, groups of nodes have to be defined to identify the contact region on both entities. In theory a contact is symmetric, but the way it is implemented in the Pre-Positioning Module it is not. *entity1* shall be the entity which has its contact region always in contact.

Contacts are defined between groups of nodes. Their property includes a thickness parameter which can maintain the contact between the objects. An example of contact is provided below.

```
<contact name="Left_hip" type="sliding">
    <thickness keep="true" />
    <entity_contact_1 name="Pelvic_skeleton">
        <setGroup>
            <keyword kw="*SET_NODE_LIST_TITLE">
                <id>9903818</id>
            </keyword>
        </setGroup>
    </entity_contact_1>
     <entity_contact_2 name="Left_femur">
         <setGroup>
             <keyword kw="*SET_NODE_LIST_TITLE">
                 <id>9901621</id>
             </keyword>
         </setGroup>
     </entity_contact_2>
</contact>
```

3.2.10 "parameter" element

The parameter element is typically used for finite element parameters which are not described by nodes or elements (e.g. material properties, shell thickness).

The "hbm_parameter" element allows to gather parameters parsed with keyword format rules (PIPER Parameter operation) in a set. The `parameternamesource` corresponds to the name of the parameter defined in the rule to parse each keywords.

```
<hbm_parameter name="parametername" source="parameternamesource">
    <keyword kw="kw1">
        <id>7000014 7000016</id>
    </keyword>
    <keyword kw="kw2">
        <id>7000015</id>
    </keyword>
</hbm_parameter>
```

There is no specific model preparation for this element.

3.2.11 "Control Points" element

To define an ordered set of control points to be used as source control points in Kriging Module. Each set of control points must have a name (any unique string) and role ("source" or "target") attributes. There are several allowed formats for defining the coordinates of the control points, as shown on the example below: they can be defined either by a keyword or directly as a set of point coordinates, with or without IDs. Regardless of the coordinate definition, each set can also have a weight (Nugget) and a Skin/bone association parameters, as shown on the "control_points_2" set.

```
<controlPoint name="control_point_0" role="source">
    <keyword kw="*NODE">
        <id>5207268 33002334 33003064</id>
    </keyword>
</controlPoint>
<controlPoint name="control_point_1" role="source">
    <keyword kw="*NODE">
        <id>6120127 5106751 5106946 35000357</id>
    </keyword>
</controlPoint>
<controlPoint name="control_point_2" role="source">
    <coord>
    23        -43.3839         20.0958         225.346
    24         -65.291               0         229.255
```

```
15     25        -21.7513              0         222.236
16     26        -43.3839        -20.0958        225.346
17     27        -26.8911              0         193.384
18     </coord>
19     <weight>-20 -20 -20 -45 -45</weight>
20     <as_bones>1 1 1 0 0</as_bones>
21     <as_skin> 0 0 0 1 1</as_skin>
22 </controlPoint>
23 <controlPoint name="control_point_3" role="target">
24     <coordfix>
25     -10 20 30
26     -20 30 40
27     -10 38 74
28     -20 12 11
29     -12 23 42
30     </coordfix>
31 </controlPoint>
```

4 Multi Finite Element Format Parser

4.1 Introduction

A dedicated XML-based language is used to define the rules needed to parse the Finite Element Model file format. The approach can be used with multiple formats and codes and has been successfully tested against LS-dyna and Pamcrash models. It was selected because it allows the user to implement additional rules for other codes or numerical features without recompiling the software. This is an alternative to a full templates (the only required rules are those needed to build the PIPER model for a given transformation).

Format rule file : in this file (xml based with extension .pfr), general information and rules to parse the finite element format are defined. In the release, rule format files are provided for LS dyna (for both fixed sized and comma separated format) and Pamcrash. The files (with a pfr extension) are stored in the Piper/bin directory. In principle, it is possible to define Format rules file for other finite element formats and use them in the PIPER application as long as:

- the finite format is keyword based
- each component is identified by a id and/or a name.

This section describes how to define rules to add support for additional keywords in the provided files, or to develop new files for another Finite Element format.

Note: this rule file is not specific to a finite element Human Body Model. If a rule file already exists for the finite element format used by your HBM, it is likely not needed to modify it even if you develop new metadata.

4.2 Format rule file

The format rule file define rules to parse keywords for a specific Finite Element format. The xml file is organized under the root element "format_description" as follows:

- "formatInformation" element
- "meshComponent" element
- "rule" element

Description of the general structure of the parser file (Text between <– and –> is comment):

```
1 <-- XML declaration -->
2 <?xml version="1.0" encoding="UTF-8"?>
```

```
3 <!DOCTYPE format_description SYSTEM "ParsingRules.dtd">
4
5 <-- Root element of the parser document -->
6 <format_description>
7     <-- formatInformation element of the parser document: Repetition 1 -->
8     <-- This element declares some basic information on the format -->
9     <formatInformation format="FormatName">
10     ....
11     </formatInformation>
12
13     <-- meshComponent element of the parser document: Repetition 1 -->
14     <-- This element declares association between keyword and PIPER component -->
15     <meshComponent>
16     ....
17     </ meshComponent >
18
19
20     <-- rule element of the parser document: Repetition once per keyword to parse -->
21     <rule keyword="kw ">
22     </ rule >
23 </format_description>
```

The two first lines are mandatory. The first one is requested for the XML format and the second one allows to check the syntax in the parser file.

4.2.1 "formatInformation" element

The `formatInformation` element has an attribute used to declare a `name` describing the format (mandatory) and is composed of:

- **"comment" element:** the character used in the FE format for comment line. Multiple characters can be defined (separated by whitespace).
- **"include" element:** the keyword used to declare include files in the FE format.
- **"separator" element:** the character used as a separator. If a whitespace is used as a separator in the FE format, it should be declared using "ws".

Example of `formatInformation` element for Pamcrash. In this format, two different characters can be placed at the beginning of comment line.

```
1 <formatInformation format="PamCrash">
2     <comment>$ #</comment>
3     <include>INCLU</include>
4 </formatInformation>
```

Example of `formatInformation` element for LS-Dyna using comma as separator.

```
1 <formatInformation format="LSDyna_separator">
2     <comment>$</comment>
3     <include>*INCLUDE</include>
4     <separator>,</separator>
5 </formatInformation>
```

4.2.2 "meshComponent" element

The `meshComponent` element defines relation between keywords of the FE format and a PIPER component. For each keyword defined in this element, a "rule" element must be defined.
Components defined in this element are:

- **"componentNode" element:** Piper node
- **"componentElement1D" element:** Piper one-dimensional element

- **"componentElement2D" element:** Piper two-dimensional element
- **"componentElement3D" element:** Piper three-dimensional element
- **"componentGNode" element:** Piper group of nodes
- **"componentGElement1D" element:** Piper group of one-dimensional element
- **"componentGElement2D" element:** Piper group of two-dimensional element
- **"componentGElement3D" element:** Piper group of three-dimensional element
- **"componentGGroup" element:** Piper group of group
- **"componentFrame" element:** Piper Frame
- **"componentModelParameter" element:** Piper parameter

In addition, the element ordering has to be specified in case the one in the considered FE format differs from the one chosen in Piper project (VTK definition adopted VTK documentation (http://www.vtk.org/wp-content/uploads/2015/04/file-formats.pdf)).
`elementOrdering` element can be used for this purpose. It describes the sequence ordering used to convert FE element definition to the Piper one for the following types:

- tri: triangle element (2D)
- quad: quadrangle element (2D)
- tetra: tetra element (3D)
- pyra: pyramidal element (3D)
- penta: wedge element (3D)
- hexa: hexahedron element (3D)

Example of `meshComponent` element:

```
<meshComponent>
    <componentNode>kw kw</componentNode>
    <componentElement1D>kw kw</componentElement1D>
    <componentElement2D>kw kw</componentElement2D>
    <componentElement3D>kw kw</componentElement3D>
    <componentGNode>kw kw</componentGNode>
    <componentGElement1D>kw kw</componentGElement1D>
    <componentGElement2D>kw kw</componentGElement2D>
    <componentGElement3D>kw kw</componentGElement3D>
     <componentGGroup>kw kw</componentGGroup>
     <componentFrame>kw kw</componentFrame>
     <componentModelParameter>kw kw</componentModelParameter>
     <elementOrdering>
         <penta>...</penta>
         <hexa>...</hexa>
     </elementOrdering>
</meshComponent>
```

Example of `meshComponent` element for Pamcrash © format:

```
<meshComponent>
    <componentNode>NODE</componentNode>
    <componentElement1D>BEAM SPRING/</componentElement1D>
    <componentElement2D>SHELL</componentElement2D>
    <componentElement3D>SOLID</componentElement3D>
    <componentGNode>GROUP</componentGNode>
    <componentGElement1D>GROUP</componentGElement1D>
    <componentGElement2D>GROUP</componentGElement2D>
    <componentGElement3D>GROUP</componentGElement3D>
     <componentGGroup>GROUP</componentGGroup>
```

```
11     <componentFrame>FRAME</componentFrame>
12     <componentModelParameter>MATER PART</componentModelParameter>
13     <elementOrdering>
14         <penta>1 2 5 4 3 6</penta>
15     </elementOrdering>
16 </meshComponent>
```

Example of `meshComponent` element for LS-DYNA © format:

```
1 <meshComponent>
2     <componentNode>*NODE</componentNode>
3     <componentElement1D>*ELEMENT_BEAM *ELEMENT_DISCRETE</componentElement1D>
4     <componentElement2D>*ELEMENT_SHELL *ELEMENT_SHELL_THICKNESS *ELEMENT_SHELL_BETA</componentElement2D>
5     <componentElement3D>*ELEMENT_SOLID</componentElement3D>
6     <componentGNode>*SET_NODE_LIST_TITLE *PART</componentGNode>
7     <componentGElement1D>*SET_BEAM_TITLE *PART</componentGElement1D>
8     <componentGElement2D>*SET_SHELL_LIST_TITLE *PART *SET_SHELL_LIST</componentGElement2D>
9     <componentGElement3D>*SET_SOLID_TITLE *PART</componentGElement3D>
10     <componentGGroup>*SET_PART_LIST_TITLE *PART</componentGGroup>
11     <componentFrame>*DEFINE_COORDINATE_NODES *DEFINE_COORDINATE_SYSTEM_TITLE *DEFINE_COORDINATE_NODES_TITLE
      </componentFrame>
12     <componentModelParameter>*MAT_ELASTIC_TITLE *MAT_PLASTIC_KINEMATIC_TITLE *
      MAT_STRAIN_RATE_DEPENDENT_PLASTICITY_TITLE *ELEMENT_SHELL_THICKNESS *ELEMENT_SHELL_BETA</componentModelParameter>
13     <elementOrdering>
14         <penta>1 2 5 4 3 6</penta>
15     </elementOrdering>
16 </meshComponent>
```

4.2.3 "rule" element

The "rule" element is used to define all operations to be performed when a previously registered `keyword` is found. For each keyword registered in "meshComponent" element, parsing actions are defined and parsed data are stored in variables (Variable manipulation). These variables are used to create and manipulate PIPER components of the model with PIPER Component operation.

```
1 <rule keyword="kw">
2     ...
3 </rule>
```

4.3 Define rules

4.3.1 Variable manipulation

The element "variableAssign" is used to assign the result of an operation to the variable identified by `name`. The element "variable" defines variable as an input of another element.

Usage of "variableAssign" and "variable" elements:

```
1 <variableAssign name="varname"/>
2
3 <variable name="varname"/>
```

The "append" element allows to add characters defined in `value` to a variable of type character.

Example of "append" usage:

```
1 <append value="_Node">
2     <variable name="GroupName"/>
3     <variableAssign name="GroupName_Node"/>
4 </append>
```

the "clearVar" element clear variable(s).

Example of "clearVar" usage:

```
<clearVar>
    <variable name="GroupName_Node"/>
    <variable name="GroupName_E1D"/>
</clearVar>
```

4.3.2 Parsing operations

4.3.3 "nextLine"" element

This elements go to the next line of the current line (without considering comment line according to comment character defined in‘"formatInformation" element).

4.3.4 "parse"" element

This element defines how to (1) parse the current line or variable of type character and (2) assign the result to a variable.

Options of the "parse" element:

- `format`: Type of data to be parsed. It can be C (characters), F (double), UI (integer), UIpos (strictly positive integer).
- `offset`: Offset in digit if no separator is defined either in `parse` element or “formatinformation” element. Number of separator if a separator is defined either in `parse` element or “formatinformation” element.
- `length`: Number of digit to parse. It must be zero if a separator is defined
- `separator`: Defines a local separator character specific to this `parse` element. Default is no local separator defined. If `length` equals 0 and a global separator is defined in "formatInformation" element, the global separator is used for parsing if no local separator is defined.
- `repeat`: repeat the parse operation considering the `offsetrepeat`. The result of parsing is accumulated in the variable as a vector.
- `offsetrepeat`: Offset in digit between repeated parse operations in case no separator is defined neither in `parse` element nor in "formatInformation" element . Default value is 0 if `length` is different from 0 and 1 if `length` equals 0.
  Or Number of separator between repeated parse operations if a `separator` is defined either in `parse` element or "formatInformation" element element.

Example 1: Parse in fixed length format.
String to parse (blanks are replaced by dot for visibility and parsed values are in bold)
.....1.5**.....0.06**.....-2.....0

```
<parse format="F" separator="none" repeat="1" offset="10" length="10">
    <variableAssign name="var1"/>
</parse>
```

As a result, the value 0.06 is assigned to the variable var1 type F.

Example 2: Parse with comma separator.
String to parse (parsed values are in bold)
1.5,**0.06**,-2,0

- If the separator for this line is different from the separator defined in "formatInformation" element element:

```
1 <parse format="F" separator="," repeat="1" offset="1" length="0">
2     <variableAssign name="var1"/>
3 </parse>
```

- Otherwise:

```
1 <parse format="F" repeat="1" offset="1" length="0">
2     <variableAssign name="var1"/>
3 </parse>
```

  As a result, the value 0.06 is assigned to the variable var1 type F.

Example 3: Parse in fixed length format with repetition.
String to parse (blanks are replaced by dot for visiblity and parsed values are in bold)
.7020553**.....0.546867.....-93.736504.....14.2672**

```
1 <parse format="F" separator="none" repeat="3" offset="8" length="16">
2     <variableAssign name="var1"/>
3 </parse>
```

The result is stored in var1 and var1 is a vector of 3 double numbers (0.546867,-93.736504,14.2672).

Example 4: Same as example 3 with comma separator.
String to parse (parsed values are in bold)
7020553,**0.546867**,**-93.736504**,**14.2672**

```
1 <parse format="F" separator="none" repeat="3" offset="1" length="0">
2     <variableAssign name="var1"/>
3 </parse>
```

The result is stored in var1 and var1 is a vector of 3 double numbers (0.546867,-93.736504,14.2672).

4.3.5 "parserule" element

This element is used to describe "parse"" element to be performed on the current line or variable of type character.
Example of a set of two parse actions on the current line:
line to parse (blanks are replaced by dot for visiblity)
.7020553.....0.546867.....-93.736504.....14.2672

```
1 <parseRule>
2     <curLine/>
3     <parse format="UI" separator="none" repeat="1" offset="0" length="8">
4         <variableAssign name="var1"/>
5     </parse>
6     <parse format="F" separator="none" repeat="2" offset="24" length="16">
7         <variableAssign name="var2"/>
8     </parse>
9 </parseRule>
```

As a result, value 7020553 is assigned to variable `var1` and `var2` is a vector with two elements (-93.736504,14.↩
2672).

Example of a set of two parse actions on variable varstr (this variable must be a variable of type C):
Content of variable varstr to parse (blanks are replaced by dot for visiblity)
.string_1.....string_2.....-93.736504.....14.2672

```
1 <parseRule>
2     <variable name="varstr"/>
3     <parse format="UI" separator="none" repeat="1" offset="0" length="8">
4         <variableAssign name="var1"/>
5     </parse>
6     <parse format="F" separator="none" repeat="2" offset="24" length="16">
7         <variableAssign name="var2"/>
8     </parse>
9 </parseRule>
```

As a result, string_2 is assigned to variable `var1` and `var2` is a vector with two elements (-93.736504,14.2672).

4.3.6 Conditional Statement

4.3.6.1 "dolf" statement

The “if” statement is used for conditional execution:

```
1 <doIf>
2     <condition>
3         <-- expression of the condition: see Boolean operation->
4         .....
5     </condition>
6     <body>
7         <-- expressions in body are executed if condition is true->
8         .....
9     </body>
10 </doIf>
```

4.3.6.2 "while" statement

The “while” statement is used to repeat execution as long as the condition is true:

```
1 <doWhile>
2     <condition>
3         <-- expression of the condition: see Boolean operation->
4         .....
5     </condition>
6     <body>
7         <-- expressions in body are executed if condition is true->
8         .....
9     </body>
10 </doWhile>
```

4.3.6.3 "forEach" statement

The “forEach” statement is used to iterate over elements of a vector variable:

```
1 <forEach>
2     <-- variable name to iterate on->
3     <variable name="iterable_variable"/>
4     <-- variable name of the iterator->
5     <variableAssign name="iterator_variable"/>
6     <condition>
7         <-- expression of the condition: see Boolean operation->
8         .....
```

```
9         </condition>
10     <body>
11     <-- expressions in body are executed if condition is true->
12     .....
13     </body>
14 </ forEach>
```

4.3.7 Boolean Operation

The following set of operations return a Boolean depending on the result of the evaluation.

- "find" operation
- "notfind" operation
- "equal" operation

4.3.7.1 "find" operation

This operation returns true if the set of characters defined in “value” is found either in the variable (variable must be a variable of type C) or in the current line being parsed. The option `pos` is used to specify if “value” can be at any position (pos="any") in the variable or only at the beginning (pos=”start”).

```
1 <-- true if value “str” is found at the beginning of the current parsed line->
2 <find value="str" pos="start"><curLine/></find>
3
4 <-- true if value “str” is found at any position in the variable varname->
5 <find value="str" pos="any"><variable name="varname"/></find>
```

4.3.7.2 "notfind" operation

This operation returns true if the set of characters defined in “value” is not found either in the variable nor in the current line being parsed. Syntax and options are the same as for the “find” operation.

```
1 <-- true if value “str” is not found at the beginning of the current parsed line->
2 <notfind value="str" pos="start"><curLine/></notfind >
3
4 <-- true if value “str” is not found at any position in the variable varname->
5 <notfind value=" str " pos="any"><variable name="varname"/></notfind >
```

4.3.8 "equal" operation

This operation returns true if the set of characters "str" defined in value is equal to the current line being parsed or to the variable "varname" of type character.

```
1 <-- true if value “str” is equal to the current parsed line->
2 <equal value="str"><curLine/></equal >
3
4 <-- true if value “str” is to the variable varname->
5 <equal value=" str "><variable name="varname"/></equal >
```

4.3.9 Utilities operation

Utilities operations correspond to operations that can help to manipulate data stored in variables.

4.3.9.1 "generateId" operation

the "generateId" element generates an id comprised between a start and an end value stored in variables of type UI and store the result in a variable (vector of UI).

```
1 <generateId>
2     <start><variable name="startvalue"/></start>
3     <end><variable name="endvalue"/></end>
4     <variableAssign name="block_id"/>
5 </generateId>
```

4.3.10 PIPER Component operation

This section is related to the operations that can be performed on a PIPER component. Each PIPER component created during the parsing of files can be identified by its Id (or name) and the keyword defined in `keyword` defined in the "rule" element element used to create it. For each PIPER component, methods are available to create it and assign to it parsed data stored in variables.

- Model File assignement
- Piper Node operation
- Piper Element operation
- Piper Group operation
- PIPER Frame operation
- PIPER Joint operation (obsolete: not needed)
- PIPER Parameter operation

4.3.10.1 Model File assignement

The "objectModelFile" element defines a FE model file to be parsed. It includes only one methode element: "setFile".

```
1 <objectModelFile>
2     <setFile><variable name="varname"/></setFile>
3 </objectModelFile>
```

4.3.10.2 Piper Node operation

The "objectNode" element adds a node in the Piper model. It includes two methods:

- "setId": creates a new node or activates one.
- "setCoord": sets coord for the active node. The variable use to assign coord must be a vector of three double.

```
1 <objectNode>
2     <setId><variable name="varnameforId"/></setId>
3     <setCoord><variable name="varnameforcoord"/></setCoord>
4 </objectNode>
```

Example of "objectNode" element usage: line that defined node in Pam-Crash format (blanks are replaced by dot for visiblity) NODE../..7020551.......-0.869206......-92.601196.........9.03719

```
<parseRule>
    <curLine/>
    <parse format="UIpos" length="8" offset="8" repeat="1" separator="none"><variableAssign name="NodeId"/>
    </parse>
    <parse format="F" length="16" repeat="3" offset="16" separator="none"><variableAssign name="coord"/></
    parse>
</parseRule>
<objectNode>
    <setId><variable name="NodeId"/></setId>
    <setCoord><variable name="coord"/></setCoord>
</objectNode>
```

4.3.10.3 Piper Element operation

"objectElement1D", "objectElement2D" and "objectElement3D" elements respectively add one-dimensional, two-dimensional and three-dimensional elements in the Piper model. It includes two methods:

- "setId": creates a new Piper element or activates one.
- "setElemDef": sets the element definition for the active element. The variable use to assign coord must be a vector owhich dimension depends of element type.

One-dimensional element:

```
<objectElement1D>
    <setId><variable name="varnameforId"/></setId>
    <setElemDef><variable name="varnameforelementdef"/></setElemDef>
</objectElement1D>
```

Two-dimensional element:

```
<objectElement2D>
    <setId><variable name="varnameforId"/></setId>
    <setElemDef><variable name="varnameforelementdef"/></setElemDef>
</objectElement2D>
```

Three-dimensional element:

```
<objectElement3D>
    <setId><variable name="varnameforId"/></setId>
    <setElemDef><variable name="varnameforelementdef"/></setElemDef>
</objectElement3D>
```

4.3.10.4 Piper Group operation

This element defines the group in PIPER model and adds content to it. `type`**:** defines type of group content (Node, Element3D, Element2D, Element1D, Group)

- "setId": creates a new Piper group or activates an already created one.
- "setName": sets a name group.
- "setPart": true/false, true to indicate that this group is a part of the finite element model
- "addInGroup": sets group content in the active group.

Example of group of three-dimensional elements:

```
<objectGroup type="Element3D">
    <setId><variable name="varnameforId"/></setId>
    <addInGroup><variable name="varnameforelementId"/></addInGroup>
</objectGroup>
```

In this example, Id stored in variable "varnameforelementId" corresponds to Id associated with one of the keywords associated in "meshComponent" element with Piper component declared in `type` of the "objectGroup" element (Element3D in the example). If Id added is associated with another keyword(s), these keywords can be specified in the "sourceId" element as in the following example. In the example, Id of groups are added in a Group of group PIPER component and Id of group(s) to be added are associated with keywords defined in the "sourceId" element.

Example of group of group using sourceId:

```
<objectGroup type="Group">
    <setId><variable name="varnameforgroupId"/></setId>
    <addInGroup>
        <sourceId>kw1 kw2 kw3</sourceId>
        <variable name="GGroupId"/>
    </addInGroup>
</objectGroup>
```

4.3.10.5 PIPER Frame operation

The "objectFrame" element adds a frame definition in the PIPER model. Methods of the "objectFrame" element are:

- "setId": creates a new Frame or activates an already created one.
- "setName": sets a name frame.
- "setOrigin": sets the origin (node Id).
- "setFirstDirection": sets the first direction (X, Y or Z).
- "setFirstAxis": sets the node id that defined the first axis.
- "setSecondDirection": sets the second direction (X, Y or Z).
- "setPlane": sets the node id to define the plane local x-y

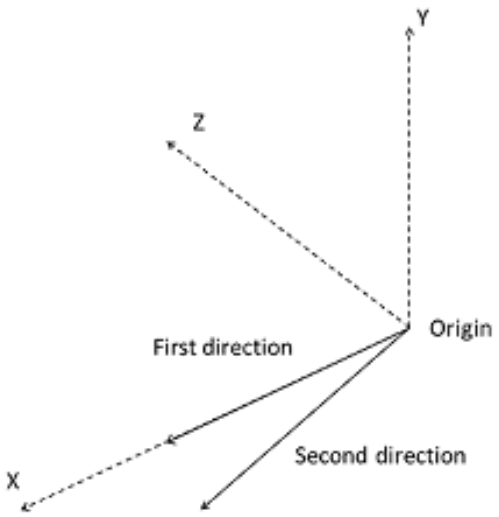


Figure 3: Frame Definition

Example of "objectFrame" element usage:

```
<objectFrame>
    <setId><variable name="FrameId"/></setId>
    <setName><variable name="FrameName"/></setName>
    <setOrigin type="coord"><variable name="Org"/></setOrigin>
    <setFirstDirection>X</setFirstDirection>
    <setFirstAxis type="coord"><variable name="First"/></setFirstAxis>
    <setSecondDirection>Y</setSecondDirection>
    <setPlane type="coord"><variable name="Second"/></setPlane>
</objectFrame>
```

4.3.10.6 PIPER Joint operation (obsolete: not needed)

The "objectJoint" element adds a joint definition in the PIPER model. Revolute and Spherical joints are the allowed type of joint. Methods of the "objectJoint" element are:

- "setId": creates a new joint or activates an already created one.
- "setType": sets the type of the joint (REVOLUTE or SPHERICAL).
- "setCenter": sets the center of the joint (nodeId).
- "setAxis": sets a second node Id to define the axis of the REVOLUTE joint.
- "setDof": sets the Degrees of Freedom of the joint. six boolean values (TX,TY,TZ, RX, RY,RZ with O: blocked; 1: free). Default for SPHERICAL joint (0 0 0 1 1 1). Default for REVOLUTE joint (0 0 0 1 0 0).

Example of "objectJoint" element usage (REVOLUTE joint):

```
1 <objectJoint>
2     <setId><variable name="JointId"/></setId>
3     <setType>REVOLUTE</setType>
4     <setCenter><variable name="JointAxis1"/></setCenter>
5     <setAxis><variable name="JointAxis2"/></setAxis>
6     <setDof>0 0 0 0 1 0</setDof>
7 </objectJoint>
```

4.3.10.7 PIPER Parameter operation

The "objectParameter" element adds a parameter definition in the Piper model. A parameter allows to associate a parameter name with a value parsed in the file. Methods of the "objectParameter" element are:

- "setId": creates a new parameter or activates an already created one.
- "setValue": sets the value for a parameter with two members elements:
- "type": sets the parameter name.
- "value": sets the value of the parameter.

Example of "objectParameter" element usage:

```
1 <objectParameter>
2     <setId><variable name="PropId"/></setId>
3     <setValue>
4         <type>ParameterName</type>
5         <value><variable name="ValueVar"/></value>
6     </setValue>
7 </objectParameter>
```

5 PIPER Graphical User Interface

This page provides basic documentation on the Graphical User Interface (GUI) of the PIPER application.

5.1 PIPER main window

The main window is composed of:

- A **menu bar** which gives access to
  - Project menu : to create from FE files, save and load a Piper model and update the FE model.
  - Target menu : manage project targets
  - Model History menu
- Module Selection to load a module
- The **module area** displays the current module user interface which are built using the same layout:

  - The center area of the main window is dedicated to the main module output, often a 3D Display of the model with feedback of the modifications applied by the module
  - The module functionalities are accessed through GUIs organized in *toolbox* windows
  - The right side of the main window contains:
    * specific buttons to show and hide these *toolbox*, this can also be done using keyboard shortcuts, `Ctrl + Shift + specific letter`
    * buttons to select the current mouse tool

- The **Log panel** outputs status messages from the application and the modules (Log panel button: show/hide the log panel).

The *toolbox* windows let the user organize his display, even on multi monitors setups, according to its current task. Moreover the size and position of the *toolbox* windows are persistent through executions of the application.

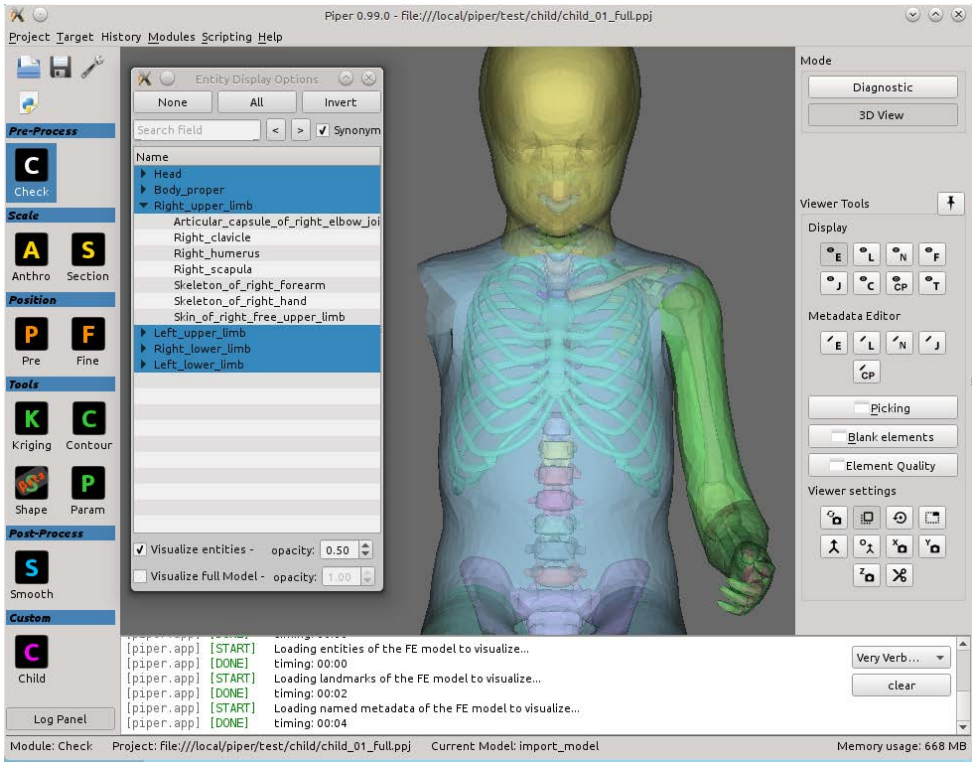


Figure 4: Piper main window

5.1.1 Project menu

- **Project** → **New** : Clear the current project and start with empty model, metadata, environment and target.
- **Project** → **Open** : Open an existing Piper project `.ppj` file.
- **Project** → **Save** : Save the current project, model, metadata, targets and links to the environment definition files.
- **Project** → **Import** → **Import HBM from vendor files** : Creates a new Piper model from a human body model (From finite element model or from graphic format, only OBJ format is supported) . It tequires to select:
  - xml file with model description (∗.pmr) (Preparing the model and metadata)
- **Project** → **Import** → **Import Piper metadata** : Import Piper metadata from a `.pmr` file metadata with selection of metadata types to import

- **Project** → **Export** → **Export HBM to vendor files** : Exports the model (transformed by the Piper application) in a new directory. It Generates a copy of the original Finite Element Model files in the selected output directory with updated node coordinates and updated parameter values.

- **Project** → **Export** → **Export Piper metadata** : Export Piper metadata to a `.pmr` file with selection of metadata types to export. All current metadata of selected types are deleted and replaced by imported ones. Export landmarks name and coordinates in a simple ascii file

- **Project** → **Export** → **Export nodes coordinates** : Export entities nodes FE code id and coordinates in simple ascii files, one file per entity

- **Manage environment models:** Add or remove environment models defined in vendor files.

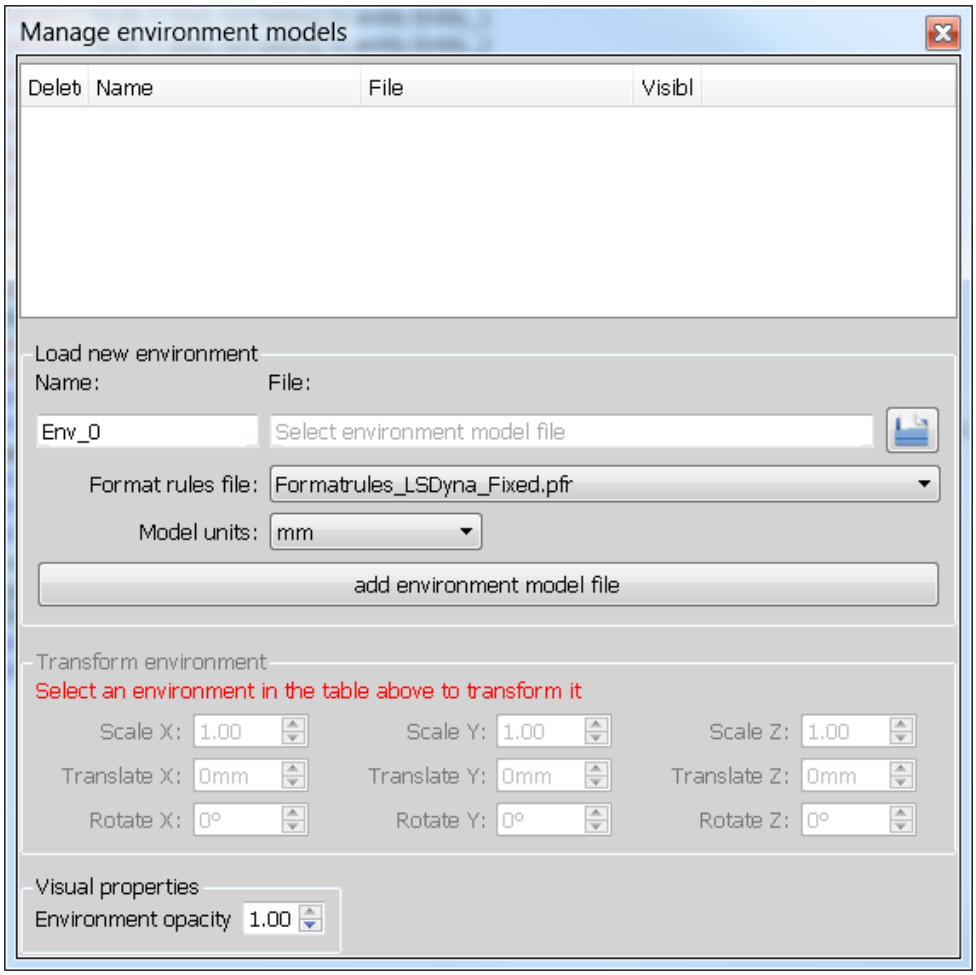


Figure 5: Manage environment models

- **Quit:** Quit the application

5.1.2 Target menu

- **Target** → **Clear targets:** Remove all current targets

- **Target** → **Import targets:** Replace the current target list with the list contained in a `.ptt` file.

- **Target** → **Export targets:** Export the current target list to a `.ptt` file.

5.1.3 Model History menu

All actions that modify the loaded model create a "history node" in the history menu. You can use this menu to undo those action. The history is linear, i.e. it is not possible to do multiple "branches": if you undo an action, you can redo it, but if you undo, then do some other action, the previously undone action will be lost forever. The current state is highlighted in the menu by **bold** font. The "state" is effectively the coordinates of the nodes of the model.

5.1.4 Module Selection

Available modules integrated in the PIPER application can be selected in the module selection panel, see PIPER Modules Overview page for documentation on each module. The application starts with the Check Module loaded.

5.2 Module parameters

Some modules have parameters which can be set in the *Module Parameters* dialog accessible in the Project menu and a tool icon. The description of these parameters can be display with the *Help* button. The modules parameters are saved in the project `.ppj` file.

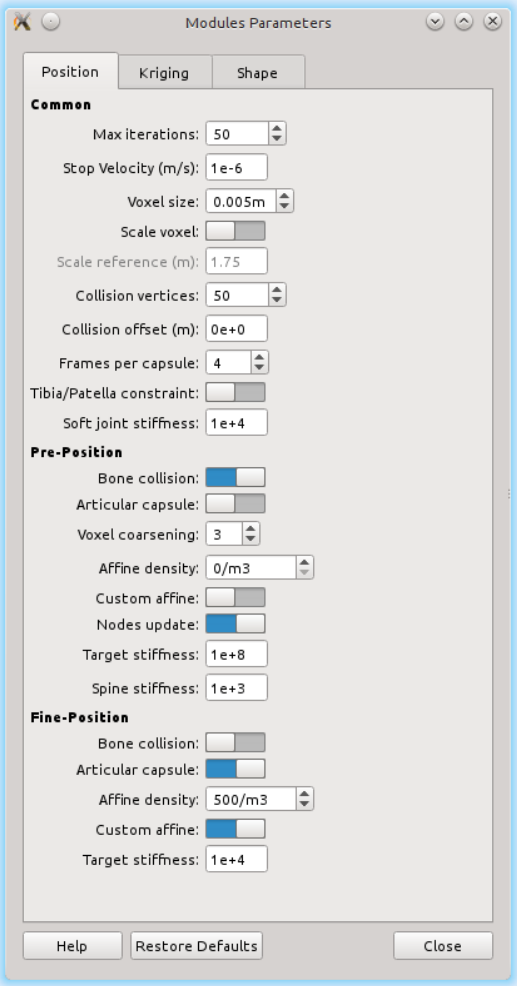


Figure 6: Set modules parameters

5.3 3D Display

The interactive navigation is performed using the mouse :

- **rotate** (yaw+pitch) : right click "press&hold"
- **rotate** (roll) : right click "press&hold" while "holding" the "Ctrl" key
- **pan** : middle click "press&hold"
- **zoom** : the scroll wheel

The left click is kept for interactive manipulation in the 3D view.

The application integrates two distinct viewer: The Physics simulation viewer available in physics based modules and the Generic viewer in other modules.

5.3.1 Generic viewer

The generic 3D viewer is a tool available in all modules where visual exploration of the data is relevant or helpful. Figure Generic 3D viewer: Tools shows an interface of a module using the generic viewer. The marked GUI controls operate tools available in the viewer: Display Settings - under number 1 in the Figure - provide a few additional options to control behaviour of camera and some rendering options; Display Options - under number 2 - are used to choose which parts of the loaded project should be displayed and which should be hidden; Picking - under number 3 - provide useful tools for selecting any part of the model seen on the screen using mouse; and Blanking - also under 3 - is a simple, yet effective tool to further ease exploration of the model by removing selected elements.
Each module that uses the generic 3D viewer is using the same "instance" of it. This means that whatever settings you change, data you load, the way you move the camera etc. will remain the same when you switch to another module. An exception for that are data that are module specific - for example, kriging control points you load in the Kriging Module will be removed from the viewer once you leave the module.
The display is capable of displaying all the geometry and metadata in your project. However, some features, such as e.g. highlighting of selected points, displaying normals, landmarks and others, require a graphic card with OpenGL 3.2 capabilities. OpenGL 3.2 is a standard from the year 2009, so you should not run into problems if your computer is not older than that. If it is, you will be warned in the application log. Some of the features will switch to using a "legacy code", so you should still be able to use them, but the performance will be significantly slower and might have some bugs, as the "legacy code" is not tested very thoroughly.
If you encounter some problems or glitches related to the 3D viewer, please first make sure that your graphic card drivers are up to date before reporting the issue.

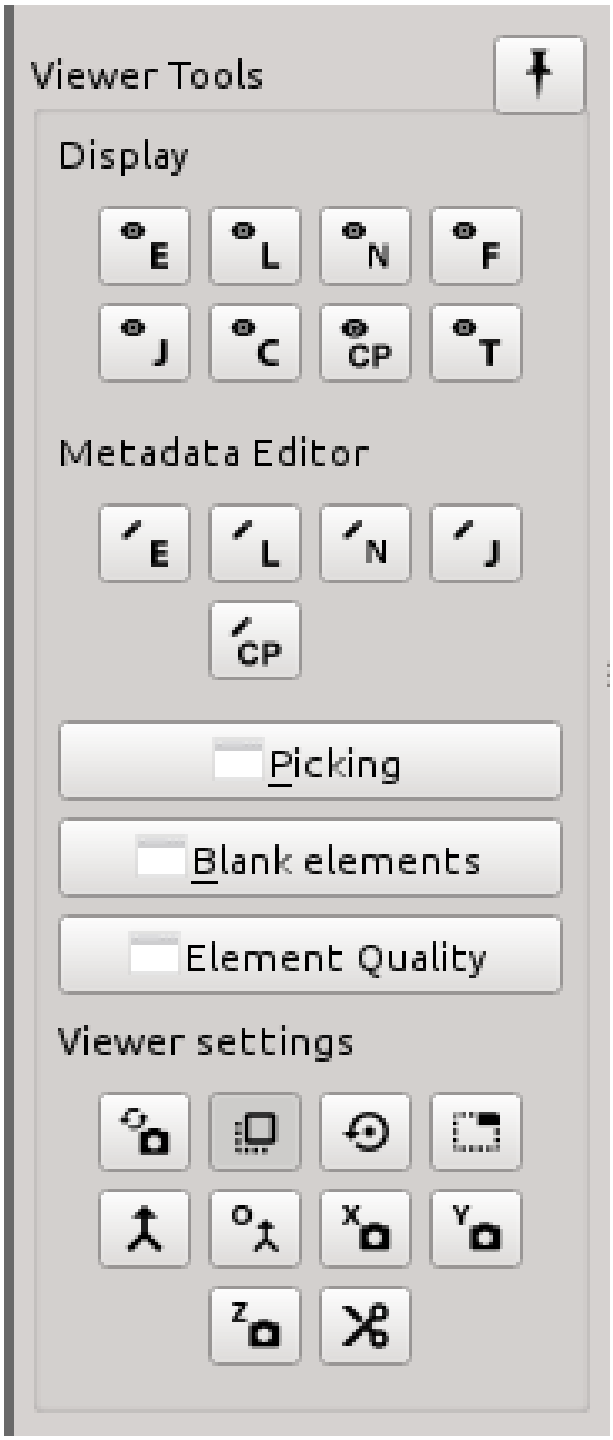


Figure 7: Generic 3D viewer: Tools

5.3.1.1 Display Settings

The group of "display settings" buttons currently consists of five buttons in two rows. The first row serves to manipulate the camera by which the scene (your model) is observed, the second row contains tools for displaying geometrical properties of the model.
Camera settings (functionality of each button, from left to right):

- Reset Camera: clicking this button will reset the focal point of the camera (the point the camera is "looking

at") to (0,0,0), resets zoom factor to 1 (i.e. no zoom) and sets the camera position to such position that all visible objects in the scene are within sight. However, it does not change the direction of the camera. It is useful mainly if you "get lost" while exploring the data.

- Toggle Projection Type: this button controls the type of geometrical projection the camera uses - perspective or parallel (orthogonal). When it is on (by default), parallel projection is used. Be aware that internally, many variables controlling the camera, e.g. zoom factor, are dependent on the type of projection, therefore, switching the projection types will likely make it appear that the moved or changed position.
- Pick Focal Point: clicking this button will enable picking of custom focal point of the camera, which is, among other thigns, the center of rotation for the camera. Click anywhere on the model to set the point as the focal point. Note that until you disable the button, the focal point picking is still active, so you can try many different points until you find one you are satisfied with. Picking the focal point will disable any other Picking you might have active and vice versa, choosing some picking tool will disable the focal point picking. This tool is useful when you need to inspect part of the model from up close.

Geometrical properties (functionality of each button, from left to right):

- See element edges: toggling this button (default off) allows you to see outlines of edges of each element.
- See element normals: toggling this button (default off) allows you to see normalized normals of each element (this can take a few seconds for large models).

5.3.1.2 Display Options

In this group of buttons, there is one button for each type of metadata (left to right, top to bottom): entities, landmarks, generic metadata, frames, joints, contours and targets. Upon clicking it, you will open a window similar to the one in Figure Display options (that one is for the "entity" metadata type). By clicking on each object in the list, you will hide or show it - by default, everything is shown. There are also buttons for showing all, hiding all and inverting the selection.
The last button in the group contains visual settings, allowing you to change the colours, opacity and other visual properties of the metadata.
The "entity" window also contains two additional checkboxes: "visualize entities" and "visualize full model". This allows you to choose the mode in which the model is displayed - either as a set of multiple objects, one for each entity as they are defined by your metadata; or as a single mesh consisting of all the elements in the model. Since they obviously exist in the same place, you will usually want to have displayed only one of them, although you have the option to see both, which might be useful in conjuction with Blanking or the opacity settings (right next to the entities).

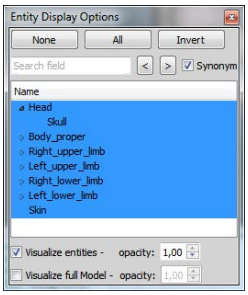


Figure 8: Display options

5.3.1.3 Picking

Upon clicking the "Picking" button in the right-hand panel, a pop-up window with picking tools will open - see the Figure below. In Piper, "picking" is the word we use for "selecting using the display". Whenever you need to select something, you will find all the available tools in this window. All selection is persistent across all modules that use the generic 3D viewer - what you select in one module, you can process in another one. Selected objects will be highlighted by bright green colour.

Hint: Note that in some cases, especially in the case of picking elements, the objects need to be coloured in another way - for example based on the element quality. This will disable the selection highlighting, but will not de-select the elements! They will still be selected - to turn the selection highlighting back on, simply click on some element picker (see below) and the "highlighting by selection mode" will be turned on again (while, naturally, "highlighting by quality" will be turned off).
The first part of the window is a grid of buttons called "Pickers". Each of the buttons initializes some type of picking on a specified "target". Currently, there are three different types and five different targets, though not all combinations are available. Before explaining each of them, a few general notes:

- Picking is done by single clicking the left mouse button. This currently cannot be re-mapped.
- You can change the behaviour of picking by using SHIFT and CTRL keys. The behaviour is slightly different based on the type of the picker, but the general idea is that by holding SHIFT, you are appending to the current selection, by holding CTRL you are de-selecting from the current selection and if you not hold anything, the previous selection is discarded before doing a new one.
- Check the "See hints" checkbox on the bottom of the Picking window to see detailed description for each picker, including the behaviour of SHIFT and CTRL keys.

Picking types:

- Single pick: picking is done by single left mouse button click on the target. Always selects only one target, useful when you are selecting only a handful of specific objects.
- Rubber band: picking is done by holding the left mouse button and dragging. This will draw a rectangle and interactively (i.e. as you are dragging) select all that is on the surface beneath the rectangle.
- Box: picking is done by holding left mouse button and dragging. As with the rubber band, this will draw a rectangle on the screen, however, picking is done only after the mouse is released. Upon the release, a box will be constructed by projecting the rectangle orthogonally "from the screen into the scene" - the front and back side of the box will be computed so that everything in the scene within the drawn rectangle is inside the box. Because the box is constructed based on orthogonal projection of the rectangle, the camera is automatically switched to orthogonal projection mode if it was in perspective. You can switch it to perspective manually after activating the picker, however, the shape of the resulting box you will get when using perspective projection will be rather counter-intuitive, so we recommend using orthogonal projection.

Picking targets:

- Entity: will pick entire entities. Note that while this will colour the entity all in the birght green colour, it does not mean that the elements are selected. If you attempt to do some selected elements-based operation, it will not do anything as all the elements will still be internally marked as not selected.
- Node: will pick nodes of the mesh. Picked nodes will be highlighted by small green spheres around them. When using the single picker for nodes, the nodes are selected based on vicinity to the point you click, i.e. you do not need to click "exactly" on the node (as that is theoretically not possible, since a node has no size), but rather a node that is closest to the point you click will be selected.
- Element: will select elements. *Known performance issue: in the current version of Piper, highlighting selected elements on transparent models uses up a lot of computational resources, resulting in slow performance of camera interaction. We are currently looking for a solution. In the meantime, it is recommended to turn off transparency when doing complicated element selection tasks (you can do that in the Display Options - entity window).*
- Landmark: will select landmark. You have to click directly on the sphere that represents the landmark. For landmarks that consist of multiple points, all points of the selected landmark will be highlighted by teal colour, while the exact selected point will be highlighted by the standard bright green.
- Frame: will select a whole frame. Simply click on any of the arrows representing the frame.

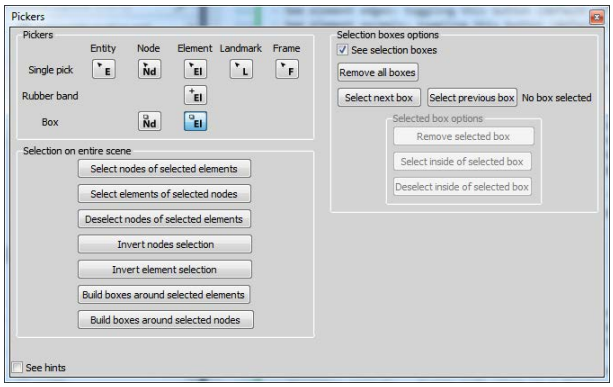


Figure 9: Picking

Second part of the Picking window - "Selection on entire scene" - contains several buttons for performing global selection tasks, most of them rather self-explanatory. The tasks are performed on each object that is in the scene and is visible. The last two buttons in the that group - "Build boxes..." - server to create a selection box around selected elements and nodes. Note that clicking these buttons will not perform any actual selection, only creates the boxes - those can than be used for selection using the "Selection boxes options" (see below) or other tasks, e.g. in the Transformation smoothing. A simple algorithm for point clustering is used for building the boxes - the points/elements are clustered and box is built around each cluster. While the algorithm will work well for some basic cases of well defined clusters, in many cases a manually constructed box will have a better fit.
The last part is "Selection boxes option". You have to select some box picker to make this part of the window enabled. It groups together several control elements for using selection boxes (from top to bottom):

- See selection boxes: this checkbox allows you to toggle the visibility of all selection boxes. When visible, boxes are rendered as semi-transparent green boxes.
- Remove all boxes: this will remove all the boxes from the viewer, but note that it will not do any de-selection of the content within the boxes. This cannot be undone!
- Select box: using these two buttons you can cycle through and select the created boxes. The selected box will be highlighted by a teal colour (it is recommended to check the "See selection boxes" if you are about to use this feature). Upon selecting some box, the buttons below will become enabled allowing some basic operations on the box:
    1. Remove selected box: removes the box from the scene. Again, note that this will not do any de-selection and that it cannot be undone!
    2. Select/deselect inside of the box: selects or deselects all primitives inside the box. The type of primitive that will be processed (element or node) depends on what picking target is chosen in the "Pickers" section of the Picking window - this, among other things, allows you to use the same box to select both the nodes and the elements using the same box.

5.3.1.4 Blanking

Element blanking is a simple tool that allows you to explore the inside of your FE model. Normally, you can only see the outer shell of your model and even with transparency, you cannot inspect the shape of internal elements. With blanking you can "cut off" part of the model to look inside. Upon clicking the "Blank elements" button, a simple window with three buttons will open: blank selected elements, undo last and undo all.
The first button will remove all elements that are currently selects among all visible objects. See Picking for description of the picking tools you can use to make the selection (box selection is probably the most useful for the purpose of blanking). Note that the blanking is done only on the visual representation of the model - it does not actually change your FE model, it is not an editing tool, simply an exploration tool.
The other two buttons serve for undoing. Undo last will undo the last action, while undo all will return to the original state. Although there is no re-do action, the elements selected before the blanking remain selected after the undo, so you can toggle back and forth between two blanked states (but going back more than one step will lose the latest

selection entirely).
The following figure shows an exemplar workflow for model exploration using blanking - selects a part of the mesh by box, blanks the selection and then uses the VTK toolkit Quality metrics to inspect quality of elements inside the model:

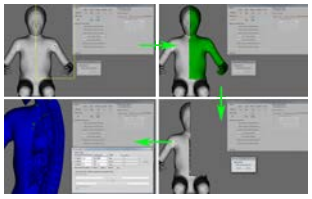


Figure 10: Selecting by box

5.3.1.5 Metadata Editing Tools

In this group of buttons, there is one button for each type of metadata for editing the respective type of metadata (left to right, top to bottom): Entity Editor, Landmark Editor, Named Metadata Editor, Joint Editor, Control Point Editor.

Entity Editor

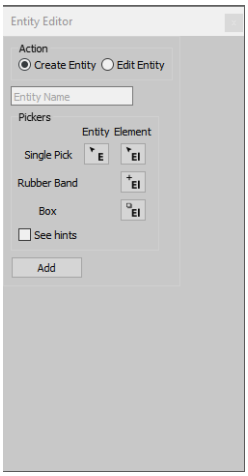


Figure 11: Entity Editor

Landmark Editor

Figure 12: Landmark Editor

Named Metadata Editor

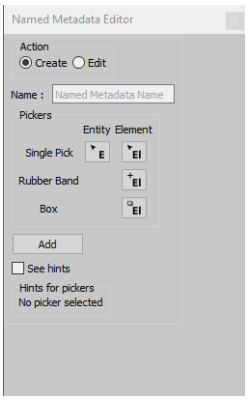


Figure 13: Named Metadata Editor

Joint Editor

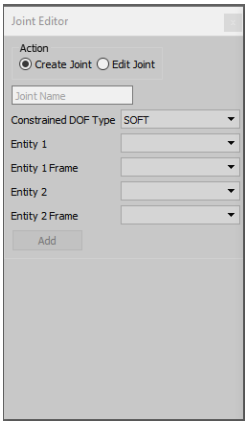


Figure 14: Joint Editor

Control Point Editor

Figure 15: Control Point Editor

5.3.2 Physics simulation viewer

This viewer is utilized in the Pre-Positioning Module and Fine-Positioning Module modules. The model displayed in this viewer is derived from the loaded FE model specifically for the purpose of physics based positioning. This means that the tools available in the Generic viewer are not available here, because the displayed model is technically not the original one.
Additional navigation tools in the Physics simulation viewer are:

- **center view** : double left click on a displayed object (available only in Physics simulation viewer)
- **align view on closest axis** : double right click any where on the view (available only in Physics simulation viewer)

5.3.2.1 MultiView

MultiView display is available by drag+dropping the very Top-Right corner of the 3D context towards the left hand side or towards the bottom. Alternatively, the bottom-left corner offers the same functionality in opposite directions.

In order to remove the Multiple views created, one has to operate backward, namely drag+dropping the very Top-Right corner of the multiple (sub) 3D context towards another sub 3D context located on its right side or on its upper side. Alternatively, the bottom left corner offers the same functionality in opposite directions.

The creation order should be reversely followed, for instance in the following image, one of the two vertically stacked 3D display context should be merged (removed) with the other one before trying to remove the remaining left or right 3D display context.

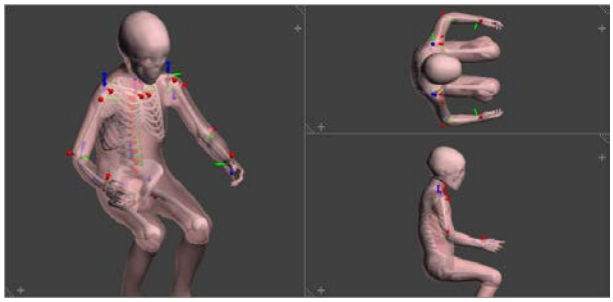


Figure 16: Physics simulation viewer: Multiview

6 PIPER Modules Overview

6.1 List of modules currently integrated

The following modules are available in the application:

- **Pre-Process**
  - Check Module : checks the HBM when it is imported.
- **Scale**
  - Anthropo Module : estimates anthropometric dimensions based on a set of predictors (scaling)
  - Scaling Constraint Module : allows building a correspondence between anthropometric dimensions and a HBM to prepare scaling, and calls the Kriging module to transform the HBM (scaling)
- **Position**
  - Pre-Positioning Module : pre-position the PIPER model using an interactive physics-based simulation (positioning)
  - Fine-Positioning Module : position the HBM using physics-based simulation (refinement of pre-position) (positioning)
- **Post-Process**
  - Smoothing module : tool to improve the quality of the HBM by smoothing the mesh or the transformation (post-process)
- **Tools**

  - Kriging Module : transforms the HBM using Kriging interpolation and sets of control points (tool)
  - Contour Deformation Module : tool to transform the HBM using contour based approaches (tool, scaling, positioning)
  - Shaping Module : tool to shape the flesh of you HBM using a set of point handles (beta version)
  - Scaling Parameter Module : tool to modify model parameters (tool, post-process)

- **Custom**
  - Scaling the PIPER child model by age : a small module/function dedicated to the age scaling of the PIPER child model (scaling)

Several modules are typically used (chained) to perform a transformation. An overview of the articulation of the modules and functionalities for positioning and scaling is provided below. A few examples of workflows are also provided in the Workflow Examples.

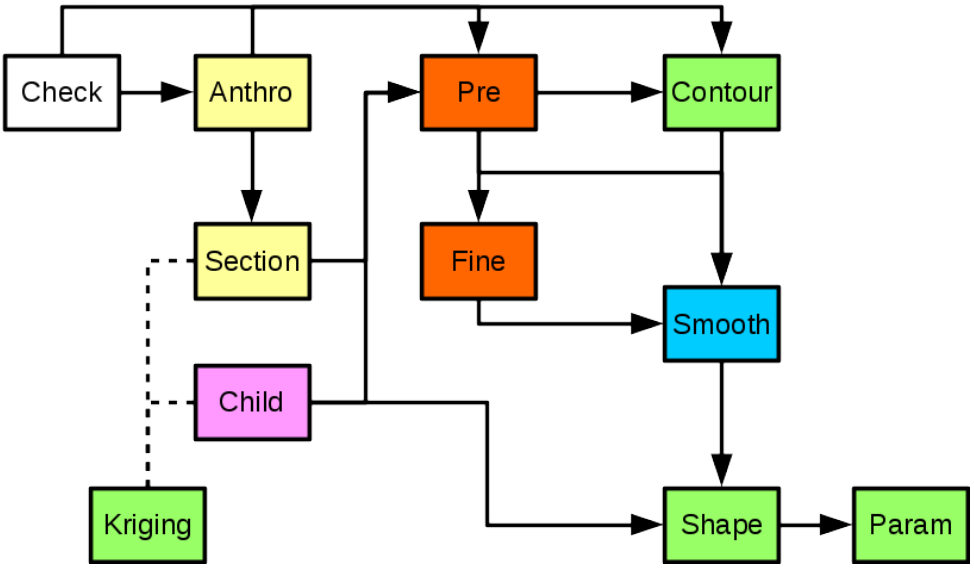


Figure 17: Overview of the articulation of the modules

6.2 Scaling in PIPER: overview

HBM Scaling approaches seem more widespread in the literature than positioning approaches. The PIPER framework provides several tools than greatly can help with HBM scaling mainly based on anthropometric dimensions or landmarks (unfortunately, no module using Statistical Shape Models could be integrated yet) The main modules provided include:

- a module to estimates anthropometric dimensions based on a set of predictors (Anthropo Module) and three public anthropometric databases from children to elderly. A functionality to predict anthropometric dimensions directly using the GEBOD regression is also included.
- the Scaling Constraint Module to build interactively correspondences between anthropometric dimensions and a HBM to prepare scaling. The module can also call all required modules to define the target and perform the transformation
- a geometrical interpolation module to support model morphing (Kriging Module). The module integrates many numerical features useful within the context of HBM scaling (allows arbitrary number of control points, automatic control point decimation, weighting of the bone and skin, use of surface distance...)
- a module (Scaling the PIPER child model by age) dedicated to the PIPER Child scaling with age, which allows generating easily models by selecting and age or stature started (based on the GEBOD regressions). The functionality to scale the material parameters with age is also included as an experimental feature.
- a Contour Deformation Module to transform the HBM using contour based approaches

6.3 Positioning in PIPER: overview

Positioning finite element Human Body Models (HBM) for safety is a challenging task. To the contrary of dummies, applying a succession of rotations and translations is not possible because the HBM includes non robotic joints with complex articular contacts because soft tissues need to be deformed along. Because of this, HBM positioning is typically performed by finite element simulation. These can be time consuming to set up and to run. Beyond that, the properties of the HBM were typically selected for their response at high acceleration levels and may not be appropriate for physiological motion or low deceleration. Furthermore, postural preferences and muscular reactions also affect the posture .

The PIPER framework aims to propose some alternative methodologies that can be used along current approaches. Positioning in PIPER typically starts with the Pre-Positioning Module (or pre-positioning). The HBM is automatically transformed into a simplified model with a limited number of degrees of freedom that can be used in physics-based interactive simulation. Despite being simplified and interactive, the simulation can among others account for collisions between bones (to prevent penetration, limit range of motion, ...) and provide a rough transformation of the soft tissues.

The pre-positioning process is the place where the user can input its various constraints, weight them, and compute a plausible posture (for the skeleton in particular). Constraints could also include *a priori* knowledge such as physiological observations or postural preferences which are not classical mechanical parameters. For now, physiological descriptions of the spinal curvature (called Spine controller) can interact with the model (e.g. collision detection on the vertebrae) during postural change.

Several options are then possible to transform the HBM using this pre-position as the target:

- the Fine-Positioning Module : the pre-positioning motion can be repeated (using the constraints or the bone positions) with finer parameters for the simulation. While more time consuming (for the initialization in particular), it can provide a more plausible deformation of the flesh.
- the Contour Deformation Module can be applied using the bony landmarks from the preposition as a target. It can also be used independently
- the pre-position can be used to generate a finite element simulation input (though a python script, an example being provided) and a full finite element simulation can be run.

In all cases, the use of the Transformation smoothing after positioning was found to greatly improve the results. In some cases (for smaller motion), the pre-position may be directly used and lead to a plausible and runnable model after smoothing.

In all cases, smoothing functionalities can be used to improve the HBM quality (Smoothing module).

6.4 Check Module

6.4.1 Overview

The application starts with this module which checks several elements of the model and outputs a Diagnostic, it also offer to explore the model in a 3D view.

- **Parameters:** None
- **Metadata:**
  - Required: None
  - Optional: Any
- **Target** Input: Any
- **Target** Output: None

Author

Thomas Lemaire - INRIA

6.4.2 Diagnostic

In this mode various information on the model and its metadata are displayed.

Figure 18: Diagnostic in the check module

- **search:** highlight a word in the diagnostic text
- **Anatomical frame:** check for anatomical frames availability (these frames are defined in Appendix: List of frames)
- **Verbose:** print even more information

6.4.3 Hints

If you see warning regarding landmarks, joints, etc, it may be that the name of the entity is not present in the anatomy database (misspelling of other).

6.4.4 3D View

In this mode the user can explore the model and its metadata using the Generic viewer.

6.5 Anthropo Module

6.5.1 Introduction

For Human Body Model (HBM) scaling, example of scenarios include users that would like to generate the most likely subject matching a few characteristics:

- a given height and weight (e.g. average female)
- one of the height, seating height and shoulder width specified in the EC Regulation R129 for child restraint systems
- the few measurements available on a PMHS in a published validation study
- ...

Alone, such characteristics may be insufficient to drive the scaling of a HBM. However, they could be completed by using statistical relationships (*a priori* knowledge) linking these known characteristics (called predictors) to outputs such as anthropometric dimensions or shapes. The anthropometric dimensions can then be used in other modules such as the Scaling Constraint Module to scale a model. The current module focuses on the prediction of anthropometric dimensions based on a limited but arbitrary set of predictors.

6.5.2 Description / methodology

The Anthropometric Prediction module can be used to generate anthropometric targets (in the sense of the PIPER Framework) based on a set of predictors and anthropometric datasets. The underlying statistical process is based on Parkinson and Reed 2010 (see References). It is fully automated.

A regression between selected predictors, such as weight, is first evaluated for a chosen population. A set of targets is then sampled from this regression measurements that can be used in other modules to personalize a model.

Targets can be sampled from the regression by including the regression error or not (sampledOutput or meanOnly). Similarly, the values of the predictors can either be forced to a given value or sampled from a statistical distribution (Fixed Predictor/Sampled Predictor). That statistical distribution is estimated from a population that can be the same as the one used to generate the regression or from a different one.

Populations are constructed by upsampling through the definition of some bins or subsets of population they consist of. Each bin is defined by a set of criteria. For example, a population could contain a single bin consisting of female subjects between the ages of 20 and 35, or it could consist of two bins of female subjects respectively between the ages of 18 and 35, and the ages of 36 and 65. Please note that you have to define the same amount of criterias in each bins.

The module can deal with multiple datasets. Three publicly released datasets are provided with the release and named as follows:

- ANSUR: public dataset with anthropometric measurements on military personnel (almost 4000 subjects, ages: 17-51. See References)
- CCTANTHRO: dataset released under a CC-BY 4.0 Licence with anthropometric measurements on Post Mortem Human Subjects (105 subjects, ages: 55-94. See References)
- SNYDER: public data corresponds to anthropometric measurements performed on children (about 3900 subjects, ages: 1-19, not all measurements available for all subjects. See References)

Remarks

Warning: while the process is automated, it is critical to remember the principles of the underlying process. The plausibility of the predicted target is widely dependent on the relationship between the set of predictors and the dataset (how far is the set of predictor from actual data present in the dataset). Depending on the selection of predictors and their value, the predicted target may be based on unrealistic extrapolations. More feedback about this will be added in the future.

6.5.3 Parameters and Options

- **Parameters :**
  Optional: none

- **Inputs :**
  Interactive menus (opening on the right of the window) help the user selecting the inputs. The set of anthropometric targets are then generated and saved in the results directory by clicking on the "Generate Targets" button. The target can be generated when all indicators are in green (all inputs ok)

- **Parameters :**
  Select if you want to generate a New Regression of reuse one that was previously generated and saved.

- **Dataset options:**

  - Select the dataset (see Description / methodology)
  - Select the variables that will be generate (=target or measure of interest) and predictors. For SNYDER: not all measurements are available for all subjects. For ANSUR: all measurements are generated.
  - Select the population characteristics: this is the population you want to base the target generation on. First we need to determine how many bins/subsets of population we want to work with. Once you have defined the number of bins, each bin can be defined with this optional set of criteria: % of population, gender, ethnicity, age
    - **% of population :** Set the percentage of the population described by this bin.
    - **Gender :** Set the gender of this bin of population.
    - **Ethnicity :** Set the ethnicity of this bin of population.
    - **Age :** Set the age range of this bin of population (pre-filled with the maximum range)

- **Target information**

  - **Sample Type:**
    Here you can select if you want the sample type to be meanOnly type, sampledOutput type or both. If you desire mean values of the anthropometric measurements for specific values of the predictors, choose 'meanOnly'. Otherwise, samples will be drawn among the subjects that share the same predictor values and you will select "Normal" for "sample output type", to indicate that the samples use normal distribution.
  - **Sample input type:**
    Here you decide if the values of the predictors you previsously selected in the regression definition, will be sampled or fixed predictor.
    - **Fixed Predictor**
      The choice of Fixed Predicators allow you to determine an exact value for each one of the predictors you selected from the predictor list. If you chose "sampledOutput" as sample type, you will have to determine the number of samples of sets of targets you desire.
    - **Sampled Predictor**
      - **Sample input type** Select the type of sample input, only "normal" available
      - **Number of samples** Define the number of sampled predictors that will be used for the target generation
      - **sample Input reference population Mat file** Define the input reference population mat file that may be used when the statistical distribution is estimated from a population different from the one used in the current regression (upsampled populations are saved in the regression $*$.mat files together with the regression information).
  - **Sample Output type** Select the type of sample output, only "normal" available

- **Output options**

  - Select where and under which name the target files will be saved. The files will be numbered automatically when there are several
  - (optional) Select where the regression will be saved if you would like to re-use it later.

Remarks

The functionalities are implemented into a set of Octave scripts called by a GUI in the PIPER application. Some parameters are not implemented in the GUI and are available in the code.

6.5.4 References

- ANSUR: Gordon CC, Bradtmiller B, Churchill T, Clauser CE, McConille JT, Tebbetts I, Walker RA. 1988 anthropometric survey of U.S. army personnel: methods and summary statistics. Technical Report NATICK TR-89 044. Natick, MA: U.S. Army Natick Research, Development, and Engineering Center. 1989. Data downloaded from http://mreed.umtri.umich.edu/mreed/
- CCTantrho: database released under CC-BY-4.0 by CEESAR in 2017 and distributed by PIPER. Publication pending.
- Parkinson, M. and Reed, M.P. Creating virtual user populations by analysis of anthropometric data. International Journal of Industrial Ergonomics, 2010, 40, 106–111
- SNYDER: Snyder, R, L Schneider, C Owings, H Reynolds, D Golomb, M Sckork. (1977) Anthropometry of infants, children, and youths to age 18 for product safety design. UM-HSRI-77-17, CPSC, Bethesda, MD. Data downloaded from http://mreed.umtri.umich.edu/mreed/

6.6 Scaling Constraint Module

6.6.1 Introduction

To scale a Human Body Model (HBM) a set of anthropometric target dimension values can be defined, using for example Anthropo Module. However, those set of target values are not sufficient to provide constraints for transformation module (like Kriging Module). This module aims to interpret this set of anthropometric target dimension values into geometrical constraints to define the scaling HBM transformation. For example, a anthropometric dimension that defines a length is defined using a segment between two landmarks on HBM. This interpretation is performed with help of a Simplified Scalable model. To construct this simplified scalable model, this current provides tools to interactively describe geometrically anthropometric dimensions on HBM and measure anthropometric dimension value. The simplified scalable model is composed of all defined dimensions, organized in tree-like structure to establish relationships between anthropometric dimensions. Applying appropriate scaling transformation according to target value for each dimension following this tree-like structure, equivalent target dimensions are defined geometrically (the position of the target segment). Thus, on both dimensions (defined on HBM and target dimensions), a set of control points can be defined to fully specify a kriging transformation to scale the HBM according to a set of anthropometric dimension target values.

6.6.2 Description / methodology

Following types of anthropometric dimension can be defined on HBM using a Simplified Scalable model:

- Length
- Multilength
- Circumference
- Width
- Depth

Length and **Multilength:** a Segment Component needs to be defined on HBM.

Circumference, **Width** and **Depth:** a Section Component can be defined on HBM.

A Simplified Scalable model is composed of:

- a list of Component organized in tree-like structure.
- a list of Anthropometric Dimensions to link dimensions to components.

6.6.2.1 Component

To measure dimensions on HBM, geometrical components can be defined interactively on HBM using this current module.

Common properties of component:

- a type: Segment Component or Section Component
- a name
- a parent name (optional): by defining a parent, parent transformation is applied first, before the component scaling transformation.

6.6.2.1.1 Segment Component

A segment component is defined by a list of via-point (minimum 2). Via-point coordinates are defined as the barycenter of a selection of HBM landmarks (at least one landmark per via-point). The length dimension is computed by the Euclidean distance between the last and the first via-points. The Multilength dimension is defined as the cumulative sum of Euclidean distances between successive via-points that composed the segment component. A local frame is associated to each component to define the scaling transformation. Z axis correspond to the normalized vector defined from the first via-point to the last one.

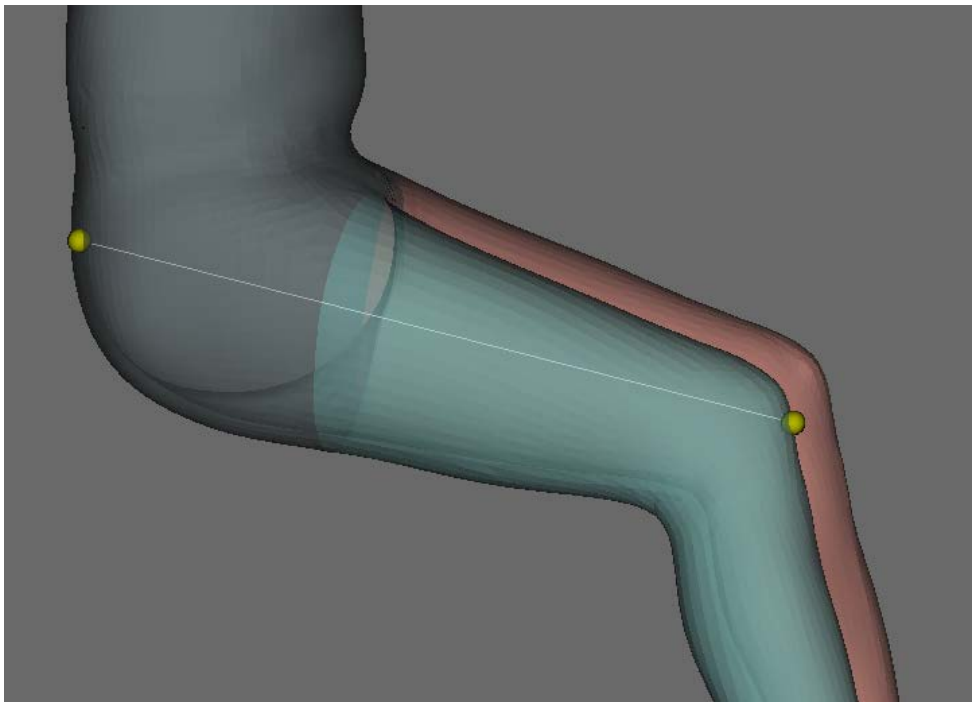


Figure 19: Example of segment to measure the buttock to knee length, defined from landmarks in yellow

6.6.2.1.2 Section Component

The section is defined as the intersection between a plane and en entity 2D envelop (can be a skin entity, a bone...). To define the plane, a relative position along a parent segment has to be defined and Z axis of the parent segment defines the normal of intersection plane. For section, Z axis is defined as the normal of the section plan. X and Y axis are interactivly defined. Circumference dimension is computed as the length of the intersection contour. The

width is computed as the difference between the maximal Y value and the minimal one in the local section frame. For depth dimension, X direction is considered.

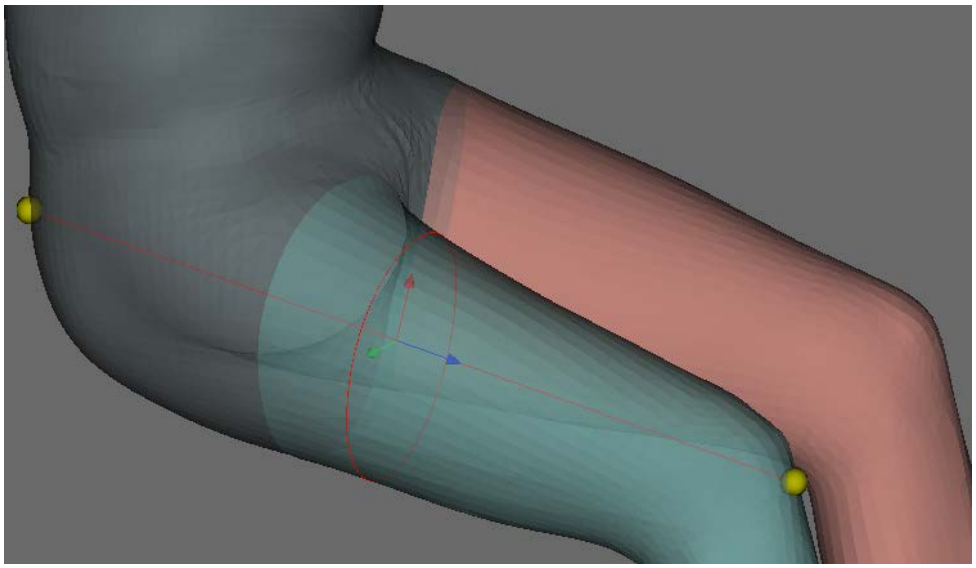


Figure 20: Example of a section define on thigh, cirumference, width or depth dimension can be measured on this section.

6.6.2.2 Anthropometric Dimensions

A name is defined for each dimension. The name of anthropometric dimension must correspond to the anthropometric target name to be applied. To compute the scale factor associated to the target value, a reference component is selected. The scaling factor is computed as the ratio between target value and value measured on HBM on the reference component according to the dimension type. Depending of the dimension type, a scaling transformation is defined. Then, this scaling transformation is applied to a list of selected components in their own local frame, after applying parent transformation.

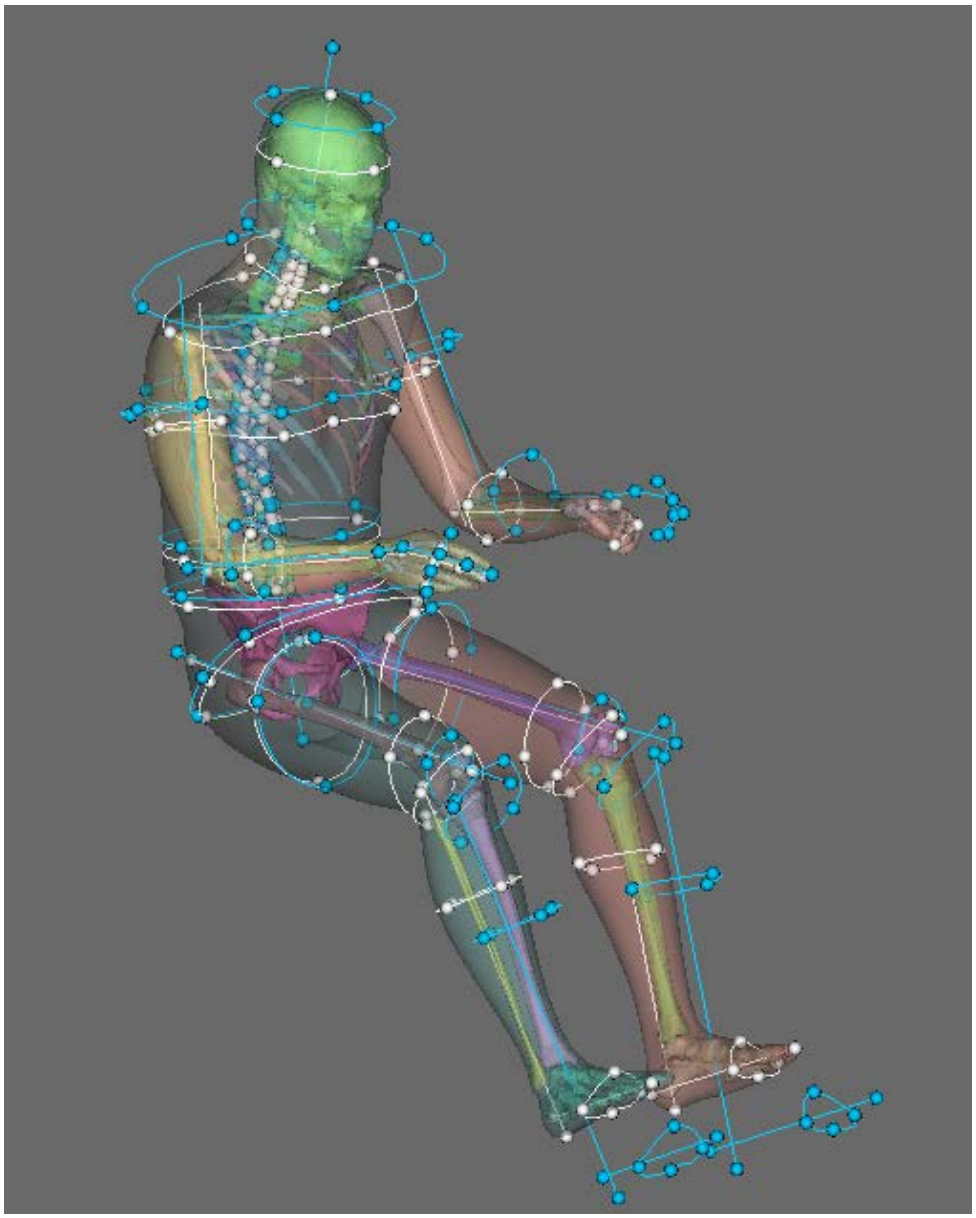


Figure 21: Example of a simplified scalable model defined to represent a selection of ANSUR database anthropometric measurements and the corresponding target.

6.6.3 Parameters and Options

- **Parameters:** None
- **Metadata:**
  - Required: entities, landmarks used used in the construction of the Simplified Scalable Model.
  - Optional: None
- **Target** Input: Target of type anthropometric and landmarks
- **Target** Output: Target of type anthropometric and landmarks
- **Inputs:**
  - The simplified scalable model can be imported from a xml file.

- **Outputs:**
  - The simplified scalable model can be exported in a xml file.

6.7 Pre-Positioning Module

This module implements a physics simulation with a reduced number of degrees of freedom.

Hence the user can *interactively* position the Human Body Model. The main focus of this module is to define the target bones position of the HBM. The final deformation can be obtained with the Fine-Positioning Module or the Contour Deformation Module. Please refer to Positioning in PIPER: overview for a general introduction.

- **Metadata:** The skin and skeleton must be defined by Entity, the kinematics of the HBM is constructed from Joint and Contact. More details are given in the Metadata section.
- **Input Targets**:
  - any user defined target (fixed bone, joint, frame to frame, landmark), see also Target
- Output Targets :
  - the current user defined targets
  - all bones inertia frame position (world frame to inertia frame targets), to work with the Fine-Positioning Module
  - all landmarks position
- **Output**:
  - Model nodes update
  - Scripting
- **Bibliography:** "Stable Constrained Dynamics", Maxime Tournier, Matthieu Nesme, Benjamin Gilles, François Faure, ACM Transactions on Graphics, Proceedings of SIGGRAPH, 2015

Author

Thomas Lemaire - INRIA

6.7.1 Metadata

- Entity
  - **[REQUIRED]** *Bones* and a single *Skin* or a set of skin entities including *Skin_of_head, Skin_of_body_proper, Skin_of_{left,right}_free_upper_limb and Skin_of_{left,right}_free_lower_limb* (see the How to...).
  - *Articular capsules* can be defined to make the flesh slide on the bone around a joint. This prevents flesh 3d elements close to the joints to suffer large deformation and quality degradation when the joint moves. Normals at the capsule surface must point outwards the flesh, hence toward the joint center. In conjunction with capsules, the corresponding bony regions included in the capsule must be defined (see below)
  - *Ligaments* must be defined when a capsule is defined, to prevent them from penetrating into the bones.
  - For constraints such that bone, and ligament collision or sliding contacts, the normals of the meshes must be properly defined pointing outside.
- Joint, Anatomical Joint and Contact are used to create the kinematics model for the positioning.
- Landmark
  - Some landmarks are required for the Spine predictor

  - Based on available landmarks, the ISB Bone Coordinate System (BCS) and Joint Coordinate System (JCS) will be available in the frame controller (Appendix: List of frames)
  - The landmarks can also be controlled with the Landmark controller

- Named Metadata
  - *custom_affine* named metadata define positions where custom affine frames can be added to a deformable entity like skin, capsule or ligament, the naming scheme is *custom_affine__Name_of_entity* - example: *custom_affine__Skin_of_left_arm*
  - *bone_region* named metadata define the bone region which is included in a given capsule, the naming scheme is *bone_region__Name_of_bone__Name_of_capsule* - example: *bone_region__Left_femur__Articular_capsule_of_left_knee_joint*

6.7.2 Simulation model

The simulation model is built using the geometry defined in the PIPER model, the metadata information is used as follow :

- *bones* Entity are simulated as rigid bodies, collision constraints are created with consecutive bones (can be disabled)
- the material inside the *skin* Entity is simulated as a uniform elastic material
- the Joint and Anatomical Joint are added as cinematic constrained
- the Contact defined between bones are also simulated, the distance between the two contact surfaces tries to be maintained
- capsules and ligaments are simulated and collision constraints are created with related bones (can be disabled)

The time needed to build the simulation is typically between less than 1 minute and a few minutes depending on the size of the HBM and the speed of your machine. The simulation is then run in the `SOFA Framework`.

6.7.3 Parameters

the Pre-Position module can be controlled by parameters, they have reasonable default values and can be defined in the Module parameters dialog. These parameters let degrade the precision of the deformation field, which is not of first interest in this module, in favor of loading time and interactivity.

6.7.3.1 bone collision

Enable/disable bone collision

6.7.3.2 articular capsule

Enable/disable articular capsule simulation, which includes : adding *affine frames* to the capsule to better deform them, the same for ligaments entities, and lastly enable collisions between the capsule surface, the ligaments and the bones involved in that articular joint.

6.7.3.3 Number of collision vertices

Default number of closest vertices considered in bone, capsule and ligaments collision, if this number is too low bones can *oscillate* between two collisions. The value can also be changed during the positioning in Simulation control. A value of 0 means all vertices are considered.

6.7.3.4 Frames per capsule

Set the number of *affine frames* sampled for each capsule, increase this number if you think the capsules do not have enough degrees of freedom to react to collisions.

6.7.3.5 custom affine

Enable/disable additionnal affines define in Metadata

6.7.3.6 voxel size

The input HBM geometry is rasterized in a 3D grid to prepare the simulation model. This parameter controls the size of the voxels of this grid. This value can be decreased if you encounter *topological* problems in the interactive model (such as close parts being attached one to each other). This value can be increased to make the module loading faster.

6.7.3.7 voxel coarsening

After the rasterization, various computations can be run on a coarser grid to speed up even ore the module loading. A value of n means 2^n times less voxels, so 1 means no coarsening.

6.7.3.8 affine density

This parameter controls the density of *affine* (deformable) degrees of freedom in the flesh. For this Interactive module it can be kept to 0.

6.7.3.9 tibia/patella constraint

Enable/disable a distance constraint between the patella (center of gravity) and the tibial tuberosity landmark (which then must be defined in the metadata)

6.7.4 User interface

This module UI is composed of a 3D Display and several toolbox windows which are described in the following sections. On the left side a set of buttons let the user select the active mouse interaction, an other set to show/hide the different toolbox windows.

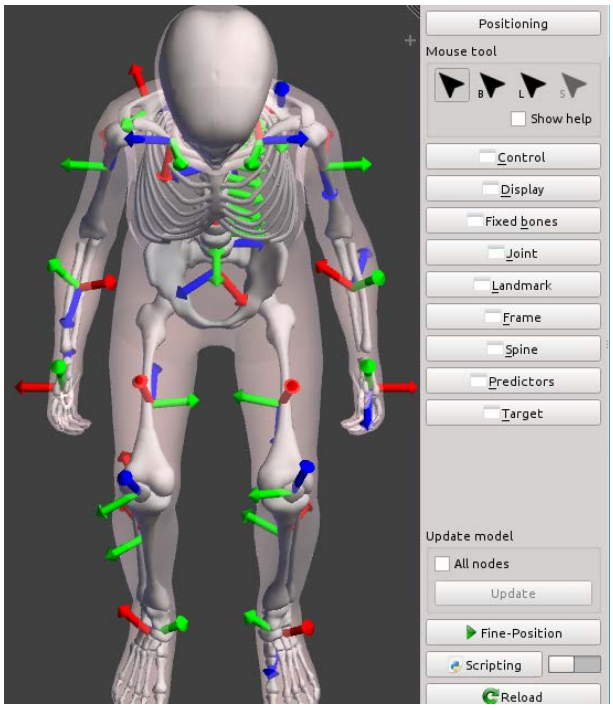


Figure 22: 3D view, bone frames are displayed

6.7.5 Interactive positioning

6.7.5.1 Simulation control

- The positioning process can be started, stopped and reset

- The simulation time step can be adjusted, increase it if you are in a hurry, reduce it when the simulation gets unstable

- The Number of collision vertices can be changed,

- If bone collision are enabled, they can be turn on and off on the fly, to increase simulation frame rate.

- The *Run script* switch triggers the execution of the lastly run python script after each simulation step (see Scripting).

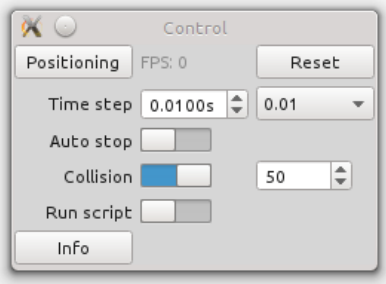


Figure 23: Positioning controller

A set of controllers allow the user to drive the positioning process, they are described in the following sections.

6.7.5.2 Fixed bone controller

Select the bones which remain fixed during the simulation in the bones tree, several bones can be selected at once when a body region is selected. Fixed bones can also be selected using the *Fixed bone* mouse interaction. At least one bone should be fixed, if not your model is likely to "fly around".

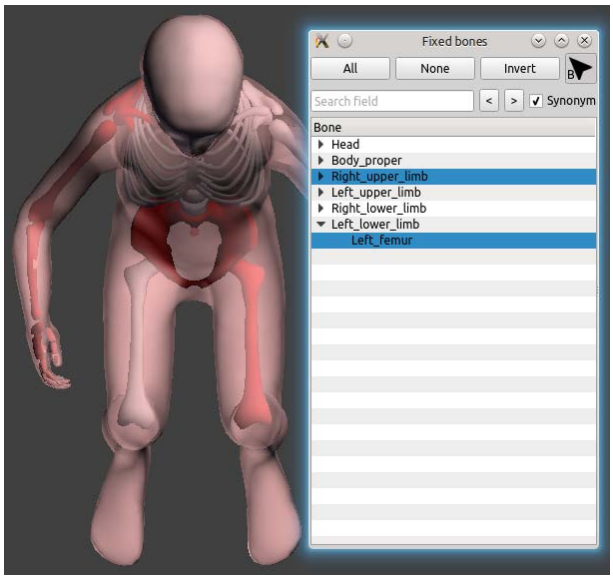


Figure 24: Fixed bones appear in red

6.7.5.3 Landmark controller

The landmark controller let the user add constraints on any set of landmark coordinates [x,y,z]. Using the mouse interaction, clicking on a *yellow* landmark select it to add a controller on it. Clicking on a *green* target makes appear a manipulator according to the controlled DoF of that landmark, the target can then be moved interactively.

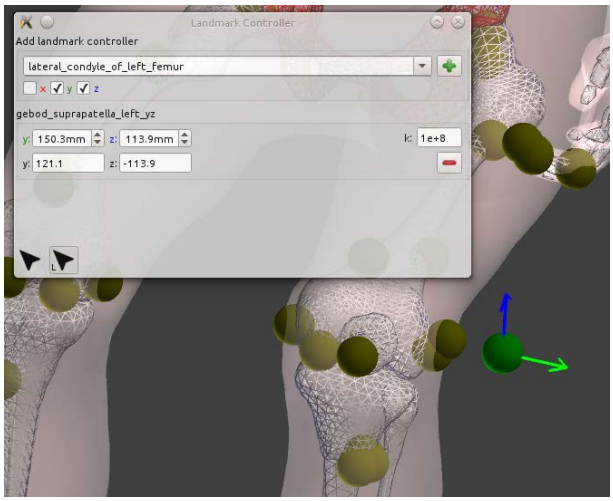


Figure 25: Landmarks are displayed in yellow, targets in green. One target is being set interactively

6.7.5.4 Joint controller

The cinematic joints defined in the HBM can be controlled (according to their degrees of freedom). Joint frames can be displayed individually for each controller.

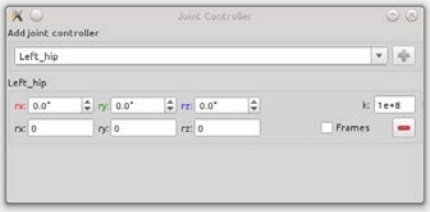


Figure 26: Joint controller

6.7.5.5 Frame to frame controller

A frame controller is created either:

- between a reference frame and a bone entity, then the relative transformation of that entity in the reference frame is constrained
- or between two frames, in that case the absolute transformation between the 2 frames is constrained.

If $target$ is the target transformation value, the objective is to get $frame_1 \circ target = frame_2$ The available frames are:

- the anatomical frames defined by landmarks (automatically created when landmarks are defined on the)
- the world frames, prefixed by `W_`
- the bone inertia frames (automatically created for each bone entity), prefixed by `B_` Also, frames can be displayed individually for each controller.

Warning

Either all or none rotation parameters can be controlled.

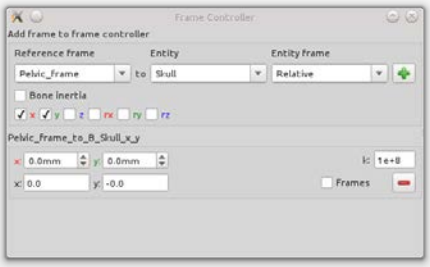


Figure 27: Frame to frame controller

6.7.5.6 Spine controller

The spine controller enables the user to draw a target Bezier curve for the spine shape. Before creating the controller, a set of vertebrae has to be selected, they are used as Bezier control points (their center of gravity) and the initial curve is computed to go through these points. Using the spine mouse interaction, the curve can be freely modified in 3D. To ease this task you can show/hide disrupting objects in the Display settings.

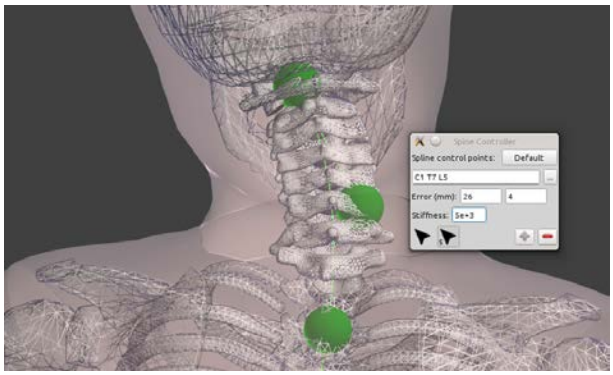


Figure 28: Interactive spine controller, the target Bezier curves and controlled points are drawn in green, also the distance from the vertebra to the curve

6.7.5.7 Positioning predictors

The predictors exploit *a priori* knowledge (statistical, bibliography study,...) to produce realistic positioning targets. These targets are then used to drive the positioning process.

6.7.5.7.1 Spine predictor

- **Parameters:** *Posture A, Posture B, Target posture, thoraco-lumbar lateral flexion angle, cervical lateral flexion angle, Align on gravity, gravity vector, Stiffness*
- **Metadata:**
  - the ISB landmarks required to compute the vertebrae and the pelvis body coordinate system needs to be defined on the model (see Appendix: List of frames). You can obtain a list of these landmarks by using the `printFrameLandmarks.py` standard script, running it with arguments *"Spine ISB_BCS"* and *"Pelvic_skeleton ISB_BCS"*
  - the *Pelvic_skeleton* entity is required
  - the *gravity* vector
- **Intended use**:
  - The spine predictor is to be used as a quick and easy physiological pre-positioning tool for the spine segment.
  - A possible workflow may be e.g. to use the spine predictor to predict a flexed forward posture and to use any other option (e.g. direct Joint angle definition or frame controller control followed by positioning) to modify the orientation of the head or cervical spine (e.g. to ensure horizontal vision), or to adapt the spinal shape locally (e.g. to a given seat).

Authors: Thomas LEMAIRE - INRIA / Bertrand FRECHEDE - UCBL/IFSTTAR

Overview

The spine predictor computes a likely shape of the spine by interpolating splines between known physiological end-postures. These end-postures represent average postures obtained from published postural in-vivo data in the sagittal plane. As of Piper version 1.0.0 the following end-postures can be chosen by the user: supine, standing erect, seated erect, seated slouch, seated driving, forward-flexed (seated). Lateral bending is controlled separately by the user by defining target angles, but relies on similar interpolations between a reference sagittal posture and published in-vivo Range of Motion (RoM) in lateral bending.

Using the tool

Upon opening the tool the user selects (i) two end-postures (*Posture A* and *Posture B*) and (ii) a postural parameter (*Target posture* cursor in the range [-0.2 1.2]). This parameter represents a percentage of transformation between the two known physiological end-postures. As the end-postures' global orientation (as defined by their pelvis orientation) is defined relative to the gravity vector, so is the resulting posture by default (*align on gravity* option *on*). In

practice, this is performed by adding a Frame to frame controller to constrain the pelvis absolute orientation and by rotating the pelvis by its target interpolated value between the two end postures. The vertical is taken as the gravity vector.

Alternatively, the spine resulting from the *Target posture* value can be aligned to the initial L5 vertebra (*align on gravity* option *off*). In this case, the resulting posture is not 'likely' in the sense that the spinal shape is not associated with a likely global orientation of the pelvis. Nevertheless, it remains a physiologically acceptable solution among others, thus allowing to define a range of possible spinal postures while keeping the pelvis in its initial orientation.

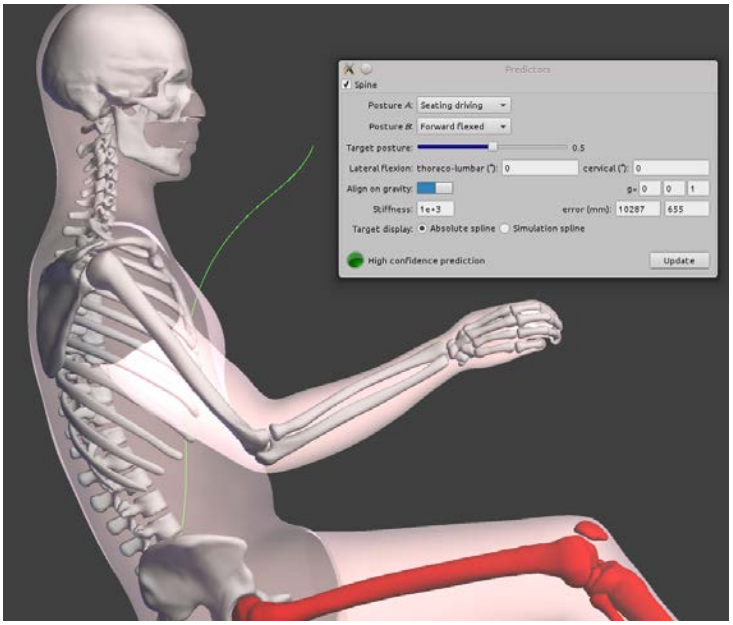


Figure 29: Spine predictor input parameters and output target spline in green

Lateral flexion is controlled independently by entering angles in the *thoraco-lumbar* and *cervical* fields. Max. values for these angles are constrained by the physiological lateral (RoM) of the spine. Thoraco-lumbar and cervical curvature prediction are based on published kinematic coordination laws (ratios between the cervical, thoracic and lumbar angles) for the spine in lateral flexion. However results of this prediction (including when used in combination with the above presented interpolation in the sagittal plane) are not considered to be more 'likely' than others as they do not rely on reference postural knowledge but only on maximum RoM distribution within the spine.

Additionally, a visual cue (colored circle) and text information is provided to the user as an indicative level of confidence for the prediction, as follows:

- Green circle / 'high confidence prediction': this is the case when the transformation is applied in the sagittal plane, within the [0 1] interval for *Target posture* interpolation.
- Orange circle / 'medium confidence prediction': this is the case if:
  - The transformation is applied in the sagittal plane within the [-0.1 0] or [1 1.1] intervals for *Target posture* interpolation,
  - OR The transformation is applied outside of the sagittal plane within the following lateral flexion RoM limits: +/- 7.4° for the cervical angle and +/- 27.7° for the TL angle.
- Red circle / 'low confidence prediction': this is the case if:
  - The transformation is applied in the sagittal plane within the [-0.2 -0.1[or]1.1 1.2] intervals for *Target posture* interpolation,
  - OR Any other combination.

If the *align on gravity* option is *off*, these levels of confidence concern only the resulting curvature of the spine and the fact that those are within measured RoMs. If it is *on*, they will also give an indication on the likeliness of the posture (i.e. including spinal curvatures and rotation of the pelvis) based on the source and target end-postures.

Limitations

- The computation of a prediction based on changes between known, absolute physiological postures, requires that any HBM's spine in input will be transformed to a reference posture in the sagittal plane (as defined by the pelvis bones). This is achieved by performing an initial positioning in this posture prior to computing the transformation from *Posture A* to *Posture B*. This step is transparent for the user. However it means that subject-specific curvatures (e.g. aged/scoliotic subject) that would exist in the HBM or Piper model prior to the use of the spine predictor will be lost through the process. Alternate octave scripts are provided as part of the release that allow applying predicted transformations directly to the input spine without going through the initial positioning phase.
- The user's target/chosen parameters are not saved as part of the history.

6.7.6 Display settings

The display toolbox enables to show/hide (and even draw in wireframe) several items:

- the skin envelop of the model in semi translucent pink if the skin entity has been defined in metadata Entity.
- the bones geometry in white (for bones Entity defined in the PIPER model)
- environment objects
- landmarks, joints frames, anatomical and bone inertia frames, world attached frames
- contact violation as yellow lines
- mesh normals

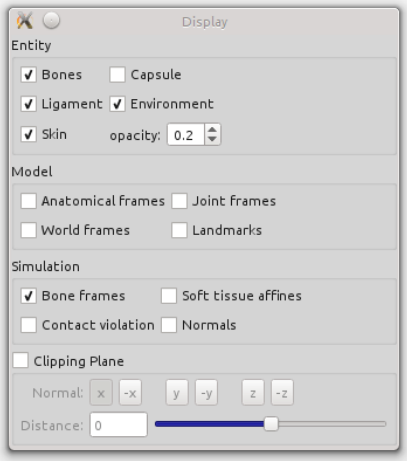


Figure 30: Display settings, bones displayed as wireframe

6.7.7 Output Targets

Several types of targets can be created in the application target list:

- the currently defined **user targets**, the main interest interest is to replay the same positioning later, or on an other model, or using batch mode
- all HBM Landmark metadata, these targets can then be used in the Contour Deformation Module

- all **bones** inertia frames absolute position, these targets can be used to deform the model in the Fine-Positioning Module

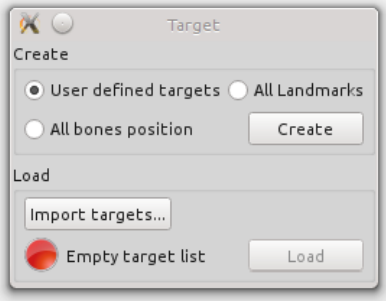


Figure 31: Target creation and loading

6.7.8 Model nodes update

The computation of model nodes must be activated first, then the PIPER model nodes coordinate can be updated. This module main functionality is **not** to compute a precise deformation field for the HBM, consider using the Fine-Positioning Module or the Contour Deformation Module for this purpose.

The *Deformation* button sends you directly to the Fine-Positioning Module executing the following sequence:

1. the current user-defined targets are saved to the application target list
2. the Fine-Positioning Module is started
3. the previously saved targets are loaded in the Deformation module

6.7.9 Scripting

The module supports specific scripting functions to access intermediate states during the positioning process and export any data that could drive an external simulation to achieve the full HBM positioning.

6.7.9.1 Specific python objects

In addition to the regular PIPER scripting (see Scripting and Batch mode), a script run along with this module can access the currently defined controllers with these 3 python objects: `jointController`, `landmarkController`, `frameController`.

```
1 # Print information for a controller.
2 # Support landmark, joint and frame controllers
3 #
4 # arguments:
5 #  controller_name
6
7 # author: Thomas Lemaire date: 2017
8
9 import piper.app
10
11 if (2 != len(sys.argv)):
12     raise RuntimeError("Invalid arguments")
13 else:
14     controllerName = sys.argv[1]
15     controller = None
16     if controllerName in landmarkController.getControllerNames():
17         controller = landmarkController
18     elif controllerName in jointController.getControllerNames():
19         controller = jointController
20     elif controllerName in frameController.getControllerNames():
```

```
21         controller = frameController
22     if not controller is None:
23         piper.app.logInfo("Controller: {0} - target: {1} - mask: {2} - position: {3}".format(controllerName
       , controller.getTarget(controllerName), controller.getMask(controllerName), controller.getCurrentPosition(
       controllerName)))
24     else:
25         piper.app.logInfo("Controller {0} does not exist".format(controllerName))
```

Also the current position of the HBM can be saved in the Model history using the `simulationController`

```
1 # Add HBM current nodes position into the history.
2 # This script requires that "all nodes" update is activated
3 #
4 # arguments:
5 #  history_name_prefix
6
7 # author: Thomas Lemaire date: 2017
8
9 import piper.hbm
10 import piper.app
11
12 model = piper.app.Context.instance().project().model()
13
14 # if the model is empty, we can do nothing
15 if model.empty():
16     piper.app.logWarning("Model is empty")
17 elif not simulationController.isAllNodesActivated():
18     piper.app.logWarning("\"all nodes\" update is not activated")
19 elif (2 != len(sys.argv)):
20     raise RuntimeError("Invalid arguments")
21 else:
22     historyPrefix = sys.argv[1]
23     global historyPrefixCounter
24     if not "historyPrefixCounter" in globals():
25         # global variable to keep track of number, must be created only once
26         historyPrefixCounter = dict()
27     if not historyPrefix in historyPrefixCounter:
28         historyPrefixCounter[historyPrefix] = 0
29     historyName = "{0}_{1}".format(historyPrefix, historyPrefixCounter[historyPrefix])
30     piper.app.Context.instance().addNewHistory(historyName)
31     historyPrefixCounter[historyPrefix]+=1
32     simulationController.updateModelNodes()
33     piper.app.logInfo("Current HBM nodes position saved in {0}".format(historyName))
34
35
36
```

6.7.10 How to...

6.7.10.1 How to fix one bone with respect to an other one ?

- If there is a robotic joint defined between the two bones, you can add a *joint controller* which constraints the relative bones position to stay the same
- In the general case you can add a *frame controller* between both bones (either 3 rotations, or 3 translations or both) to achieve it

6.7.10.2 How to reach a complex position ?

You can concentrate on positioning a sub part of the HBM, then fix the corresponding bones and continue the positioning task on an other sub part of the HBM.

6.7.10.3 The skin of the arm looks attached to the body, how to fix it ?

Depending of the initial position of your HBM, the rasterization process used to set up the simulation might *attach* close parts of the skin. To get rid of this kind of problems you can split your skin entity in several entities, at least close skin parts should be put in separate entities.

6.7.10.4 The simulation is unstable even with no target

If you have bone collision enabled try to disable it in the Module parameters, if this fixes the problem it is likely that you have wrongly defined normals for some bone meshes. You can check the normals using the Display settings in this module, or the Generic viewer in the Check Module, and fix them in your favorite preprocessor software.

6.8 Fine-Positioning Module

6.8.1 Overview

This module implements the same simulation as in the Pre-Positioning Module. Compared to it, more degrees of freedom are added to simulate the soft tissue so as to get a high quality deformation of the model. As a consequence the simulation runs slower and is not interactive.

- **Parameters:** voxelSize, affine density (voxelCoarsening is 1)
- **Metadata:** same as Pre-Positioning Module
- **Target Input**: any user defined target (fixed bone, joint, frame to frame, landmark) can be loaded, however the standard workflow is to load bones inertia position created in the Pre-Positioning Module.
- **Target Output**: None
- **Output**: updated model nodes

Author

Thomas Lemaire - INRIA

6.8.2 Parameters

- **voxel size**: decrease this value to have more precise deformation field, increase it to have faster loading time
- **affine density**: increase this value to get a better quality deformation, decrease it to have faster loading and faster simulation

6.8.3 User interface

This module UI share is similar to the Pre-Positioning Module UI, with less components since it does not provide user interaction with the model.

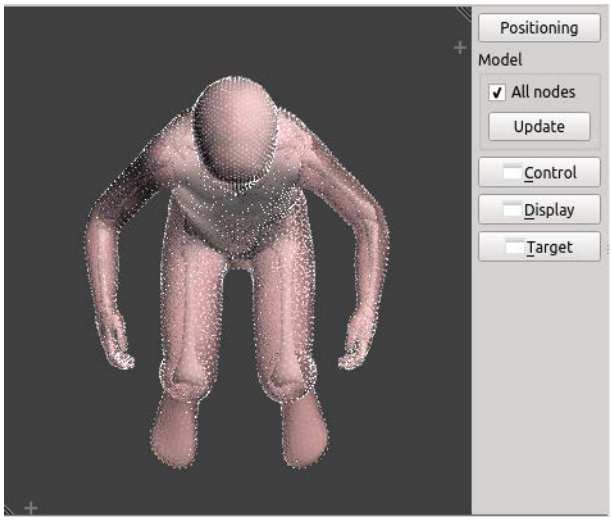


Figure 32: 3D view, model nodes are computed

6.9 Smoothing module

6.9.1 Overview

After some transformations (positioning in particular), it is frequent that the element quality is degraded compared to the original model. This can occur on the surface or inside the mesh (3D elements) and it can lead to inverted elements (negative volume or negative jacobian) which prevent the simulation from starting. This module provides different tools to try to help with this problems.

First, the element quality needs to be assessed. There are many metrics and definitions used to define element quality depending on the FE solver and pre-processor. It is assumed that the user can always export the HBM and to compute its favorite quality metrics in another pre-processor (which PIPER does not aim to replace). Within PIPER, two libraries are used to compute quality metrics:

- VTK toolkit Quality metrics, which is used for the transformation smoother.
- MESQUITE Quality metrics, which support the mesh smoothing in the Mesquite library

They will provide results different results. In both cases, the variation of element quality (before and after transformation) can be computed which can be helpful to detect problematic regions for smoothing and assess the quality of the transformation (independently from the quality of the original model).

Then to try to improve the quality, three methods are provided. Their principles are summarized below:

- Improving mesh quality using Mesquite metrics : a 3D mesh optimizer (using the Mesquite library) attempts to optimize the quality metrics within a 3D mesh without changing its boundaries. As a result, if the problems are located on the surface of the mesh, the problem cannot be completely solved.
- Smoothing the surface of an entity : a simple Taubin smoothing algorithm is available to smooth the surface of entities. It is often useful for the skin after positioning. A Surface crease detection algorithm can be used in preparation to detect the regions of interest for smoothing. As only the surface is smoothed, another smoothing methods for the 3D mesh is typically required after.
- Transformation smoothing : the geometrical transformation between two models is smoothed. This is not a mesh smoothing method per say (the method is not aware of the mesh). This approach attempts to maintain the original element quality by minimizing the local distortions. In practice, it can solve many of the local problems resulting from the transformations without generating unwanted penetrations (e.g. after positioning).

Overview of the smoothing methods

Method	Applies to	Strength and usage scenario
Mesquite Mesh optimizer	3D mesh (inside)	Optimize mesh. Use alone or after the others.
Surface smoother	Surface mesh	Local defects (on skin). Use first.
Transformation smoother	All (not mesh aware)	Minimize distortion without generating penetrations. Use second.

All three methods can be applied even if the model transformation was not made in PIPER (the transformation smoother can smooth between two arbitrary models).

6.9.2 Module GUI

There are currently four different methods available in this module. Only one can be active at a time. To switch between them, use the buttons in the right panel, marked by number "1" in the Figure Module Main Window. Clicking them will also open a window with options of the given method. See below for details on the individual methods.
The results display has two modes - information grid and 3D view. The mode can be changed using the two buttons on the right panel marked by number "2" in the Figure Module Main Window.
The information grid mode is currently used only by the MESQUITE Quality metrics method. It displays the amounts

of low quality elements in individual parts of the FE model as text.
The second mode switches to a standard 3D view of the model. Most of the quality methods have a way to mark the low quality or otherwise interesting elements in the 3D view. Also, standard 3D view tools such as metadata showing/hiding, picking etc. are of course available in this mode and can be used e.g. to select which parts of the model to apply a mesh optimization to.

Hint: Most of the mesh optimization methods work with the entire, "full", model (including all nodes and elements). The visualization of finite element model will therefore be activated by default and the entity visualization (PIPER model) automatically switched off. If needed, you can switch it back via the "Display -> Entity" window, using the checkboxes on the bottom of the window. But keep in mind that if you have both modes turned on, you will likely see the model twice, once as a full model, once as entities.

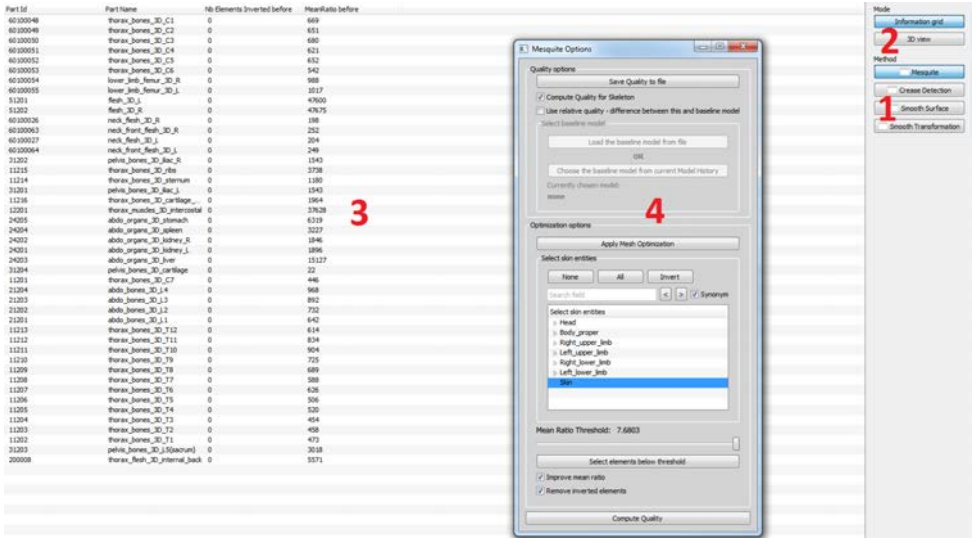


Figure 33: Module Main Window

6.9.3 VTK toolkit Quality metrics

The VTK toolkit integrates quality metrics defined in the Verdict Library. The metrics are computed on 2D and 3D elements (pyramid and penta element metrics are not available in the Verdict Library).
These metrics are not used for any optimization. It is possible to display them in every module that utilizes the 3D view. You can bring it up using the "Element Quality" button in the right panel.
Figure Color by Quality shows the options window for this method. You can choose from a number of quality metrics and set-up up to three quality ranges, each of which will be colored by a different color scheme. Upon pressing the "Display" button, the quality will be computed for each element for which the metric is defined and the elements will be colored. You can save the results to a file using the "Save Quality" button.
Setting up the ranges is simple - tick a checkbox for each range you want to use, then enter the desired thresholds. The extremes of the ranges will be automatically modified so that no two ranges overlap.
The two comboboxes after the range values are used to set up which color will be used with which range. By default, as in the Figure Color by Quality, only the first combobox has some value. In such a case, each element that belongs to this range will be colored by this value. If you specify a second color of the range, the color associated with the element will be a linear interpolation of the two colors. E.g. if you specify a range from 0 to 1 and set colors from green to red, an element with a quality 0.5 will be yellow.

Hint: you can easily create a range that goes from dark to light tones of your favourite color. Simply specify black as the second color to create a range that goes darker with increasing values, or white to create a range that grows lighter. Or use your color as the second one and white/black as first for a similar, yet different effect.

To select elements within some range, use the "Select" button. The checkboxes next to it let you specify which range to include in the selection. Note that using the selection, the 3D viewer automatically switch to rendering the elements based on the selection, not based on the quality, i.e. you will see selected elements in green color, deselected in white. You can then click the "Compute" button again to switch back to coloring the elements by the quality, but the selected elements will remain selected (only not "highlighted").
If you check the "Use relative quality" checkbox, you can specify a "baseline" model, i.e. a different version of your current model (the node and element counts must match!), e.g. before some deformation. You can either specify the model from file or from the Model History, i.e. use the version of the model before you applied some other PIPER operation on it. The quality will then be computed as a difference between the quality of the current model and the baseline one. This way, you can easily see if the deformation you created increased or decreased the element quality of the model.

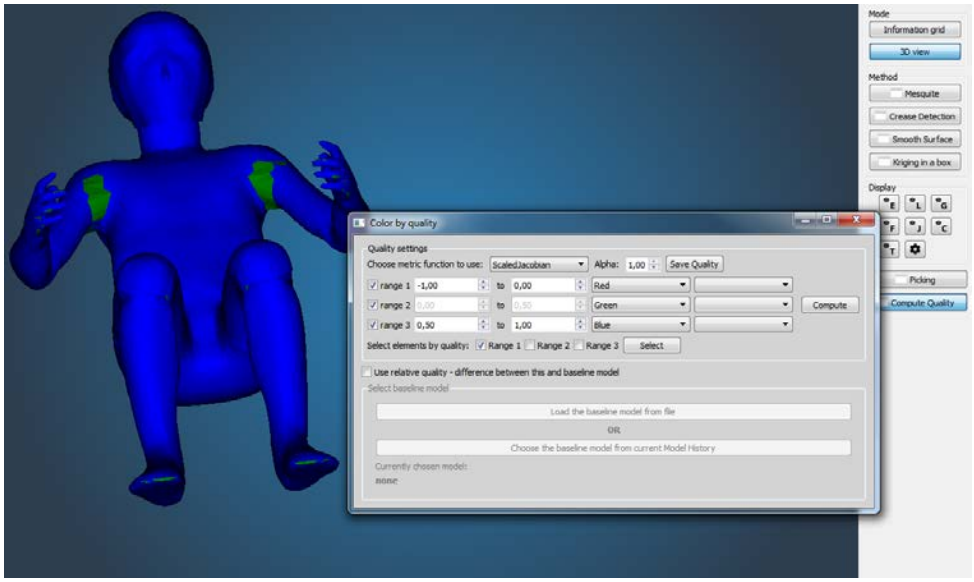


Figure 34: Color by Quality

6.9.4 MESQUITE Quality metrics

MESQUITE Quality metrics are defined on 2D and 3D elements but only 3D elements metrics are considered in this module. Inverse mean ratio metric and untangle Beta metric (to detect inverted elements) are computed.

To perform a quality analysis using Mesquite, the user has to:

- Open the Mesquite Options window using the "Mesquite" button on the right panel (see number 1 in Figure Module Main Window - this opens the window "4" in the same figure)
- Activate/Deactivate the option to include bones in the analysis - use the "Compute Quality for Skeleton" checkbox
- Click on "Compute Quality" button. When computation is done, results are presented in the table (see Figure Module Main Window, part 3)
- Alternatively, a "baseline" model can be specified, either from a file or from the model history. If a baseline model is used, the results are not absolute values of the quality metric, but instead values relative to this baseline model, computed as quality of the current model minus the quality of the baseline model.

Results of the quality evaluation are presented in a table with each line corresponds to part defined in the Finite Element Model and the following columns:

- The Part Id
- The Part Name (if defined in Finite Element Model)

- For each quality metric, the number of elements that do not respect current quality criteria value
- If optimization is applied to the mesh, additional columns with the number of elements that do not respect current quality criteria value after the optmization are added

The button "Save Quality" export the quality metrics for all elements of the model.
The button "Apply Mesh Optimization" launches the optimisation process - see Improving mesh quality using Mesquite metrics

6.9.5 Improving mesh quality using Mesquite metrics

The second part of the Mesquite Options window allows you to perform some mesh optimization. The GUI for optimization is enabled only if quality is computed and up to date - you have to use the "Compute Quality" button before the optimization.
The optimization is applied on each part (list in the table) with elements that not respect quality criteria. When mesh optimization is applied, each part is optimized individually. Nodes at the free surface of the part are defined as fixed nodes. When an element with quality that does not respect a specified criterion has nodes that are used in the entities that make the skin (use the provided "Select skin entities" box to defines which entities belong to skin), these nodes are not set fixed to allow improvement of this element. The criteria to use can be modified in the "Optimization options" part of the Mesquite Options window (see Figure Module Main Window, window labeled by 4), namely you can specify to improve the mean ratio metric and also set the threshold that should be used to determine what is an acceptable quality; and you can specify whether to attempt to remove inverted elements or not (this is only a binary choice with no further parameters).
You can use the "Select elements below threshold" button to perform a selection of elements. If you need to select elements above the threshold instead, simply use the "invert element selection" from the standard `Picking` menu.
If you specified that you want to use the relative quality in the "Quality options" part of the Mesquite Options window, the value of the mean ratio threshold will also be relative - value of 0 will mean elements that have the same quality both in the current and baseline mesh, negative values will be elements that have lower quality now than in the baseline mesh and positive values mean elements that improve their quality in comparison to the baseline mesh.

6.9.6 Smoothing the surface of an entity

The surface smoothing method uses a windowed sinc function interpolation filter to relax the node positions on a surface of an entity (also known as Taubin smoothing). This means that only 2D elements (triangles, quadrilaterals...) will be take into account for the smoothing. Therefore, this method is useful in preprocessing surfaces that you intend to use as targets for Transformation smoothing or similar purposes, or fixing small defects, but will not increase the quality of the model too much in terms of its suitability for FE simulation as problems are often in the 3D elements.
After the smoothing is finished, a new model history is always added, so you can undo the smoothing at will. Apart from starting the smoothing process by pressing the "Smooth surface" button, there are three options you can change in the GUI of the method:

- Processed entity: the table in the top half of the window (see Figure Surface smoothing) allows you to set which entities to smooth. An entity named "Skin" will be selected as default if it exists, if it does not, nothing will be selected. Again, keep in mind that only 2D elements will be smoothed for any of the entity you select. The selected entities are merged into one surface before the process. This ensure correct behaviour even on "seams" between entities. That is useful if you have for example the skin of your model defined as several entities.
- Smoothing parameters: there are two parameters of the smoothing algorithm that can be changed:
  1. The number of iterations, which in essence specifies the degree of the polynomial used for the interpolation - see the `documentation of the smoothing filter` for details. The higher, the smoother will the result be. A small amount of iterations (< 5) should be avoided to get good results.
  2. The "pass band value", a number between 0 and 2, which specifies what node valence distribution to consider smooth. The lower, the smoother (i.e. more regular) will the resulting mesh be.

The default values, number of iterations 20 and pass band value 0.1, are a reasonable setting which will be sufficient in the most usual cases. The Figures Surface smoothing, before and Surface smoothing, after shows an example of the smoothing with those settings applied to the skin in a shoulder area.

- Smooth only selected: the checkbox in the bottom of the option window allows to specify whether the smoothing shall be applied to the whole entity or only the nodes that are selected. Node picking tools are available in the standard "Picking" toolbox. As Figure Surface smoothing, before shows, the picked nodes are highlighted as green spheres on the models. Only the selected nodes will be smoothed if this checkbox is ticked. The crease detection method described below can be especially useful to select the regions of interest for surface smoothing
  Hint: Note that the filter operates only on selected **nodes**, not elements! However, if you want to smooth an area that you selected using element selection, you can simply use the "Select nodes of selected elements" button in the "Picking" toolbox to select the appropriate nodes and then run the smoothing.

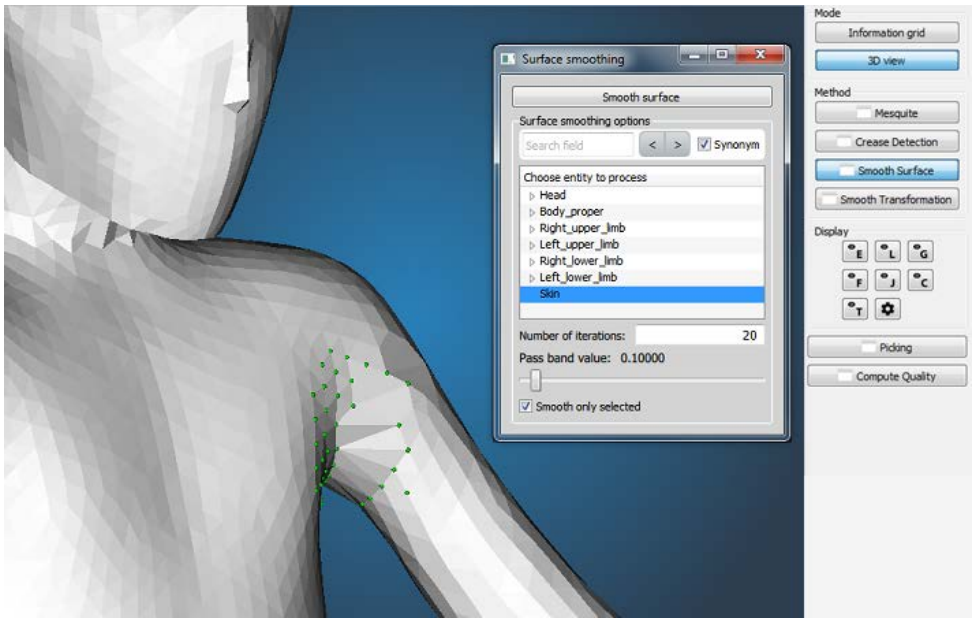


Figure 35: Surface smoothing, before

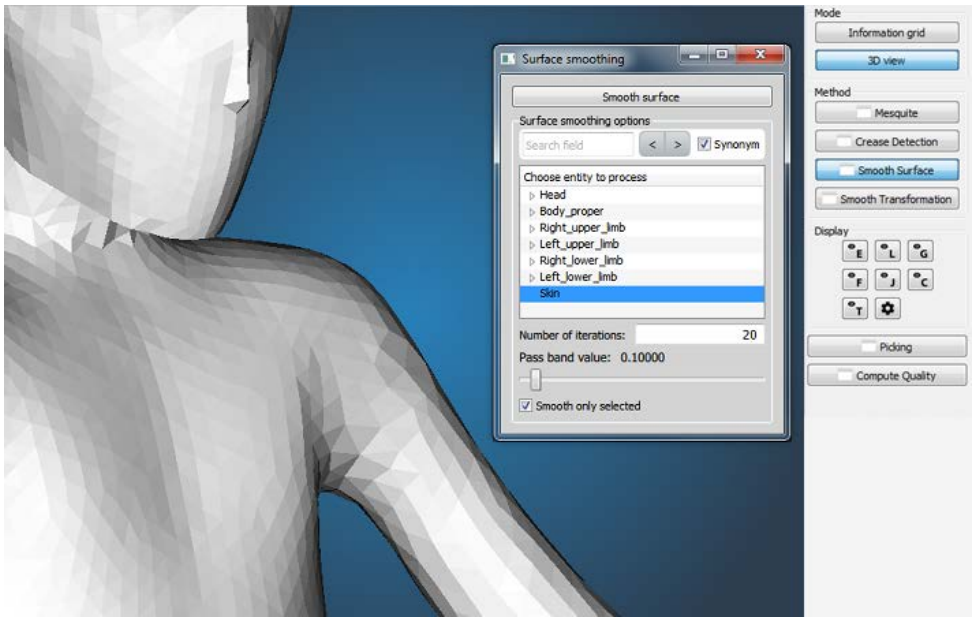


Figure 36: Surface smoothing, after

More details can be found in the documentation of the smoothing filter or in the original paper: Optimal Surface Smoothing as Filter Design, G. Taubin, T. Zhang and G. Golub, IBM tech report RC-20404, 3/12/96.

6.9.6.1 Surface crease detection

One of the most common types of unwanted deformation that happens to an FE mesh after positioning is that unnaturally sharp creases appear around areas that were bent. This method, which require to specify a baseline model, automatically detects those areas and selects them. It does not have any tool to remove this damage, but the selection can be carried over and used as a target for other optimization methods, such as Transformation smoothing and Smoothing the surface of an entity.
The method uses a simple algorithm that divides the mesh into a set of clusters of elements based on the change of the dihedral angle of elements (the angle between two neighboring elements) before and after a deformation. If the change of dihedral angle between two elements is larger than a specified threshold, each of those elements is added to a different cluster. In the example in Figure Surface crease damage detection, elbow was extended. Ideally, this will put elements in the forearm and the rest of the body to two different clusters, as the change of dihedral angle around the elbow will be large. However, in practice, the positioning methods will likely not be able to create a smooth transition in this area, instead some unnaturally sharp transitions will appear around the elbow, as we can see in the Figure. Using the clustering algorithms, elements that form these sharp transitions will create narrow clusters, only few or even one element "thick", i.e. most of the elements of the given cluster will be on its boundary. This method detects and selects those clusters.

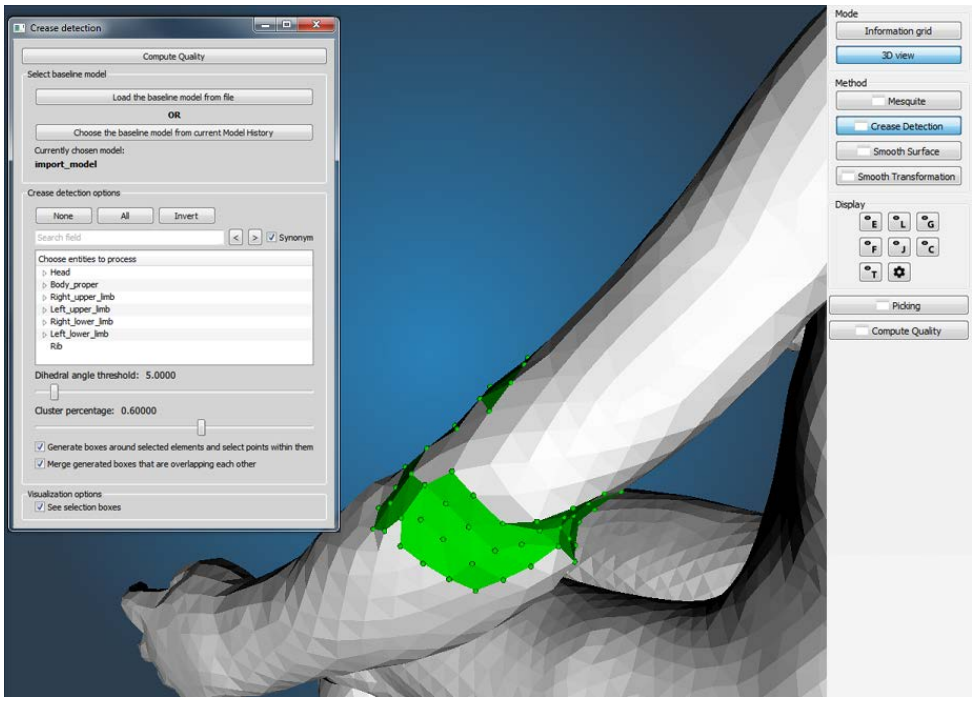


Figure 37: Surface crease damage detection

The options window of the crease detection method is divided into these four parts, going from top to bottom (please refer to Figure Surface crease damage detection):

1. The Compute Quality button: performs the detection and selects the elements and points that belong to clusters that are evaluated as damaged. Note that this will reset all selection that you might have made previously! The button will not be available until you load a model and select a baseline model.

2. Baseline model selection: this section allows you to either load a baseline model from a PIPER project file (the button will open a standard "open file" dialog), or select an entry from the current model history, i.e. choose a version of the current model before you applied some PIPER action to it (the button will open a dialog where you select one of the model history names and confirm). The name of the currently chosen model - either a model history name or path to the file - is written below the two buttons. Note that the baseline model is shared among all other methods, so you do not have to re-load the file / re-select the history when you go to another mesh optimization method.

3. Detection options: this section contains the main settings of the algorithm. First, you need to select which entities to process. As was the case with Smoothing the surface of an entity, the crease detection works only on the surface of the mesh. The "Skin" entity is selected by default if it exists, nothing is selected if it does not exist. The selected entities are merged into one surface before the process. This ensure correct behaviour even on "seams" between entities. That is useful if you have for example the skin of your model defined as several entities, but also please realize that if you select multiple entities that are disjoint, the result might not be very useful to you.
   Below the entity selection are two sliders that change the parameters of the algorithm: angle threshold, which decides whether two elements are added to the same or different clusters - the lower it is, the less tolerant will the algorithm be - for threshold 0, it will put any two elements to different clusters; and the percentage of elements that must be on the boundary for the cluster to be considered narrow - the lower it is, the more clusters will be evaluated as damage (although in practice, the really damaged clusters have very high percentage, $> 90\%$, while the non-damaged cluster have very low, $< 10\%$, so in most cases, changing the value of this slider will not have a large impact).
   Lastly, the two checkboxes below the sliders serve to automatically create selection boxes around the damaged areas, which is especially helpful for Transformation smoothing. If you check the "Generate boxes" checkbox, boxes around the damaged clusters will be created and all points inside them will be selected. This will likely widen the area selected by the detector - Figure Selection by box shows a box created around

the areas selected in Figure Surface crease damage detection. The second checkbox allows you to specify whether overlapping boxes shall be merged or not. Non-merged boxes will contain less elements and nodes, but Transformation smoothing is more effective if the area of effect has at least a few hundred elements, so if you want to create the boxes for this purpose, it is recommended to check the merge boxes option. `Hint: you can work with the boxes created with crease detector just like with any other boxes you created with the "Picking" toolbox using the box manager inside "Picking" toolbox.`

4. Visualization options: currently, there is only one option for convenience - you can turn on or off visualization of boxes (you can do the same from the "Picking" toolbox).

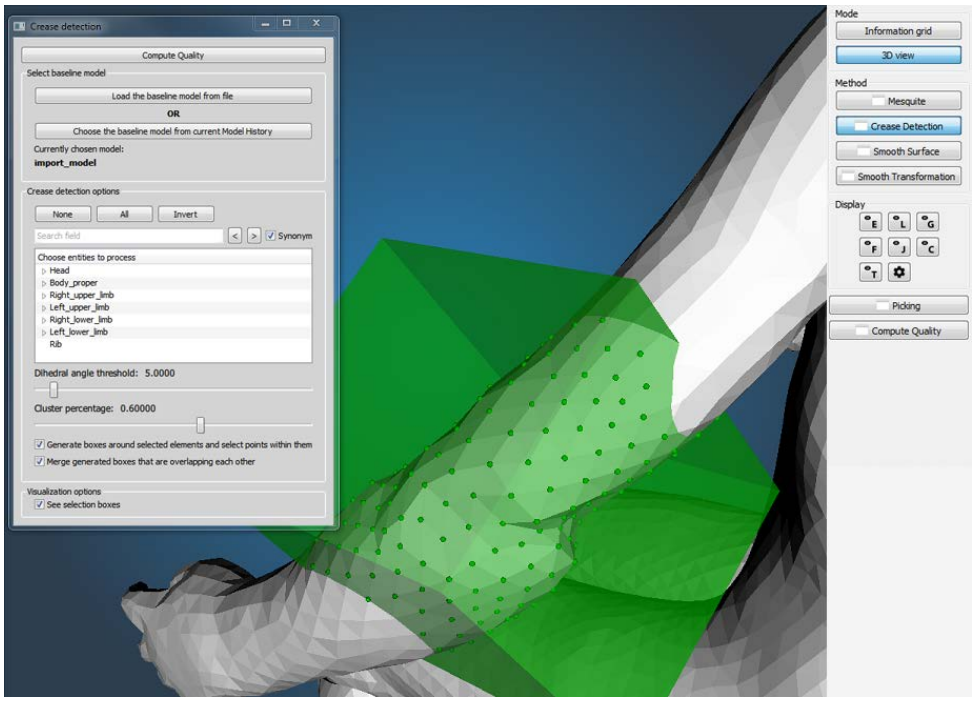


Figure 38: Surface crease damage detection, selection by box

6.9.7 Transformation smoothing

6.9.7.1 Overview

Mesh smoothing algorithms usually smooth the mesh based on some general quality metrics. However, smoothing disconnected groups of elements independently could lead to penetrations between them if the boundary is not kept, and to insufficient correction if it is. In the case of positioning and personalization, the user has access to the mesh before the deformation that potentially caused the damaged (assuming the initial model quality is acceptable). This transformation smoothing method takes this "baseline model" and the target model obtained from positioning or personalization and creates a smooth transformation of the former to the latter. This has the advantage of working with the entire model at once while not generating penetrations (as long as the transformation is smooth). A publication on this approach is being prepared. PIPER currently provides one algorithm to perform this transformation smoothing. It is based on dual Kriging interpolation (see Kriging Module).

6.9.7.2 Basic smoothing with default parameters

The first step is to select a baseline model (from file or model history. Please refer to the Surface crease detection section above for details) as the transformation will be smoothed between the baseline and the current model.

The region in which the transformation is smoothed is solely based on selection boxes. You can create the selection boxes manually using the "Picking" toolbox or with the Surface crease detection. When using the "Picking" toolbox, it does not matter whether you are creating the boxes in the node selection or element selection mode - only the

boxes themselves will be considered.
Hint: check the "See selection boxes" in order to visualize them. Hold CTRL when drawing the box to activate the de-selection mode. This way, nothing inside the box will actually be selected.

Surfaces of entities selected with "Select target entities" are used to constrain the transformation, i.e. taken as a target. This means that nodes on the selected entities will be considered as control points for the kriging (but of course only those that are in a selection box) and all other nodes will be transformed according to them.

Once target entities and selection boxes have been selected, you can start the smoothing by pushing the "Smooth by kriging" button. **Please note that for larger models and/or selected areas, this action can take up to several minutes.**

Remarks

- it is advised to use other techniques such as Smoothing the surface of an entity or external surface editing to pre-process the surface of the entities selected for target until they are deemed acceptable (as transformation smoothing will not affect them)
- for transformation smoothing after positioning, bones should be selected to ensure they are kept rigid.
- you only want to smooth the soft tissues between the bones and the skin, then both have to be selected as targets.
- The "Smooth by kriging" button will not be available both current and baseline model are specified.
- if you see the message "Smoothing not processed, check if all required parameters are set", may mean that some of the required parameters are not set or that there is no selection box.

Figure Smooth transformation shows the option window.

6.9.7.3 Details and advanced parameters

Advanced parameters are related to (1) the "strength" of the smoothing allowed on both parts that are smoothed and on the entities selected for targets ("nugget" parameter), and (2) the strategy to deal with the large amount of control points typically used in the process (computing cost issues). The parameters and options are described below. Experimenting with a small model or region can be helpful to better understand their effects.

Another three lines of controls serve to set up nuggets of skin control points and bone control points.

- You can either check the "Compute nuggets of skin control points based on distance" to set the nuggets automatically (nugget will be equal to a negation of square of the shortest distance to a bone), or set it manually (if you uncheck the box) in the text box below. Note that the Nugget must be a negative number. The skin control points are nodes of all skin entities from among the selected targets. By default, the nugget is set to be computed according to the distance to bones. Therefore, the skin nodes will be allowed to move a bit in order to improve the global result of the interpolation. However, if you pre-processed the skin somehow and you want the skin nodes to be preserved, set the nugget to zero.

- The bone control points are the surface nodes of all selected bony target entities. By default it is zero, so the kriging will not change the coordinates of bone nodes at all. Setting this to a non-zero value can be useful if you are smoothing the results of some scaling process that changed the skeleton.

- Progressive kriging: this checkbox controls how are treated nodes that belong to more than one box, you can read the explanation directly in the GUI. In most cases, this option will not have a large impact on the results, but feel free to experiment with it.

Remarks

Box splitting options (available in the global parameter menu): the computation cost of the kriging grows rapidly with the number of control points, which makes the "divide and conquer" paradigm fit to apply here. By splitting a box that is too large, the computation time can be greatly enhanced without sacrificing much in terms of quality of the interpolation. This box splitting is automatic and should not be confused with the selection boxes. Parameters controlling this splitting are in the global parameter menu. The first two input text boxes allow you to specify a number of control points and total points allowed in a box. If the box has more, it

will be recursively split into two until those thresholds are satisfied.
The slider below allows to set some overlap between those split boxes. This was initially used to ensure that there are no large discrepancies on their boundary, i.e. so that the interpolation still behaves more or less as if it was only a single box. However, this can significantly increase the computation time. Therefore, a different method for preserving the smoothness across boxes was introduced in version 0.99 of Piper: a small, fixed number (although relative to the size of the box) of control points near the boundary of the box from surrounding boxes are added to each box. For now, it is still possible to use the overlap parameter, but its default value is now zero and it is expected that it will be removed in future version, once more thorough testing is done and it is confirmed that the continuity is sufficiently preserved using only the new method.

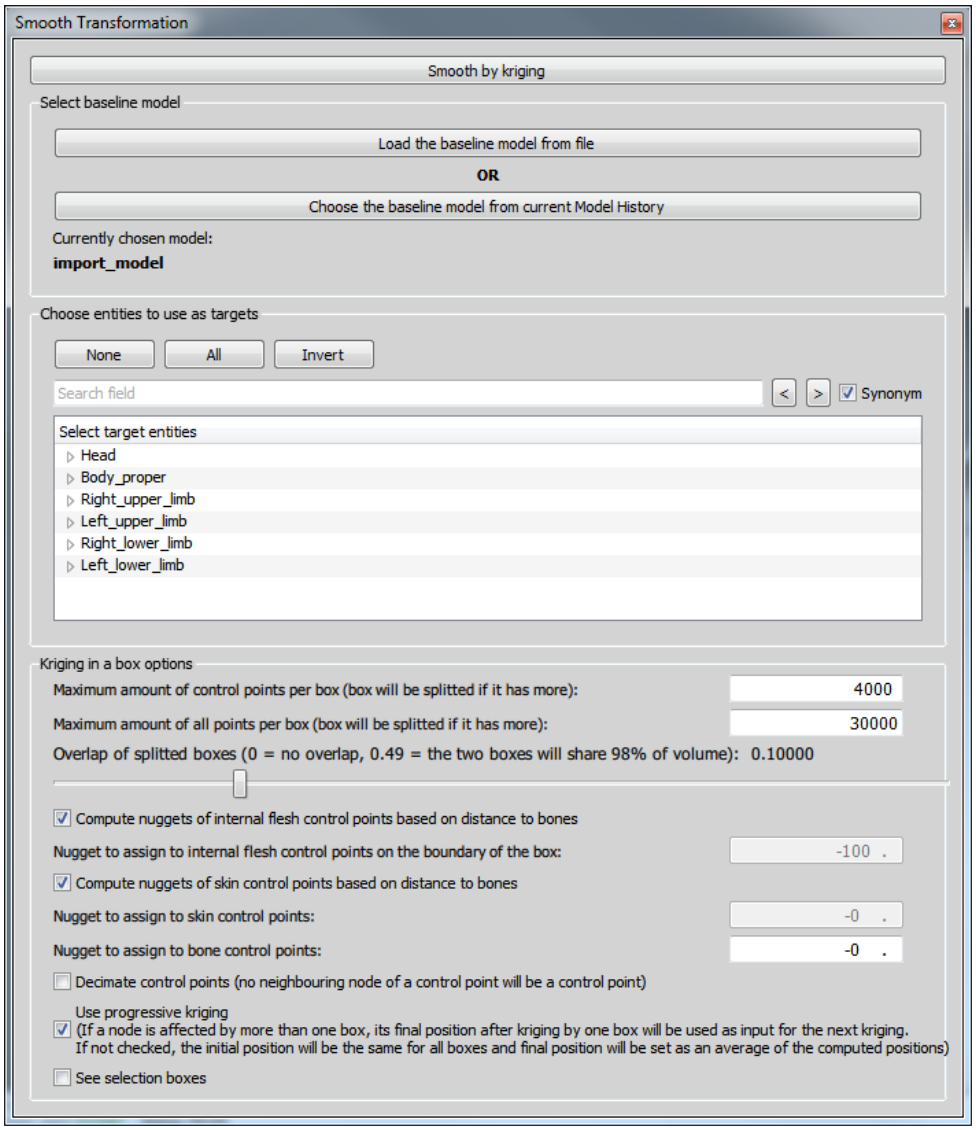


Figure 39: Kriging in a box GUI

6.10 Kriging Module

6.10.1 Overview

The kriging module allows to deform the PIPER model or the full FE based to a set of source and associated target control points. More generally Kriging is an interpolation method commonly used to deform geometrical models (along with Radial Basis Function, RBF and Moving Least Square MLS, Jolivet et al., Stapp 2015). Kriging

has a lots of similarities with RBF (cost, ability to relax the transformation, etc) and with specific parameters, the formulations are equivalent.

The deformation proposed is based on Dual Krigging formulation with generalized covariance (3D Euclidian distance) as detailed by Trochu (F. Trochu: “A contouring program based on Dual Kriging interpolation”, Engineering With Computers, Vol. 9, pp. 160-177, 1993).

The implementation that is proposed aims to help a number of issues related to Kriging:

- size of the problem: Kriging typically becomes challenging in terms of computing costs when the number of control points becomes large (e.g. over 20000, size of the square matrix to invert). The implementation uses an automatic moving box strategy allowing to use arbitrary numbers of control points (e.g. up to 300000 tested) without noticeable effect on the results.
- cost: several approaches are used to reduce the computing cost. First, the process is parallelized then a sampling strategy is used to select and remove control points which affect the least the interpolation (hereby reducing the computing time without affecting widely the result). The downsampling parameters are accessible by the user.
- distance formulation: the interpolation can use a surface distance instead of an euclidian distance to drive the scaling. This is especially relevant for regions that are close in the euclidian space (e.g. arm and torso) but far away on the skin surface (and anatomically independent). It is based on an approximation of the geodesic distance on a surface and can be used to improve the realism of the skin. The formulation used by Piper is the Biharmonic Distance (Yaron Lipman, Raif Rustamov, Thomas Funkhouser: "Biharmonic distance", ACM Transactions on Graphics 29, issue 3, 2010). Piper uses the "approximate distance" formulation, which uses only a small subset of eigenvalues of the linear system that is being solved to compute the distance. The "Topological distance precision" parameter in Module parameters can be modified to change this number of eigenvalues - it is computed as [the parameter value] / [number of nodes].
- to scale independently the skin, the bones, and then use their surfaces to scale the whole model (called dense scaling in the application).
- change as estimate the relaxation parameter (called nugget) globally or depending on the variability of the interpolation field in a zone and the relative motion of the bones and skin.

Combined, this provides the user a flexibility and adaptability not typically found in existing "morphing" approaches (publications are in preparation). The same approaches are also used to smooth the transformation field.

6.10.2 Description

- **Parameters:** None
- **Metadata:**
  - Required: None
  - Optional: Control points defined on HBM
- **Target** Input: target of type ControlPoint
- **Target** Output: target of type ControlPoint
- **Outputs:**
  - a deformed model according to control point and associated target points using Kriging.

Next figure illustrates the Kriging Module GUI, composed of three set of tools:

1. Control points tools to define Control Points and Target Points
2. Deformation Tools
3. Display

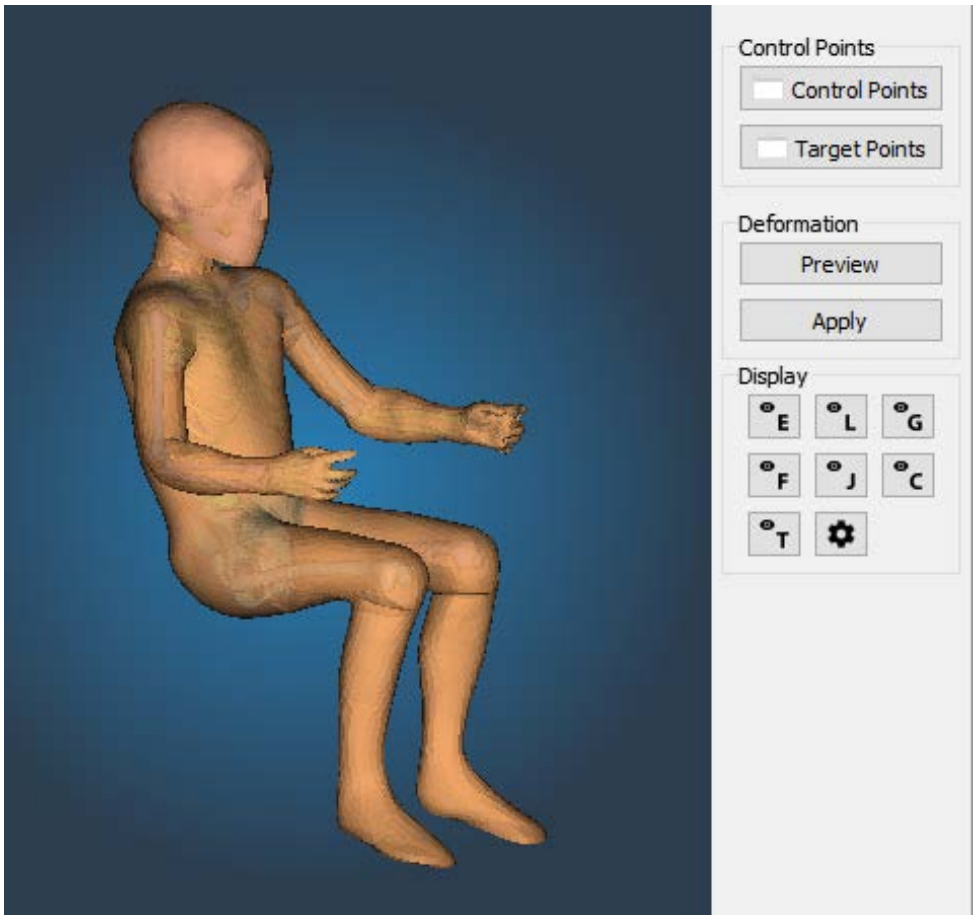


Figure 40: Kriging Module GUI Overview

6.10.3 Control Points

Select set of Control Points to be used for the Kriging deformation. The name and the number of control points of the control points set is displayed.

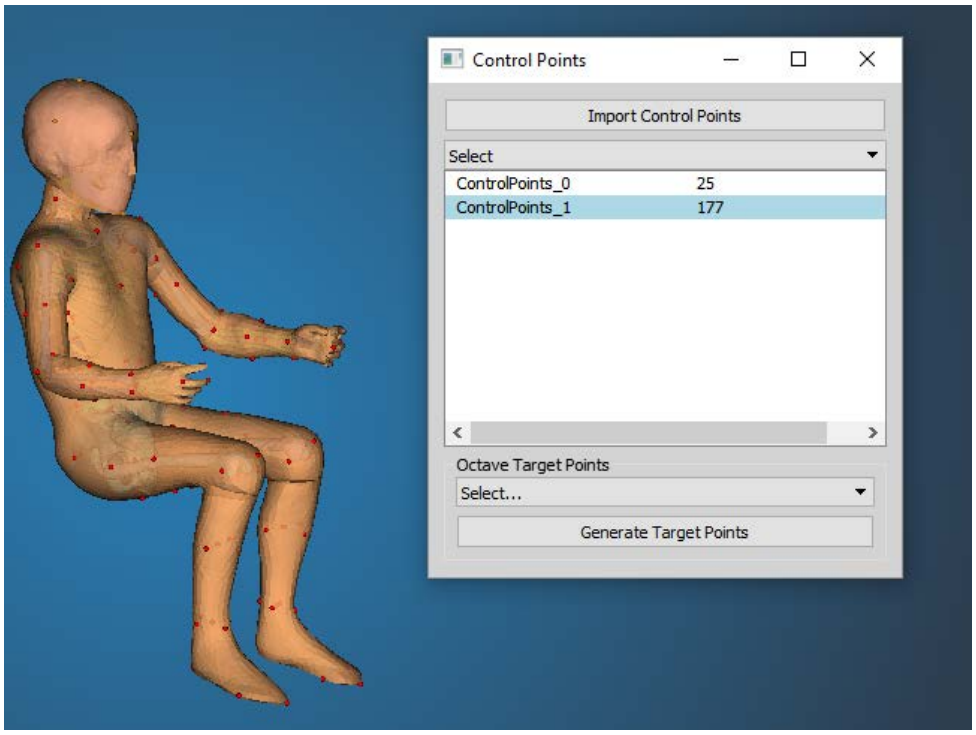


Figure 41: Control Points module tool

A set of control point coordinates can be imported from a text file:

Example of control points file:

```
-163.87512 -188.34157 -497.91904
-163.87512 188.34157 -497.91904
-249.919373518425 0 -231.471675832181
-48.5283182585665 0 -325.381867183739
-168.1071016685 -170.31156382101 -269.621364726814
-168.1071016685 170.31156382101 -269.621364726814
....
```

To generate target points associated with the current set of control points, a octave cript an be defined (TODO: describe octave API for script)

6.10.4 Target Points

Select target points existing in the project and define the association with a set of control points.

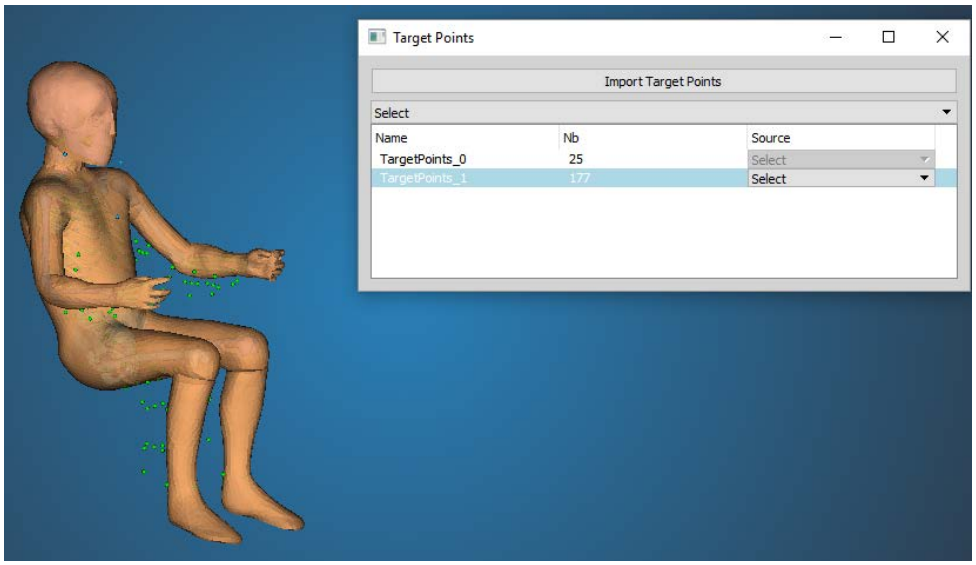


Figure 42: Target Points module tool

A set of target point coordinates can be imported from a text file:

Example of target points file:

```
-163.87512 -188.34157 -497.91904
-163.87512 188.34157 -497.91904
-249.919373518425 0 -231.471675832181
-48.5283182585665 0 -325.381867183739
-168.1071016685 -170.31156382101 -269.621364726814
-168.1071016685 170.31156382101 -269.621364726814
....
```

For each target point set, select the control point set to define the deformation. The number of control and target points should match, otherwise an error occurs during deformation. Control points should not be duplicated - duplicates will be automatically removed.

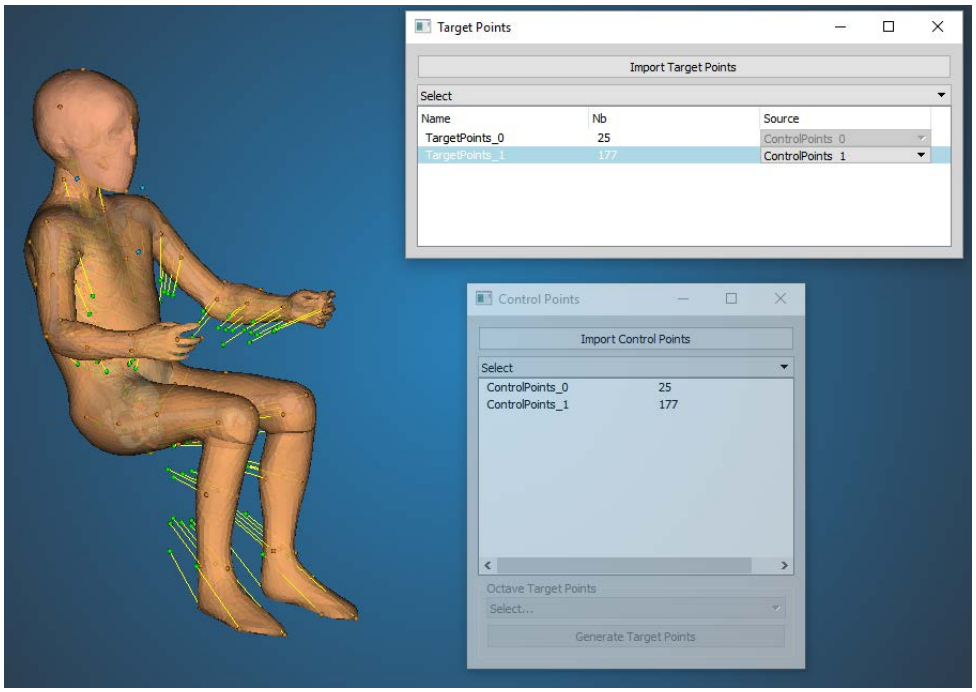


Figure 43: Associate Target Points with Control Points

6.10.5 Deformation Tools

To apply the deformation, open the "Perform Deformation" tool window. There are two modes available: kriging with or without intermediate targets. On the very bottom of the tool window there are two "Apply - ..." buttons, one for each of the modes. Clicking them will deform the loaded HBM model and create a new history node for it (see Model History menu).

6.10.5.1 Kriging with intermediate targets

Kriging with intermediate targets does not use the input control points to directly deform the entire model, but rather to create two intermediate targets - one for skin and one for bones. This means that only nodes of entites marked as skin in the anatomy database (see "entity" element) will be deformed separately and so will nodes marked as bone. Results of this deformation are temporarily stored and used as the intermediate targets, i.e. all nodes of bones and skin are used as control points, with targets defined by the input set of control points.
There are numerous options that allow for different workflows when the intermediate targets are used. The following list describes them as they are placed in the tool window, top to bottom:

- **Skin target options:**
  - Topology-aware distance: This allows toggling the distance formulation as discussed in the Overview. Computing the topology-aware distance will take usually around 10-30 seconds, but in some cases may significantly improve the result and also, once it is related only to the source, so until it changes (model changes, control points are added or removed), it does not have to be recomputed again for different targets.
  - Used for skin threshold: Only nodes that have the "use for skin" (see Skin/bone association) parameter equal to or higher than this threshold will be used to create the skin intermediate target.
  - Preview: A preview of the skin after kriging can be visualized as a semi-transparent red surface (see the Figure below). Note that neither selection, blanking or quality computation are possible on the preview surface - it is intended only to provide a quick overview. Changing any parameters relevant to the deformation, such as position of target points, will automatically disable the visualization. Upon ticking the preview box again, it will recompute it using the updated input.

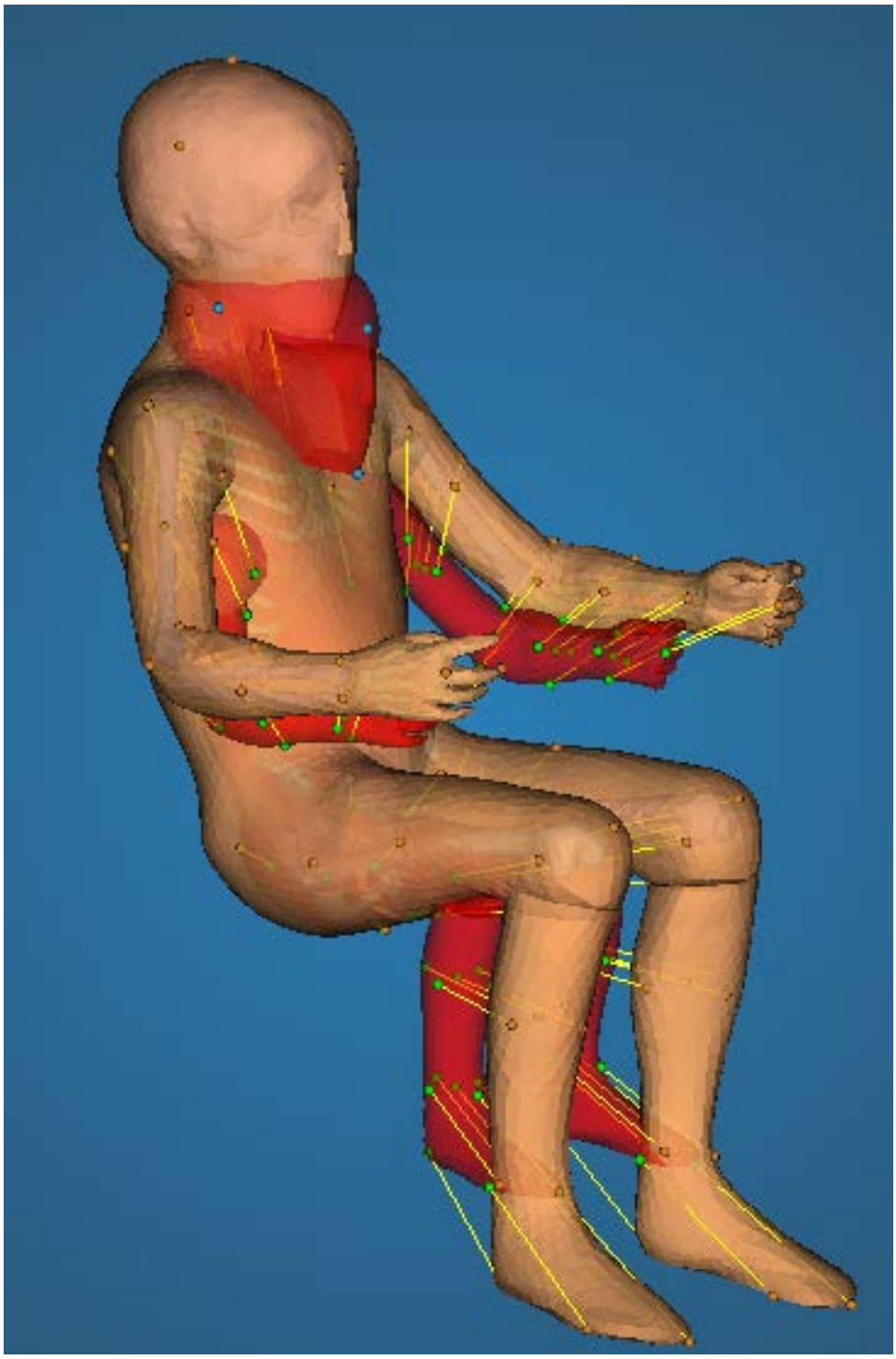


Figure 44: Deformation Preview

- **Bones target options:**
  - Used for bones threshold: Only nodes that have the "use for bones" (see Skin/bone association) parameter equal to or higher than this threshold will be used to create the bones intermediate target.
  - Fix bones: Enabling this option will prevent any nodes that belong to bone entities to move during the deformation. Note that they will still be used as control points in order to constrain the movement of the surrounding flesh, but their position will not change, even if some nugget is assigned to them.
  - Preview: Bone preview is visualized as a semi-transparent yellow surface. It behaves the same way as the skin preview (see description above).

- **Decimation:** In order to decrease the computation time of the deformation, an automated algorithm for control points decimation is employed. For each control point on the intermediate target (i.e. each node of the intermediate target), a deviation of that nodes source to target displacement from the distance-weighted average of displacement of all surrounding nodes is computed. If that deviation is lower than a specified threshold, the node is considered to be insignificant - it's surroundings describe the same motion, so it is not needed - and will not be used for the kriging. The following list describes the options for this tool:
  - Influence radius: the aforementioned surroundings is a sphere around the node that is being processed. This parameter describes how large that sphere is. Note that even though there is no button to explicitly turn off the automated decimation, setting the radius to 0 will effectively do it.
  - Maximum deviation: this is the deviation threshold. Here deviation means the distance between the actual target and the target that would be obtained if the processed point used the average displacement.
  - Decimate control points: pressing this button will do the decimation. Note that this is useful only for the visualization - upon pressing the "Apply" deformation button, the decimation will be done anyway if it wasn't done already.
  - Visualize control points on intermediate targets: after decimation, the remaining control points can be visualized as blue spheres on the preview of intermediate targets (the preview has to be turned on).
- **Automatic nugget computation:** There are currently three ways to set the Nugget of control points: define it in the input files (see "Control Points" element), use the default nugget, or use the automatically computed nugget. The automatic computation works on the same principle as the decimation algorithm described above - if all points in the vicinity of a given control point "move the same way", it is not needed. But in the other case, where points in close proximity have very different displacements, high nugget will probably be needed to avoid penetrations and other artifacts. The nugget is therefore set based on the deviation from the average displacement in a given surrounding of each point. Several parameters, described in the list below, are introduced in order to scale this result.
  One difference between the look-up of surrounding between the decimation and nugget algorithms is that the decimation algorithms searches for points that are close to the other point in terms of their source positions, while the nugget uses the target positions instead. The assumption that led to this arrangement is that the most common type of artifact that can be removed by an appropriate nugget is penetration of elements. Therefore, it is more important to track points that are close to each other after the deformation which can signify a potential penetration. Of course, if the penetration is larger than the influence radius, the penetration will not be discovered.
  Note that this feature has not been tested thoroughly yet and probably will work as intended only in some cases.
  - Influence radius: raidus of a sphere around the processed point to consider as surroundings.
  - Skin scale: The computed nugget will be multiplied by this number if it belongs to a skin node.
  - Bones scale: The computed nugget will be multiplied by this number if it belongs to a bone node.
  - Bone-skin distribution ratio: this parameter allows to enforce the constraints defined either on skin, or bone. The average displacement is computed separately for nodes in the surroundings that belong to bones (DEVb) and separately for those that belong to skin (DEVs). The nugget is then computed using this parameter (DISTR) as: $Nugget = DISTR * DEVb + (1 - DISTR) * DEVs$. For example, setting it to 0 means only deviation from skin will be considered in setting the nugget, effectively saying that displacement of bone is not relevant and should be suppressed, while displacement of skin should be respected. If the processed point is a skin point itself, it will therefore be assigned no nugget, because it will have similar displacement as the average of skin nodes displacement, whereas bone node will have a different displacement and therefore will be assigned a high nugget.
  - Compute nuggets: this will perform the computation. As with the "Decimate control points" button, this is relevant only for visualization - the computation will be done before the final Kriging deformation whether this button was pressed or not. If you don't want to use this option, check the "use default nugget".
  - Visualize nuggets: once the automatic nugget is computed, it can be visualized on the preview surfaces (they must be enabled). The nugget is visualized on a continuous scale from 0 to the highest (negative) nugget value by transition of blue to red color, respectively. As of current Piper version (1.0.0), there is no visualization of the numerical values of the scale, only the highest nugget is written in the console. If decimation was performed, points that have been decimated obviously have no nugget, so they are assigned white color.

- **Intermediate target selection:** In some workflow, it will be better to use only one of the intermediate targets to transform the model. Either one can be disabled using the check boxes on the bottom of the tool window.

6.10.5.1.1 Skin/bone association

These two parameters express how much should a given control point affect bone or skin entities. It is a number between 0 and 1, where 0 means the point does not affect skin/bone at all and 1 is interpreted as "is an actual point of point/skin". This enables to have plausible deformation for both intermediate targets without introducing artifacts coming from control points that are too far away from the points they deform (e.g. control points for spine affecting the skin etc.).

6.10.5.1.2 Domain decomposition of HBM

Although some artifacts of the deformation can be mitigated by using the aforementioned tools, there can still be issues in some scenarios. Namely, when several significantly different constraints (control points' targets) are used, the resulting deformation field might be different from what is needed: Kriging will always try to create a single, smooth interpolation based on the provided constraints. Imagine a simple 2D example: let's assume two control point sets, points A and left of them points B, relatively close to each other. Points A move right, away from points B while points B remain stationary, i.e. their target position is equal to their source position. If a single smooth interpolation is defined based on these control points, then on the right side of points B, this interpolation will be represented by a displacement proportional to the displacement by which points B are moved. But since points B do not move, i.e. their displacement is zero, the deformation field on the left side of points B will be a displacement in the opposite direction. This will often not be intended, in fact, the non-moving constraint is probably intended as "everything behind it should also not move" - but that will not be the case in a single smooth field is imposed on the area.
To mitigate this drawback, it is possible to decompose the model into "domains", i.e. areas that are expected to move in a uniform way so that the assumption that the deformation can be represented by a single smooth deformation field is more or less upheld. For most cases, it will be beneficial to decompose the model into parts that are individually, i.e. at each major joint. To minimize the effort needed to do this decomposition, the only metadata that needs to be defined is a set of nodes that create a closed polyline on the skin of the model. This polyline represents the boundary between two domains. Any number of such boundaries can be defined. Piper will subdivide the model into domains and perform kriging on each domain separately and in the end assemble the results back as a single HBM.
To define the boundary point sets ("rings"), use Named Metadata following these rules:

- The name can be an arbitrary string ending with "_decomposition". For example, "Left_arm_decomposition", "Left_knee_decomposition" etc.
- Each ring must be made of nodes that are on the skin, i.e. are part of entities that are defined as skin in the anatomy database (see "entity" element)
- Each ring must form a closed polyline, i.e. for a ring with N nodes, there exists exactly N edges and each node has exactly two neighbors among the other points on the ring to which it is connected by those edges. However, the nodes do not have to sorted in any particular order in the input file.
- Each edge in the ring must be manifold and non-boundary, i.e. exactly two elements must share that edge.

Rings that do not follow the aforementioned rules will simply be ignored (a warning will be written in the Piper console).
This option can be turned off with a check box on the bottom of the tool window. If it is turned on and the required metadata are not present in the model, the option will be ignored and the transformation will be perfomed as if the option was turned off.

6.10.5.2 Direct Kriging

The "direct" kriging does not use any preprocessing of the control points - the loaded control points are directly used to define the deformation. In case there is only a small amount of control points (relative to the number of the model's nodes), it will likely lead to some artifacts. For this reason, direct kriging should be used only for a dense network of control points or in conjuction with some model specific knowledge. For other cases, the Kriging with intermediate targets should be preffered.

Note that even though the intermediate targets will not be used for Direct Kriging, one can still use the previews of intermediate targets to get a fast assessment of the effects of the deformation.

6.10.6 Nugget

A nugget is a "relaxation weight" one can assign to a kriging control point. I represents the uncertainty (variance) of the target position for a given point. A non-zero nugget will allow the control point to violate the target position by at most the nugget value distance in order to make the global transformation more smooth. It is useful in situations where it is not necessarry to follow the target perfectly.
You can specify a global nugget in the Module parameters menu. This value will be used for all control points for any PIPER tool that utilizes kriging (e.g. the Kriging module or the scaling tools in Anthropometry module), unless the nugget is explicitly specified to some other value, as it is for example in the Transformation smoothing tool or by the control points definition when imported from a file. The default value is 0, meaning the target positions of control points will be maintained perfectly.

6.10.7 Display

TODO?

6.11 Contour Deformation Module

This module implements a Contour based approach to position the Human Body Model.

The GBP method of mesh modification is based on the use of contours that envelop the body. The module has primarily two aspects: (1) positioning and (2) personalization.

The method of mesh modification is based on the use of contours that envelop the body. The contour based deformation (personalization as well as repositioning) method has 4 steps

1. Contour Generation
2. Delaunay Mapping
3. Contour Transformation
4. Remapping.

In the first step, we load the contourCL.xml file from the positioning window. The cotnour Control Line(contourCL) is a yellow line structure made up by joining the anatomical landmarks on the Human Body Model. With the help of this contourCL contours are generated all around the GHBMC model. These contours and some typical key points (for instance the nodes of the bones in some cases) are used to generate Delaunay Tetrahedrons which cover whole model.
All the mesh nodes of the model are then mapped to these Delaunay Tetrahedrons using volume coordinates. Third step consist of transformation of these contours as per the desired input (for instance personalization / repositioning parameters). It is expected that the key points (for instance the bone nodes) are also transformed using the same parameters. Figure 1 shows a set of contours for the GHBMC model.
At the end of third step final positions for all contour points as well as all the key points will be available in the final position. Using these transformed contours and key points, the mapped mesh nodes are remapped and their new positions are determined. This completes the deformation process.
Target Input : For target based personalization and positioning Contour Deformation module handles the following targets:

- For positioning it handles the Landmark Targets
- For personalization it can handle GEBOD target and ANSUR targets generated in the Anthropometric Module.

6.11.1 Metadata

The required metadata for positioning and personalization are :

- Required : contourCL, Landmarks needed to define contourCL, Skin Entity, Flesh GenericMetadata, Contours
- contourCL : contour control line is formed from the DMA landmarks. It contains the body region and joint information.
- Contours : Contours are typically closed curves like circle, ellipse, polyline or cubic splines. The contour modification routines shall modify them accordingly.
- Landmarks (defined to create contourCL)
- The bones entities (as defined in Appendix: List of entities)
- Genericmetadata

6.11.2 UI for Contour Deformation Module

6.11.2.1 POSITIONING

Allows the user to carry out the Positioning process. The UI of the POSITIONING Window consists of the following features :

- Load ContourCL file : It loads the contourCL metadata. It is required for generation of contours around the body and the information required to reposition the model at a specified joint by giving a specific angle wih respect to the current position of the model. In current piper application contourCL is the yellow highlighted line segment. It is basically made up by joining the anatomical landmarks of the human body.
- Load Contours (for hip positioning) File icon : User can externally load the contours. Curently It is used when one wants to the the repositioning at hip joint. User needs to load the contours from following path : {model↩ _path}/GHBM/metadata/contours_for_hip/Contour_Input.txt . Deform Contours button will remain disabled for Hip Joint unless user loads the required contours from the specified path.
- Module Opacity : It controls the opacity of human body model.
- Contour CL opacity : It controls the opacity of contourCL and generated contours.
- Repositioning Parameters : This section takes input from the user about the joint at which he wants to do the positioning, the type of movement, the side(left/right) and the angle(with respect to the current position).
- Select joint : Select the joint from a list of joints- Knee, Ankle, Hip, Neck, Elbow or Wrist
- Select side : Left or Right Side
- Select type : Type of movement- Flexion/Extension, Inversion/Aversion, Lateral Rotation or Lateral Flexion etc.
- Angle : Angle must be given with respect to the current position. eg. A value of 10 degree at LEFT KNEE joint shall do a flexion of knee by an angle of 10 degree.
- Generate Contours : Contours are typically closed non-intersecting curves like circle, ellipse or a polyline enveloping the HBM. In some cases they can also be open cubic splines. The contour modification routines shall modify them accordingly. Typically, the contours should not be on the skin but slightly bigger and away from surface of mesh. Generate contour will create these contours.
- Deform Contours : Contours will be deformed by the input angle at the selected joint.
- Reposition : The model will be repositioned.
- Use Landmark Switch : If the positioning is to be done using the landmark targets import the target file from the file menu and click on this switch to start the positioning through landmark targets

- Process Landmark Targets : Click on this button to compute the input required to do the positioning.
- Continue : Click on this button to continue with the process of landmark target positioning till a prompt of positioning complete comes up

6.11.2.2 PERSONALIZATION

Allows the user to personalize the model using a number of parameters.
The Personalization window UI :

- Load Personalization Contours : User needs to load the contours for personalization. The path for the Personalization_Contours is "{model}\GHBM\metadata\contour_input_for_personalization"
- Load Body Data : Load BodyData.txt file from the path : "{model}\GHBM\metadata\BodyData" to enable the personalization process
- Load Anthropometric Target File : Load the target file generated from the anthropometric module, it contain the values corresponding to ANSUR body dimensions. Load this file when you need to do the personalization using the target ANSUR body dimensions.
- Options : At present the user can populate the 35 GEBOD dimension values by selecting among 5th,50th,75th or 95th percentile or by selecting the generated GEBOD or ANSUR target dimensions. GEBOD & ANSUR target dimensions can be generated through Anthorpometric Module.
- Personalize : Personalize button will start the personalization for the full GHBMC model for the given 35 parameter values

GEBOD Body Dimension Descriptions:

- 1. WEIGHT: Subject stands on scales (nude or wearing lightweight undergarments) with feet parallel and weight distributed equally on both feet.
- 2. STANDING HEIGHT: Subject stands erect, head in the Frankfort plane, heels together, and weight distributed equally on both feet. With the arm of the anthropometer firmly 64 MADYMO Utilities Manual Release 7.5 touching the scalp, measure the vertical distance from the standing surface to the top of the head.
- 3. SHOULDER HEIGHT: Subject stands erect looking straight ahead, heels together, and weight distributed equally on both feet. With an anthropometer, measure the vertical distance from the standing surface to the right acromial landmark.
- 4. ARMPIT HEIGHT: Measurement derived by subtracting Scye Circumference divided by π. from Shoulder Height (see above). Scye Circumference: Subject stands erect looking straight ahead. The right arm is abducted sufficiently to allow placement of a tape into the axilla. With a tape passing through the axilla, over the anterior and posterior vertical scye landmarks and over the right acromial landmark, measure the circumference of the scye. The axillary tissue is not compressed.
- 5. WAIST HEIGHT: Subject stands erect, his head in the Frankfort plane.Using an anthropometer, measure the distance from the standing surface to the omphalion landmark. The subject must not pull in his stomach.
- 6. SEATED HEIGHT: Subject sits erect, head in the Frankfort plane, upper arms hanging relaxed, forearms and hands extended forward horizontally. With the anthropometer arm firmly touching the scalp, measure the vertical distance from the sitting surface to the top of the head.
- 7. HEAD LENGTH: Subject sits. With a spreading calliper, measure in the midsagittal plane the maximum length of the head between the glabella landmark and the occiput.
- 8. HEAD BREADTH: Subject sits. With a spreading calliper measure the maximum horizontal breadth of the head above the level of the ears.
- 9. HEAD TO CHIN HEIGHT: Subject stands under the headboard looking straight ahead. The headboard is adjusted so that its vertical and horizontal planes are in firm contact with the back and the top of the head. Positioning the head in the Frankfort plane and using the special gauge, measure the vertical distance from the horizontal plane to the menton landmark.

- 10. NECK CIRCUMFERENCE: Subject sits erect, head in the Frankfort plane. A piece of dental tape is placed around the neck, passing over all four neck landmarks. The measurer marks off with her thumbnail a length of tape corresponding to the subject's neck circumference, and then measures this tape segment with a standard tape.

- 11. SHOULDER BREADTH: Subject sits erect looking straight ahead, upper arms hanging relaxed, forearms and hands extended forward horizontally. With a beam calliper, measure the distance between the acromial landmarks.

- 12. CHEST DEPTH: Subject stands erect looking straight ahead, heels together, and weight distributed equally on both feet. With a beam calliper, measure the horizontal depth of the trunk at the level of the bustpoint landmarks. The reading is made at the point of maximum quiet inspiration. 65 Release 7.5 MAD↩ YMO Utilities Manual

- 13. CHEST BREADTH: Subject stands erect looking straight ahead with arms slightly abducted. With a beam calliper, measure the horizontal distance across the trunk at the level of the bustpoint landmarks.

- 14. WAIST DEPTH: Subject stands erect looking straight ahead, arms at sides heels together, and weight distributed equally on both feet. With a beam calliper, measure the horizontal depth of the trunk at the level of the waist landmarks. The reading is made at the point of maximum quiet inspiration. The subject must not pull in her stomach.

- 15. WAIST BREADTH: Subject stands erect looking straight ahead with arms slightly abducted. With a beam calliper, measure the horizontal breadth across the trunk at the level of the waist landmarks.

- 16. BUTTOCK DEPTH: Subject stands erect, heels together and weight distributed equally on both feet. With a beam calliper, measure the horizontal depth of the trunk at the level of the buttock landmark.

- 17. HIP BREADTH, STANDING: Subject stands erect, heels together and weight distributed equally on both feet. With a beam calliper, measure the maximum horizontal breadth of the hips.

- 18. SHOULDER TO ELBOW LENGTH: Subject stands erect looking straight ahead and with arms relaxed. With a beam calliper held parallel to the long axis of the right upper arm, measure the distance from the acromial landmark to the radiale landmark.

- 19. FOREARM-HAND LENGTH: Measurement derived by summing Radiale- Stylion Length and Hand Length. Radiale-Stylion Length: Subject stands erect with arms relaxed. With a beam calliper held parallel to the long axis of the right forearm, measure the distance from the radiale landmark to the stylion landmark. Hand Length: Subject sits, right forearm and hand raised with palm up. The fingers are together and straight but not hyper-extended. With the bar of a sliding calliper parallel to the long axis of the hand, measure the distance from the wrist landmark to the dactylion.

- 20. BICEPS CIRCUMFERENCE: Subject stands with right arm slightly abducted. With a tape held in a plane perpendicular to the long axis of the upper arm, measure the circumference of the arm at the level of the biceps landmark.

- 21. ELBOW CIRCUMFERENCE: Subject stands, right upper arm raised so that its long axis is horizontal, elbow flexed 90 degrees, fist tightly clenched and biceps strongly contracted. With a tape passing over the tip and through the crotch of the elbow, measure the circumference of the elbow.

- 22. FOREARM CIRCUMFERENCE: Subject stands erect with right arm slightly abducted and hand relaxed. With a tape held in a plane perpendicular to the long axis of the forearm, measure the circumference of the arm at the level of the forearm landmark.

- 23. WRIST CIRCUMFERENCE: Subject stands with right arm slightly abducted. With a tape held in a plane perpendicular to the long axis of the forearm and hand, measure the circumference of the wrist at the level of the stylion landmark. 66 MADYMO Utilities Manual Release 7.5

- 24. KNEE HEIGHT, SEATED: Subject sits with his feet resting on a surface adjusted so that the knees are bent at about right angles. Using an anthropometer, measure the vertical distance from the footrest surface to the suprapatella landmark on the right knee.

- 25. THIGH CIRCUMFERENCE: Subject stands erect, heels approximately 10 cm apart, and weight distributed equally on both feet. With a tape held in a plane perpendicular to the long axis of the right thigh measure the circumference of the thigh at the level of the lowest point on the gluteal furrow. Where the furrow is deeply indented, the measurement is made just distal to the furrow.
- 26. UPPER LEG CIRCUMFERENCE: Measurement derived by summing the Thigh Circumference (see above) and the Knee Circumference (see below) and dividing the sum by two to obtain the average.
- 27. KNEE CIRCUMFERENCE: Subject stands erect, heels approximately 10 cm apart, and weight distributed equally on both feet. With a tape held in a plane perpendicular to the long axis of the right leg, measure the circumference of the knee at the level of the mid patella landmark. The subject must not tense her knee during the measurement.
- 28. CALF CIRCUMFERENCE: Subject stands erect, heels approximately 10 cm apart, and weight distributed equally on both feet. With a tape held in a plane perpendicular to the long axis of the right lower leg, measure the circumference of the calf at the level of the calf landmark.
- 29. ANKLE CIRCUMFERENCE: Subject stands erect with weight distributed equally on both feet. With a tape held in a plane perpendicular to the long axis of the right lower leg, measure the circumference of the leg at the level of the ankle landmark.
- 30. ANKLE HEIGHT, OUTSIDE: Subject stands with weight distributed equally on both feet. With the special measuring block, measure the vertical distance from the standing surface to the lateral malleolus landmark on the right leg.
- 31. FOOT BREADTH: Subject stands erect, right foot in the measuring box, left foot on a board of equal height, and weight distributed equally. The right foot is positioned so that its long axis is parallel to the side of the box, the heel touches the rear of the box, and the medial metatarsal-phalangeal joint touches the widest part of the foot, measure on the scale of the box the breadth of the foot.
- 32. FOOT LENGTH: Subject stands erect, right foot in the measuring box, left foot on a board of equal height, and weight distributed equally. The right foot is positioned so that its long axis is parallel to the side of the box, the heel touches the rear of the box, and the medial metatarsal-phalangeal joint touches the side of the box. With the measuring block touching the tip of the most protruding toe, measure on the scale of the box the length of the foot.
- 33. HAND BREADTH: Subject sits with right hand resting on a table, palm up, fingers extended and together. The thumb is held away from the hand. Using the sliding caliper, measure the maximum breadth from Metacarpale II to Metacarpale V. 67 Release 7.5 MADYMO Utilities Manual
- 34. HAND LENGTH: Subject sits with his right hand resting flat on a table, palm up, fingers extended and together. With the bar of the sliding caliper parallel to the long axis of the hand, measure the distance from the wrist landmark to the tip of the finger.
- 35. HAND DEPTH: Subject.s right hand is held palm down with fingers extended and together, narrow profile towards the measurerer. Maintaining light pressure on the spreading caliper, measure the thickness of the hand at the metacarpalphalangeal joint of the third finger.

6.11.2.3 Display Tab

Allows the user to selectively display of entities, landmarks, generic metadatas, frames, joints, contours and targets.

6.11.3 Limitations

The current repositioning works for full GHBMC Model. So to proceed with the repositioniong one has to import the FULL GHBMC.

- For repositioniong at pelvis joint one has to first load the Contour_Input.txt file from the "{model_path}/GH↩ BM/metadata/contours_for_hip/" folder. After loading the Contour_Input.txt file the contours will be generated for the model.
- Hip repositioning will then require the basic inputs like type of movement(flexion/extension,internal/external rotation or abduction/aduction), angle and left/right side.

6.11.4 POSITIONING Procedure

Steps to re-position the model using the contour deformation module :

- 1. Open Repositioning window by clicking on the "Positioning" button on the right panel
- 2. Load the contourCL from contourDeformation Module
- 3. Select the joint to be re-positioned and & type of movement intended :
  - 3a) Knee : type of repositioning [flexion/extension]
  - 3b) Ankle : type of repositioning [flexion/extension, inversion/aversion]
  - 3c) Neck : type of repositioning [flexion/extension, lateral rotation, lateral flexion/extension]
  - 3d) Elbow : type of repositioning [flexion/extension, pronation/supination]
  - 3e) Wrist : type of repositioning [flexion/extension]
  - 3f) Hip : type of repositioning [flexion/extension, abduction/adduction, internal/external rotation]
    - 3fa) Click on the button with the "File icon" and load the contour file present in piper_model/GHB↩M/metadata/contours_for_hip folder
- 4. Enter the angle by which you want to do the selected movement at the selected joint
- 5. Click on the "Generate contours" button to create the contours and wait for the contours to be generated
- 6. Click on the "Deform Contours" button to deform the contours.
- 7. Click on the "Reposition" button to re-position the model to the new position and wait for the re-positioning to be completed.

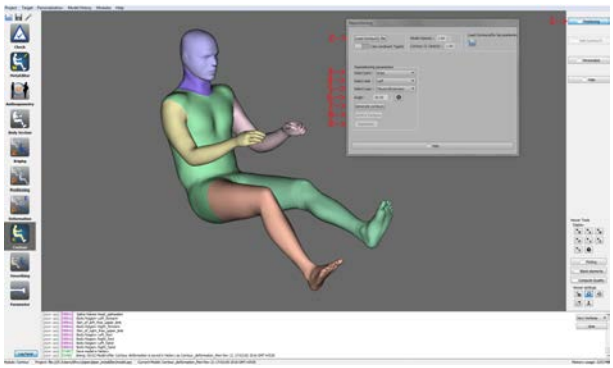


Figure 45: Positioining UI with stepwise description to do repositioning

Steps to re-position the model using the contour deformation module using landmark targets :

1. Import the target file using Target Menu
2. Go to Contour Deformation Module
3. Press Positioning button on the side bar. This will open a new window having controls for re-positioning the HBM
4. Press "Load ContourCL file" button to open the contourCL.xml. This step will cause yellow colored control lines to appear with the model display. These control lines provide a guiding structure for the generation of Contours around the HBM. It is also used in the Target Based Repositioning for traversal around the various Body Regions and to find the joint angles.
5. Press "Use Landmark Targets" switch to go into re-positioning based on Targets mode.

6. The target processing is an iterative process. The body regions are traversed along the contourCL structure. In order to start this iterative process follow the instructions below:

   (a) Press "Process Landmark Targets button". Please note as of now only Landmark Targets are supported.

   (b) The immutable fields Joint,Side, Type and Angle will get populated automatically. Please note, currently only flexion-extension is supported in this Mode.

   (c) Press "Generate Contours" button in order to generate the contours around the HBM.

   (d) Press "Deform Contours" button in order to deform the contours according to the computed Joint,Side,Type and Angle computed in the Step 6-b.

   (e) Press "Reposition" button. This will run the contour based repositioning algo to deform the model, based on information synthesized from Landmark Targets.

   (f) Press "Continue" button to move on to the next target for the remaining body regions, until all the body regions are processed.

   (g) If the computed angle is zero, stop.

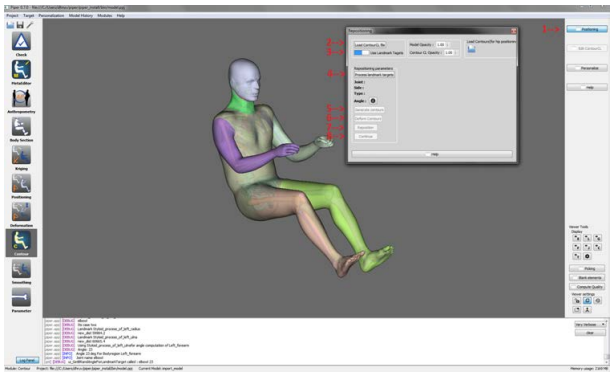


Figure 46: Positioining UI with stepwise description to do repositioning using Landmark Targets

Steps to personalize the model using the contour deformation module :

- 1. Open up PIPER and load the GHBMC model.
- 2. Open Personalization window by clicking on the "Personalization" button on the right panel.
- 3. Click on the "Load personalization contours" button and select the Personalization_Input.txt file to load the personalization contours.
- 4. Click on the "Load body data file" button and select the BodyData.txt file to load the personalization ratios.
- 5. Choose a percentile from the option tab or GEBOD body dimension targets or generate the ANSUR targets.
- 6. Click on the start button to personalize the model.

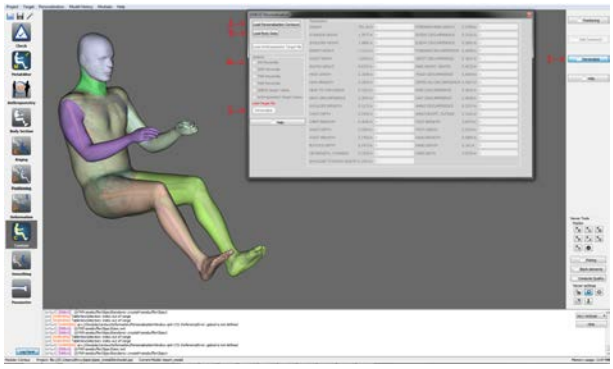


Figure 47: Personalization UI with steps to perform personalization

Steps to reposition the module in case of absence of a required entity (eg. SkinHeadNeck for positioning at cervical region) :

- 1. Open Repositioning window by clicking on the "Positioning" button on the right panel
- 2. Load the contourCL from contourDeformation Module from the path : "{model}\GHBM\metadata\MetaEditorWorkFlowEntityEditor" folder.
- 3. A pop window will propmpt about missing SkinHeadNeck entity.
- 4. Goto MetaEditor Module. Click on Complete contourCL button. A window will pop up showing the missing SkinHeadNeck metadata in the list.
- 5. Click on the missing metadata from the list and click on Create Object button.
- 6. Entity editor will pop up, now proceed as follows :
  - 6a) Select the entity picker from the picker options displayed in the entity editor window
  - 6b) Open up the Entity display window and selectively display Skin_Of_Head and Skin_Of_Neck entities
  - 6c) Click on shift and select the Skin_Of_Head and Skin_Of_Neck entities.
  - 6d) Click on add in the entity editor window to create the new entity SkinHeadNeck.
- 7. Now go to contour Deformation Module.
- 8. Open the Positioning window from the right tab.
- 9. Load the contourCL from contourDeformation Module from the path : "{model}\GHBM\metadata\MetaEditorWorkFlowEntityEditor" folder.
- 10. Select the following options in the Repositioning Parameters :
  - 10a) Select Neck in the Joint options
  - 10b) Select among the following movements : flexion/extension, lateral rotation, lateral flexion/extension
  - 10c) Enter the angle by which you want to do the selected movement at the selected joint
- 11. Click on the "Generate contours" button to create the contours and wait for the contours to be generated
- 12. Click on the "Deform Contours" button to deform the contours.
- 13. Click on the "Reposition" button to re-position the model to the new position and wait for the re-positioning to be completed.

Steps to reposition the module in case of absence of a required landmark(eg. footl_distal1 for positioning at ankle joint) :

- 1. Open Repositioning window by clicking on the "Positioning" button on the right panel

- 2. Load the contourCL from contourDeformation Module from the path : "{model}\GHBM\metadata\MetaEditorWorkFlowLandmarkEditor" folder.
- 3. A pop window will propmpt about missing controlpoint landmark footr_distal1.
- 4. Goto MetaEditor Module. Click on Complete contourCL button. A window will pop up showing the missing footr_distal1 metadata in the list.
- 5. Click on the missing metadata from the list and click on Create Object button.
- 6. Landmark editor will pop up, now proceed as follows :
  - 6a) Select the single node picker from the picker options displayed in the landmark editor window
  - 6b) Open up the Entity display window and selectively display Right_foot_Skeleton entity
  - 6c) Click on second distal phalange of the Right_foot_Skeleton entity
  - 6d) Click on add in the landmark editor window to create the new landmark footr_distal1.
- 7. Now go to contour Deformation Module.
- 8. Open the Positioning window from the right tab.
- 9. Load the contourCL from contourDeformation Module from the path : "{model}\GHBM\metadata\MetaEditorWorkFlowEntityEditor" folder.
- 10. Select the following options in the Repositioning Parameters :
  - 10a) Select Ankle in the Joint options
  - 10b) Select among the following movements : flexion/extension, inversion/aversion
  - 10c) Select Right in the Side options
  - 10d) Enter the angle by which you want to do the selected movement at the selected joint
- 11. Click on the "Generate contours" button to create the contours and wait for the contours to be generated
- 12. Click on the "Deform Contours" button to deform the contours.
- 13. Click on the "Reposition" button to re-position the model to the new position and wait for the re-positioning to be completed.

6.11.5 Contour Personalization

6.11.5.1 Overview

This module provides various tools for performing personalization on the Human Body Model (HBM) with the help of dynamic contourCL. The main approach of this module is to use contourCL as reference in order to perform personalization routine. The process can be performed in various ways such as Custom Personalization or Target Based Personalization.

6.11.5.2 Custom Personalization

This tools aims to provide the user to control the dimensions of the HBM in order to perform personalization. Two major tools being :

- 1. Circumference Scaling Tool
- 2. Length Scaling Tool

6.11.5.2.1 Custom Personalization: Circumference Tool

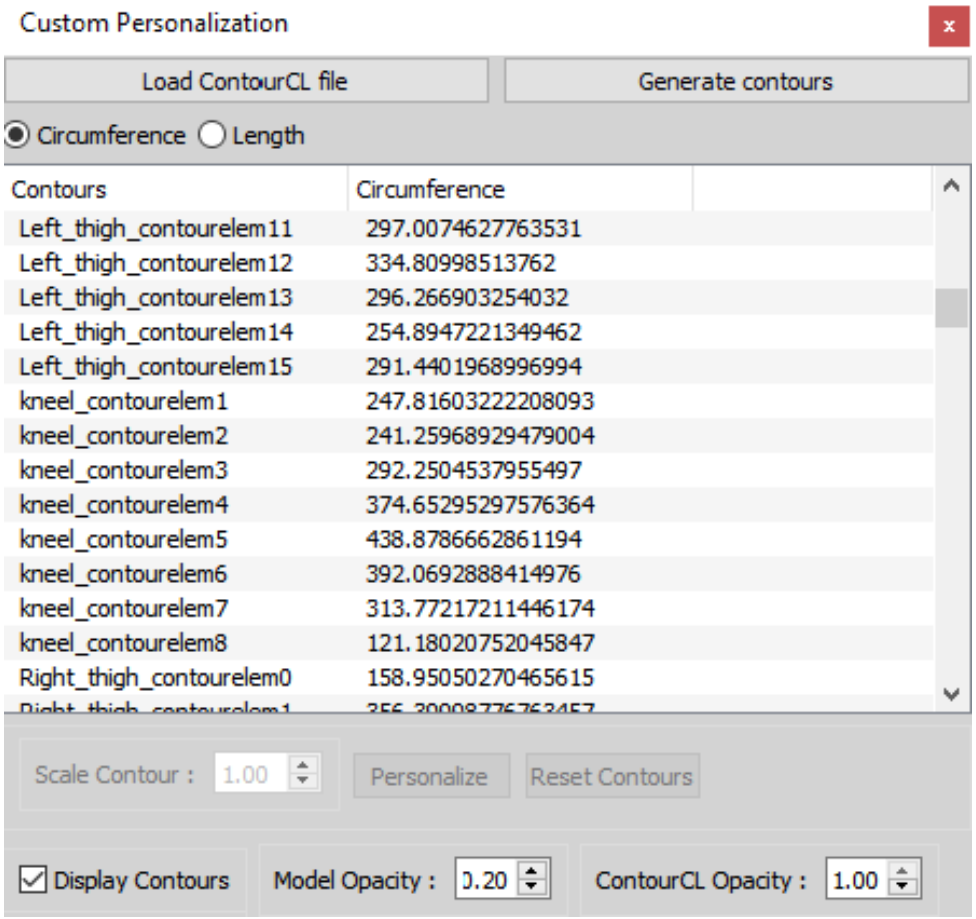


Figure 48: Custom Personalization Window: Circumference Tab

The above snapshot shows the Custom Personalization: Circumference Tool which is used to scale the circumference of the selected contour. The procedure for using this tools is as follows:

-1. Click on "Load ContourCL File" button and select the valid contourCL xml file. -2. Click on "Generate Contours" button in order to Genrate the Contours in the display. -3. Select the contour to be scaled from the provided table. -4. Change the value of "Scale Contour" spinbox and view the results in the display. -5. Click on "Reset Contours" incase of any error and perform the previous step again. -6. Click on "Personalize" button to perform personalization.

6.11.5.2.2 Custom Personalization: Length Tool

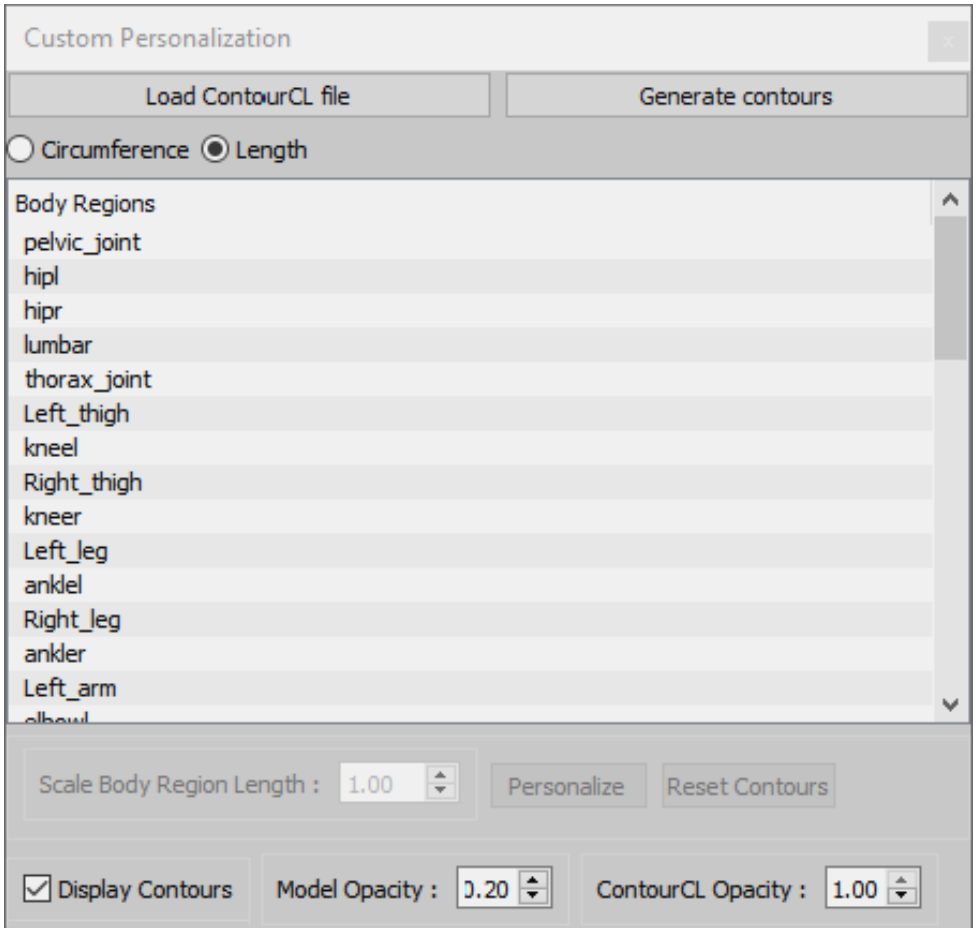


Figure 49: Custom Personalization Window: Length Tab

The above snapshot shows the Custom Personalization: Length Tool which is used to scale the length of the selected contour body region. The procedure for using this tools is as follows:

-1. Click on "Load ContourCL File" button and select the valid contourCL xml file. -2. Click on "Generate Contours" button in order to Genrate the Contours in the display. -3. Select the contour body region to be scaled from the provided table. -4. Change the value of "Scale Body Region Length" spinbox and view the results in the display. -5. Click on "Reset Contours" incase of any error and perform the previous step again. -6. Click on "Personalize" button to perform personalization.

6.11.5.3 Target Personalization

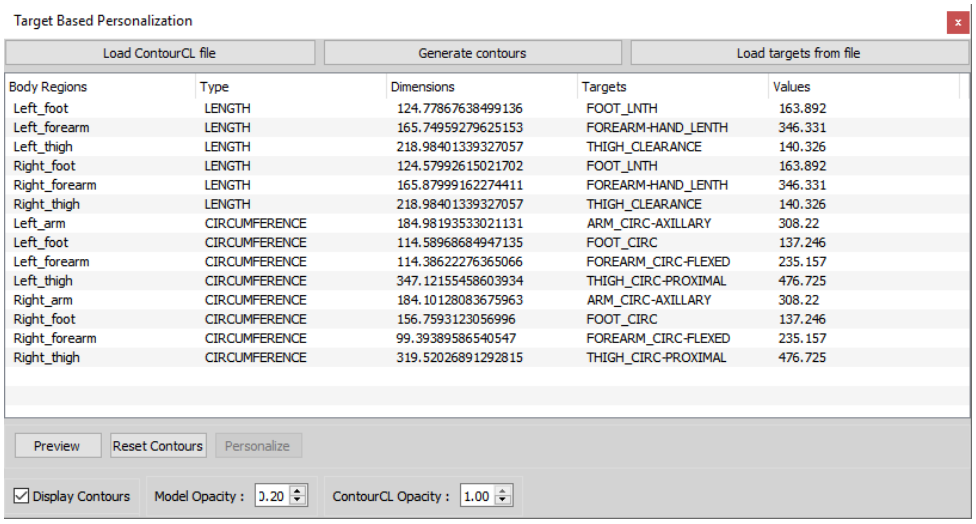


Figure 50: Target Personalization Window

The above snapshot shows the Target Personalization window. this tool provides the user the user to load the Anthrometric Targets generated from the Anthropo Module . The procedure for using this tools is as follows:

-1. Click on "Load ContourCL File" button and select the valid contourCL xml file. -2. Click on "Generate Contours" button in order to Genrate the Contours in the display. -3. Click on "Load Target File" button if the targets have not been loaded yet. The values of the targets will be updated in the table below that button. -4. Click on "Preview" to preview the modified contours. -5. Click on "Reset Contours" incase of any error and perform the previous step again. -6. Click on "Personalize" button to perform personalization.

6.12 Shaping Module

This module implements a physics simulation where the whole skeleton is a fixed affine frame and the soft tissue degrees of freedom are points sampled on the skin.

The skin gets elasticity from a triangular FEM to ensure a smooth transformation.

- **Metadata:** The skin and skeleton must be defined by Entity. The skin can be either defined in a single or multiple entites.
- **Input Targets**: currently the module does have support for loading any target.
- **Output**:
  - Model nodes update

Warning

This module is in a *beta* stage and has been tested only with the PIPER child model.

Author

Thomas Lemaire - INRIA

6.12.1 Parameters

6.12.1.1 Points handle

Controls the number of points degrees of freedom sampled on the skin.

6.12.1.2 Points handle

Default stiffness for the point handles target. The current value can be changed in the module.

6.12.1.3 Skin young modulus

Skin triangular FEM Young Modulus. Increase the value to get smoother deformations, decrease it to get sharper deformations. The current value can be changed in the module.

6.12.2 User interface

6.12.2.1 Simulation control

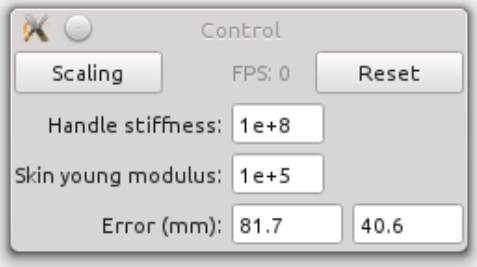


Figure 51: Simulation control

6.12.2.2 Point handle controller

Once the *Point handle interactor* mouse tool is active :

- Click on a red spfere to add a controller for that point
- Click again on the red sphere to remove the controller
- Drag the red, green, blue arrows to set the target value for that controller

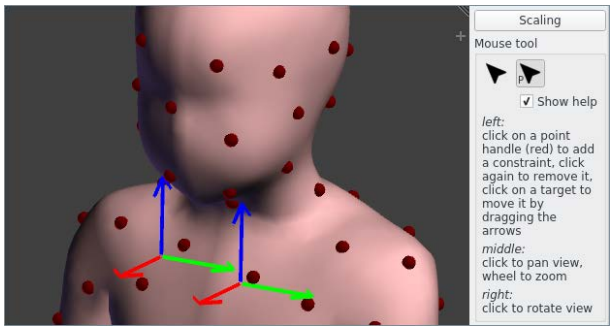


Figure 52: Use of the point handle mouse interactor

6.12.2.3 Display settings

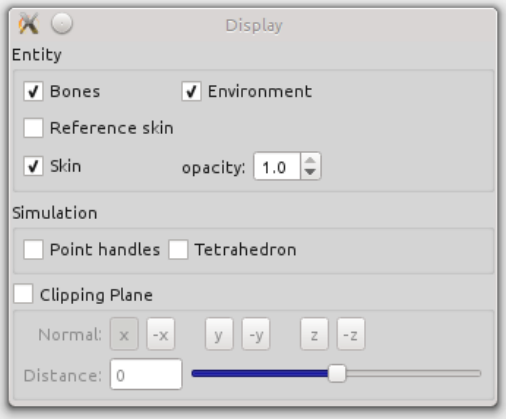


Figure 53: Display settings

6.12.3 Model nodes update

The computation of model nodes must be activated first, then the PIPER model nodes coordinate can be updated.

6.13 Scaling Parameter Module

The Scaling Parameter module allows to apply a scaling factor to parameter values defined in the parsing rule file ("parameter" element).

A typical usage scenario would include the change of material parameters or shell thickness that is correlated with another predictor. An example of application (including a python script to compute material parameters as a function of age) is provided for the PIPER child model.

The module GUI is divided in two part:

- A table: a list of parameters defined in the PIPER model identified by their name in the first column of the table. A default value is set to 1.O for the scaling factor displayed in the second column. The scaling factor can be modified by selecting and setting a new value of the appropriate line.

- A module output: in this panel, the Scaling Parameter module provides output.

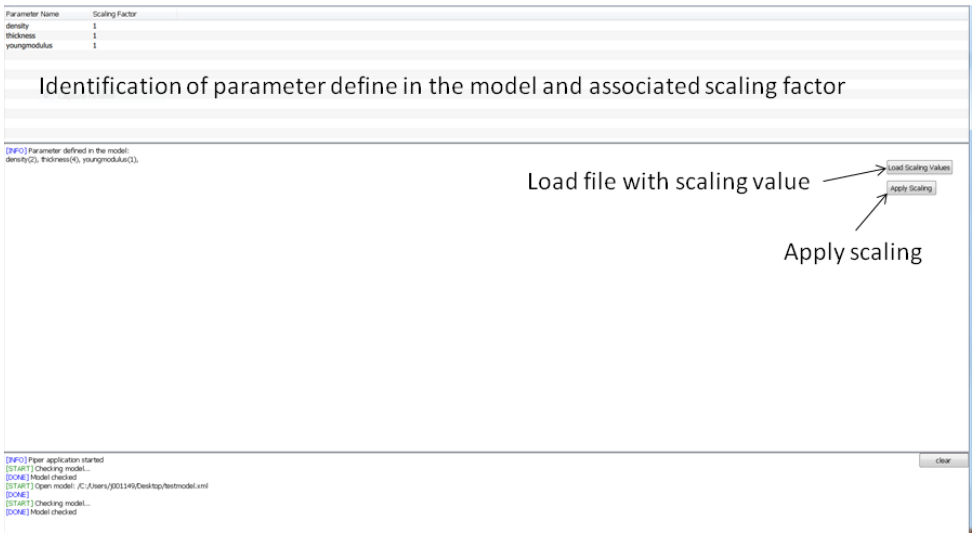


Figure 54: Scaling Parameter Module UI

Button "Load Values": loads scaling factors values defined in a file. Scaling factors values are updated according to values defined in the file.

Example of control points file:

```
parameterName1 1.2
parameterName2 0.85
parameterName3 3
....
```

Button "Appy Scaling": applies scaling factors values to parameter values.

6.14 Scaling the PIPER child model by age

This small module is designed to work with the PIPER child model.

It allows to scale the anthropometric dimensions of the PIPER child model between 1.5 and 6 Y.O. (and beyond but without any check on the performance of the scaled model). The dimensions of the model are derived from GEBOD regressions and expanded below 2 Y.O. using published Candat data. There are also some controls on the shape of the vertebrae. Note also the option to scale the material properties of the model with age. This option is currently experimental and should not be used prior to additional testing.

The module is straightforward to use:

- Open the Anthropo Module.
- Load the PIPER child model that you received together with the PIPER application.
- Click on "PIPER-child scaling" in the right menu, the Child scaling GUI will appear.
- Choose the "Child_scaling" set of control points in the combobox.
- Check the "Preview" checkbox.
- Choose the target age using either the slider or the arrows next to the number of months, or move the second slider to select the desired height (in mm). Whenever you set a new target, you should see the red preview of what the model will look like.
- Click Apply to get the scaled model. *Note: Nugget settings affect this deformation as well*

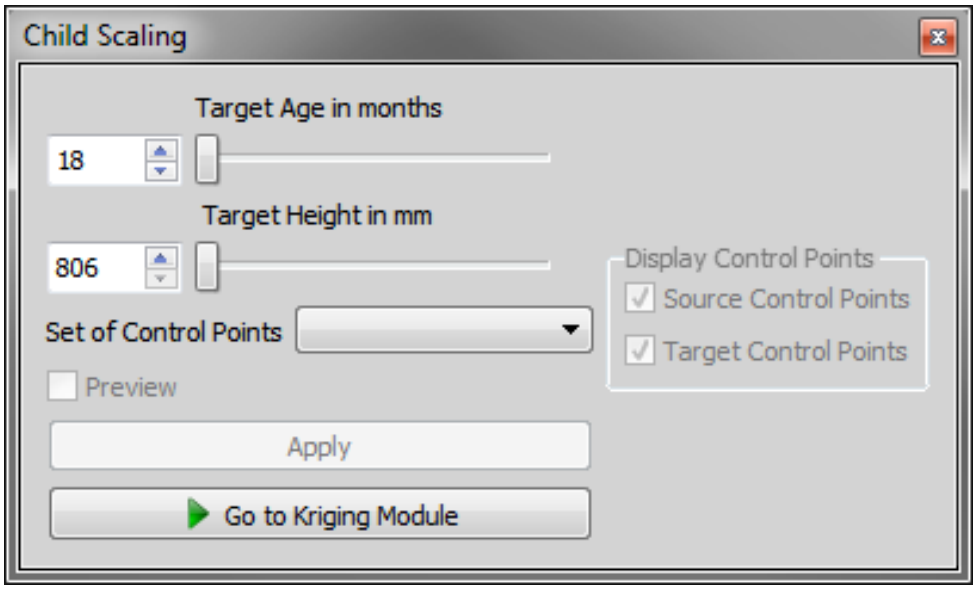


Figure 55: Child scaling GUI

Remarks

- The scaling is done using the kriging method implemented within piper, but the target control points for this transformation are obtained by using a custom octave script tailored specifically for this model to ensure appropriate behaviour (located in share/octave/ChildScaling). PIPER only sends a set of input parameters and name of the output file it will expect upon the scripts termination as arguments to the script. Then the script is invoked and afterwards, the output file with control points is loaded into PIPER. This allows simple integration of model specific personalization scripts.
- For the material scaling, a python script (child_scaling_parameters.py) is used to provide the values of the material parameters which are to be modified by the parameter module .
- This script was integrated as is for historical reasons but it for a new implementation with another model, it could be preferable to use a single python script to update the model, or to use an anthropometric model.

7 Workflow Examples

The PIPER application allows versatile workflows by sequencing different modules.

The following pages explore a few example workflows that you can try.

7.1 Scaling HBM using anthropometry

This workflow will show you how to generate target files of coherent anthropometric dimensions by using the Anthropo Module and use them to scale a Human Body Model.

7.1.1 Create target file

7.1.1.1 Set up the anthropometry module

- Open the Anthropo Module.

Workflow	Model	Modules used
Scaling HBM using anthropometry	GHBMC M50	Anthropo Module, Scaling Constraint Module
Pre-crash positioning and smoothing	Child	Pre-Positioning Module, Smoothing module
Positioning with spine posture prediction	GHBMC M50	Pre-Positioning Module
Model Contour Deformation Target Based Positioning	GHBMC M50	Pre-Positioning Module, Contour Deformation Module
Model Contour Deformation Target Based Personalization	GHBMC M50	Anthropo Module, Contour Deformation Module
Export Partial parts of the mesh to Obj format a re-import it to Piper	GHBMC M50	any module

Table 1: List of workflows

- Click on "From dataset" in the right menu.

7.1.1.2 Generate a new regression file

To generate some target files, you will first need to generate a regression file (if you already have a regression file, you can load it by clicking on "Existing regression" in the "Generate target from" subsection).

- Click on "New regression". A new subsection will appear: "DatasetOptions".

- Click on "Select Dataset" button, and then select the "Adult Dataset(ANSUR)"

- Now we select the predictors we want our regression file to be based upon. For this example, select "STA↩TURE" and "CALF_CIRC".

- Then click one the "Set Population Descriptors" subsection, select one bin, representing 100% of the population, with the gender as "male", and set the age from 30 to 40 years.

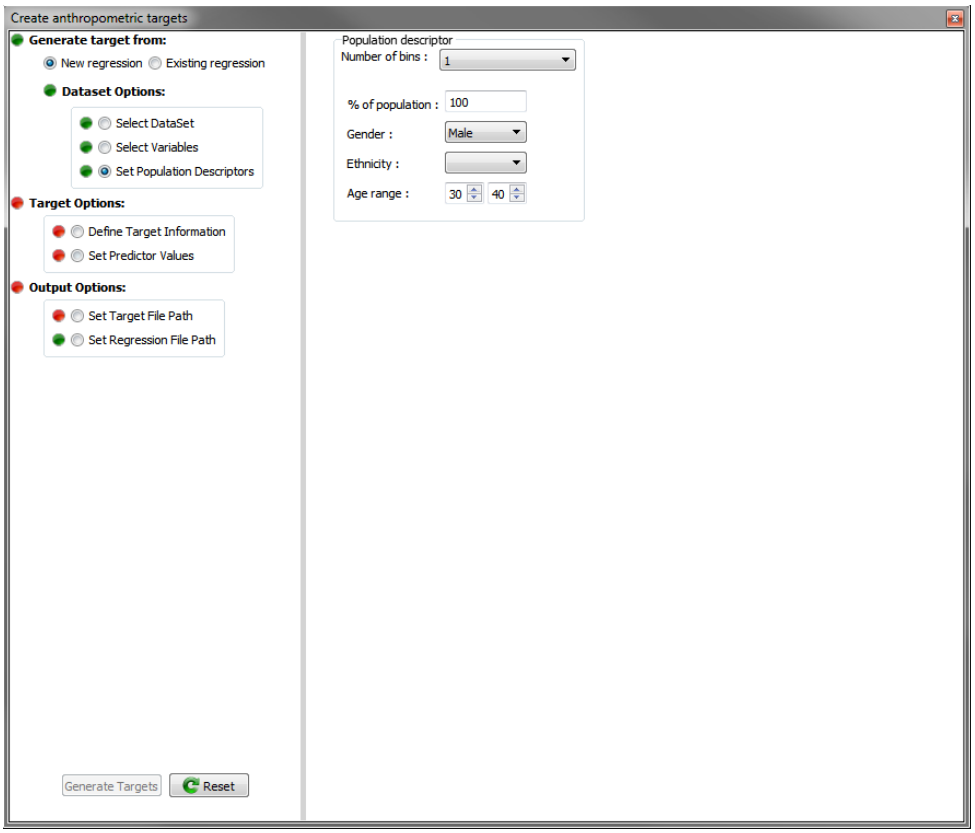


Figure 56: Input data in the GUI

7.1.1.3 Define samples information

Now it is time to define the target values for the anthropometric measurements that we have chosen.

- Click on "Define target information" subsection.

- Select a "sample type" - choose "MeanOnly".

- Select a "sample input type" - choose "Fixed predictor". A new subsection called "Set Predictor Values" appears, that will allow you to set values for the previously selected predictors.

- Leave "sample output type" as "Normal".

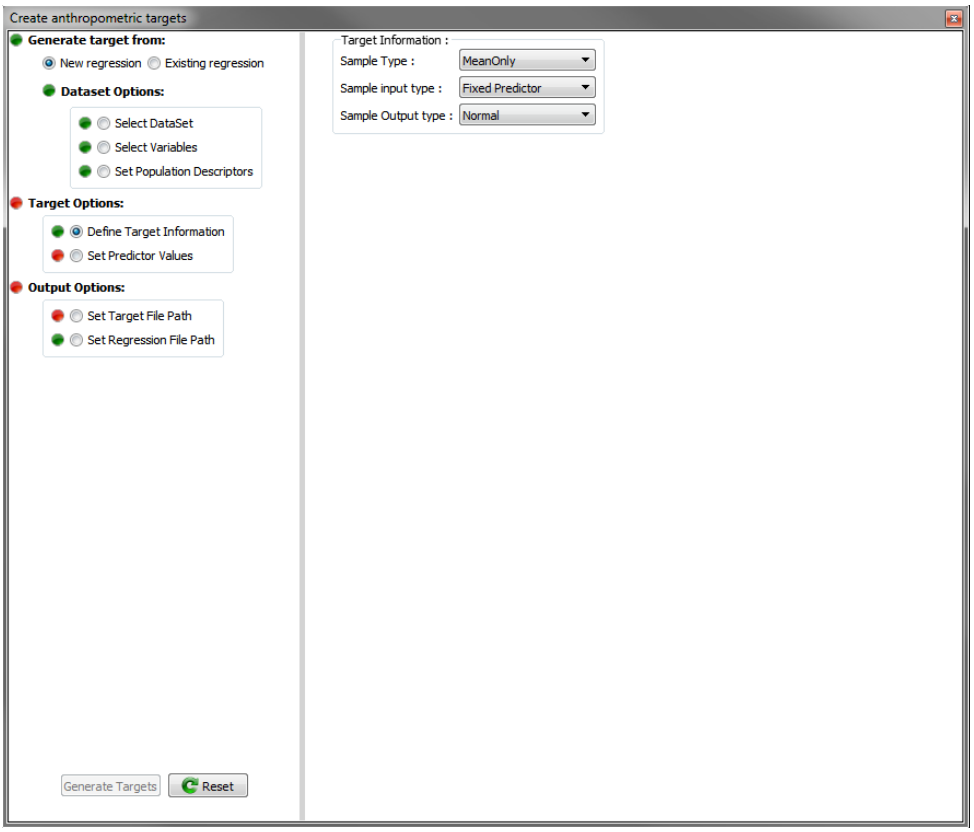


Figure 57: Input data in the GUI

- In the "Set Predictor Values" subsection, set the values for "STATURE" to 1890 and "CALF_CIRC" to 380. (the unit for any length or circumference on the GHBM model are milimeters. The original values of the GHBM is 1790 mm for stature and 354 mm for calf cirumference).

7.1.1.4 Define output information

In this section we will define paths for the target output file, and the regression file if wanted.

- Click on "Set Target File path", and define the location you want to save the target file in, in the right menu.
- By defaut, the regression file are save in the Piper temporary folder. That's why, if you click on "Set Regression File Path", you will see the "No" selected by default under the question "Save Regression ?" question. If you want to save the regression in a particular location, choose "Yes" and define the desired path for the regression path.

7.1.1.5 Generate files

Once your regression and sample information is defined, all the trafic lights should be green, and now you can click on the "Generate Targets" button. It will create the target file in the specified directory. Once you have generated some target files with the anthropometry module, you are able to use them across piper modules to personalize your model.

7.1.2 Personalize the model

You can use the Scaling Constraint Module to link the set of anthropometric targets to the HBM through a Simplified Scalable Model. An example of Simplified Scalable Model designed for anthropometric dimensions defined in A↩ NSUR database can be found here

7.1.2.1 Import Simplified Scalable Model

- Open the model, either through an existing PIPER project or by importing a new one.(see Project menu).
- Import the target file you generated in the Define samples information step using the Target menu.
- Click on the "Anthropo. Model" button and then on "Import Anthropo. Model." and select xml file with the description of the Simplified Scalable Model for ANSUR. (ANSUR_AnthropoModel.xml if you use the one we have download from the link above.)

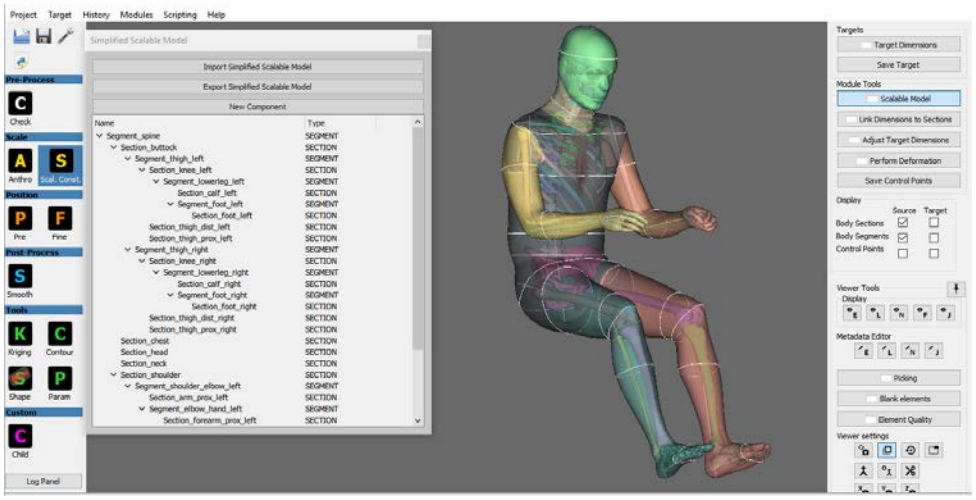


Figure 58: Import Simplified Scalable Model

7.1.2.2 Apply targets

- Click on the "Target Dimensions" button and then on "Use current targets". You should see something similar to Figure Source and target visualization - the simplified scalable model and associated control points are visualized in white color, the target in blue.
- You can edit target values (double click on last column of the table after clicking on the "Adjust Target Dimensions" button)
- To visualize a preview of bone and/or skin deformed shape, click on "Perform Deformation" button and select option to display intermediate skin (or bone) target. (Source and target visualization with preview)

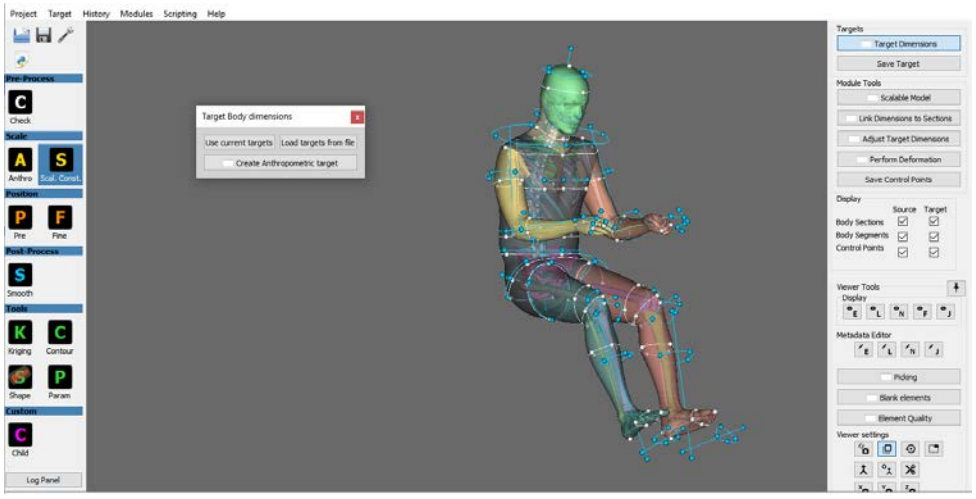


Figure 59: Source and target visualization

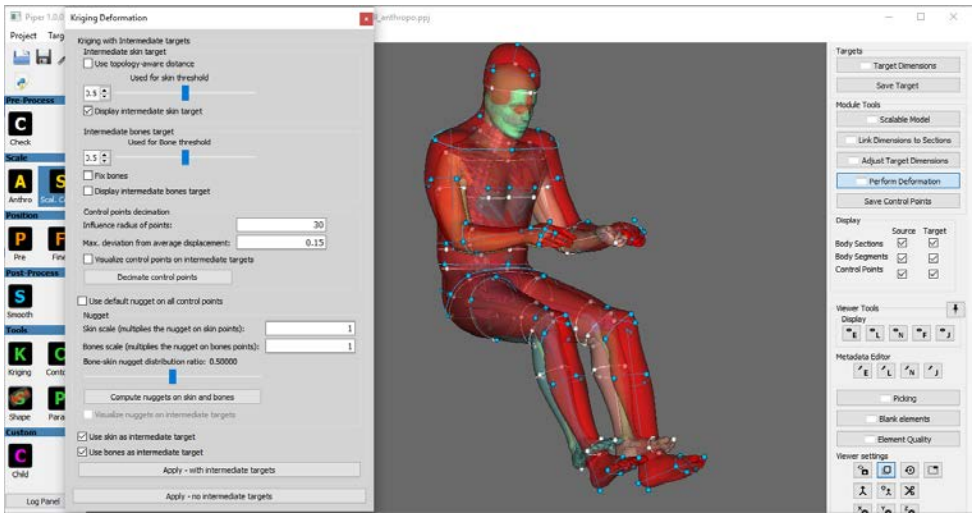


Figure 60: Source and target visualization

7.2 Pre-crash positioning and smoothing

1. Import the Child model using the `Child_model.pmr` and `Child_model.dyn` files (see Project menu)

2. In the module parameters (Module parameters), set bone collision off for *Physics based Positioning*

3. Start the **Positioning** module and do some positioning tasks, for instance :

   (a) Head positioning - **WARNING: Head positioning broken with new child model, skip this step for now**:

      i. Fix both legs (Fixed bone controller)

      ii. Display *World frames* to see the reference coordinate system (Display settings)

      iii. Add a frame controller between **W_Origin** and **B_Skull** on x and y (Frame to frame controller)

      iv. Set x value to **400mm**

      v. Run the positioning until it is stable (Simulation control)

   (b) Left arm positioning

      i. Fix all bones except the left arm, be sure to fix scapula and clavicle too

      ii. Display *Joint frames* to see the joint axis

      iii. Add joint controller for **Left_wirst**, **Left_elbow** and **Left_glenohumeral** (Joint controller)

      iv. Set Left_elbow ry to **-45°** and Left_glenohumeral ry to **-45°**

      v. Run the positioning until it is stable

4. You should get something similar to the following screenshot:

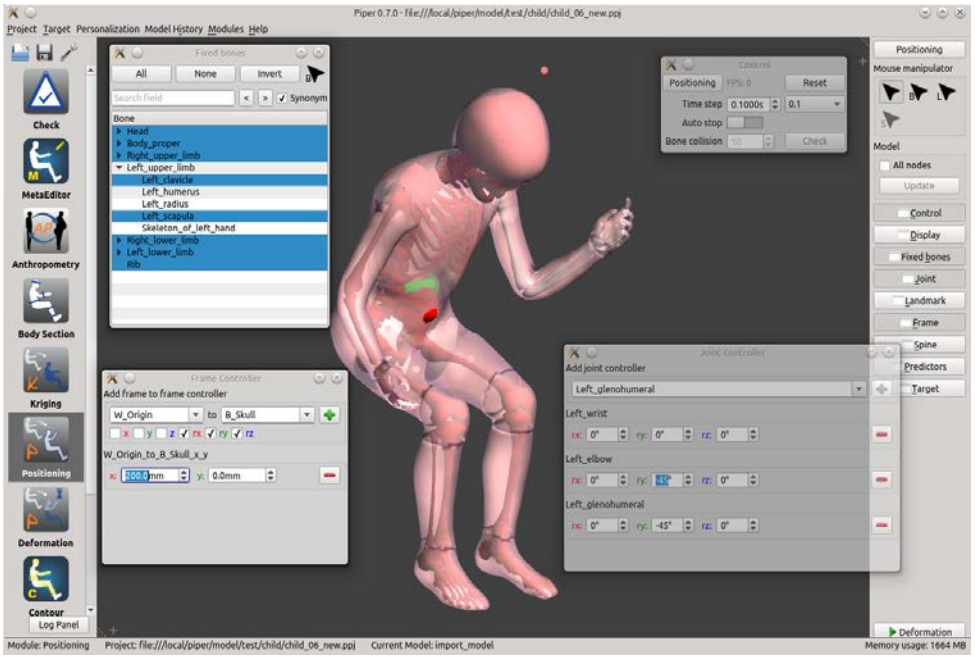


Figure 61: Positionned child model

1. Update the model nodes coordinates (Model nodes update)

2. Start the **Smoothing** module

   (a) Open *Crease Detection*, a window will pop up. Do these actions in it:

      i. Select the imported from history model as baseline

      ii. Select *Left_upper_limb -> Skin_of_left_free_upper_limb* , *Body_proper -> Skin_of_trunk*, *Left_↩ lower_limb -> Skin_of_left_free_lower_limb* and *Right_lower_limb -> Skin_of_right_free_lower↩ _limb* entities to be processed

      iii. Check both the "Generate boxes..." checkboxe on the bottom of the window, uncheck "See selection boxes"

      iv. Press Compute Quality (see Surface crease detection for details). This will perform automatic selection of damaged parts of the skin - you can manually enhance this selection if you feel that the automatic one left out some part that should be considered as damage too.

   (b) Then open *Smooth Surface* and in the tool window do:

      i. Select the same entities as in the previous step

      ii. Check *Smooth only selected* (should be checked by default)

      iii. Press Smooth Surface (see Smoothing the surface of an entity for details). You should get something similar to the following screenshot:

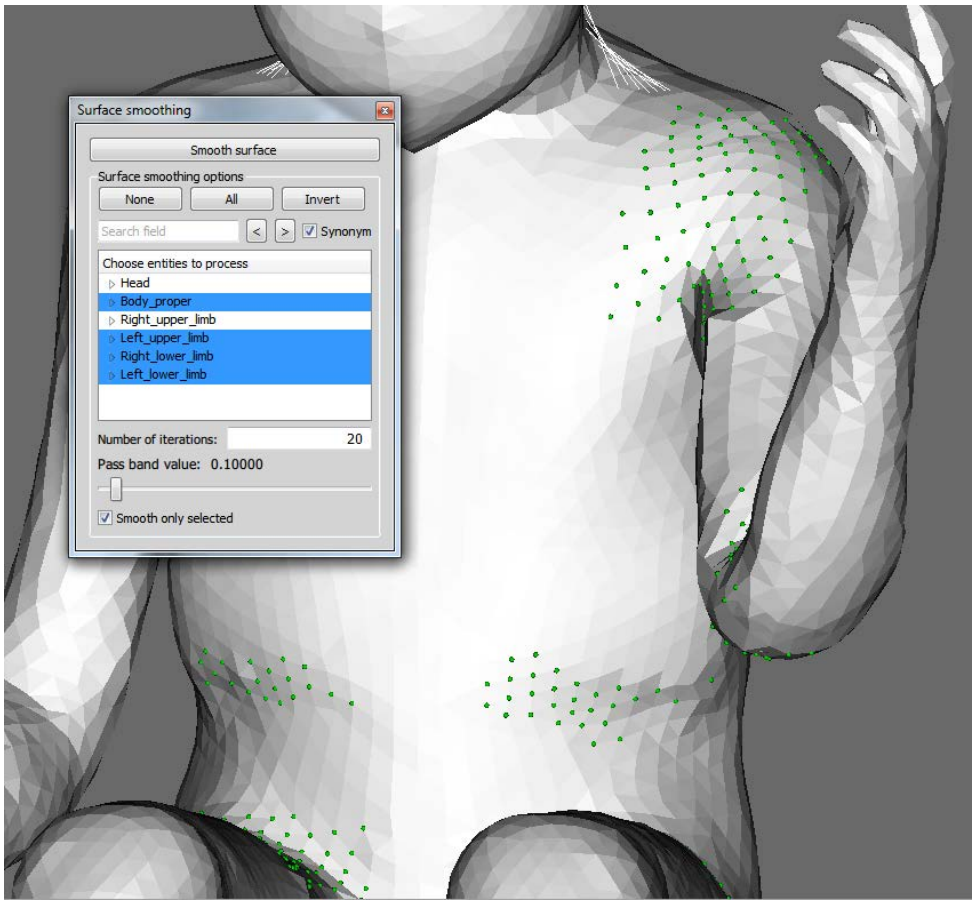


Figure 62: Smoothed skin of the arm

(c) The skin now looks better, but we can use Blanking and VTK toolkit Quality metrics to see that the inside elements are significantly distorted. *Note - we no longer need the nodes to be selected, you may deselect them in order to "clean-up" the visualization.*

i. Use element selection by box to select the left arm and half of the shoulder

ii. Blank selected elements using the *Blank elements* tool

iii. Use the *Compute Quality* tool too see the element quality - the color scale setting can be seen on the screenshot (white elements are elements out of the scale, in this case with scaled jacobian below -0.3):

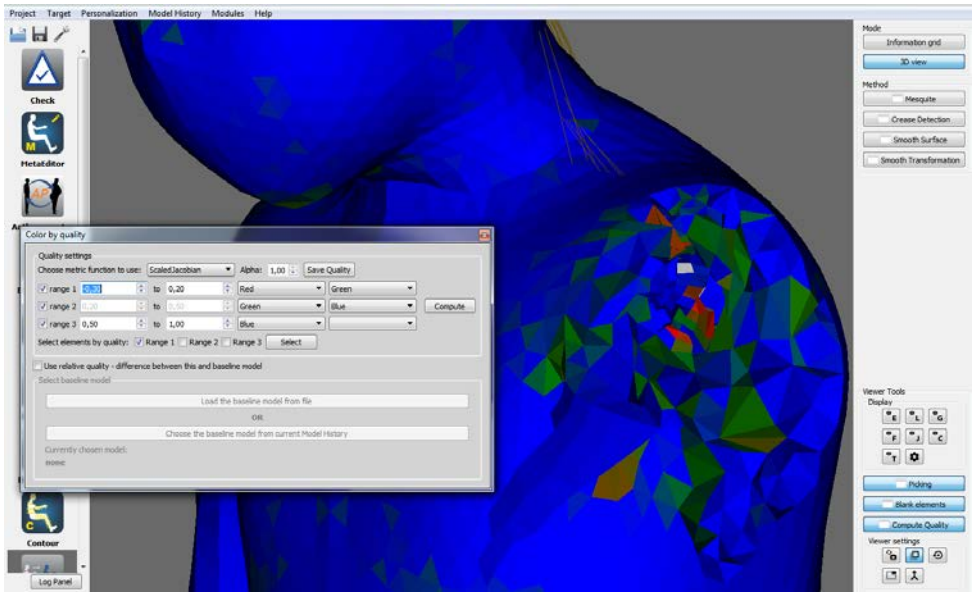


Figure 63: Internal element quality of shoulder

(d) We will smooth the transformation to get a better result:

i. Use the Picking toolbox to create a box around the shoulder area, similar to the one in the following figure (we used the node box picking and held CTRL so that no points were selected, but anything that will create some box will do the job):

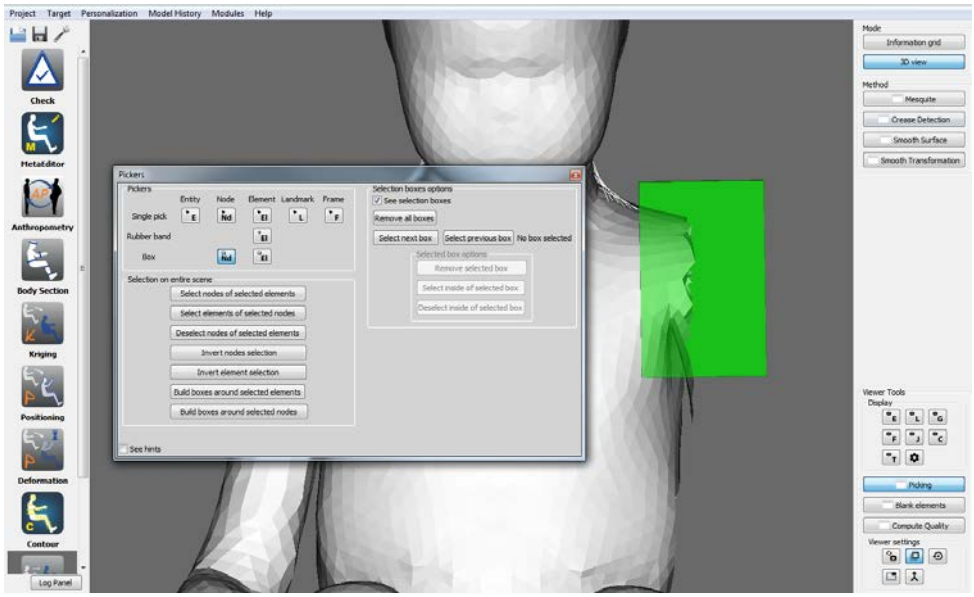


Figure 64: Box for transformation smoothing

ii. Open the Transformation smoothing window

iii. Select the whole *Body_proper* and *Left_upper_limb* regions - not only the skin, but whole regions, this way both the bones and the skin will be used as targets.

iv. Press *Smooth by kriging*, wait for the result

v. Compute the element quality again using the same color-scale. As you can see in the followin figure, the element quality will increase significantly:

vi. Now you can repeat this step (6d) in a similar way to smooth the hip joints region as well - or do it all at once, the Transformation smoothing will process all boxes that you create. However, don't forget to additionally select entities relevant as targets for the hip smoothing, i.e. both lower limbs

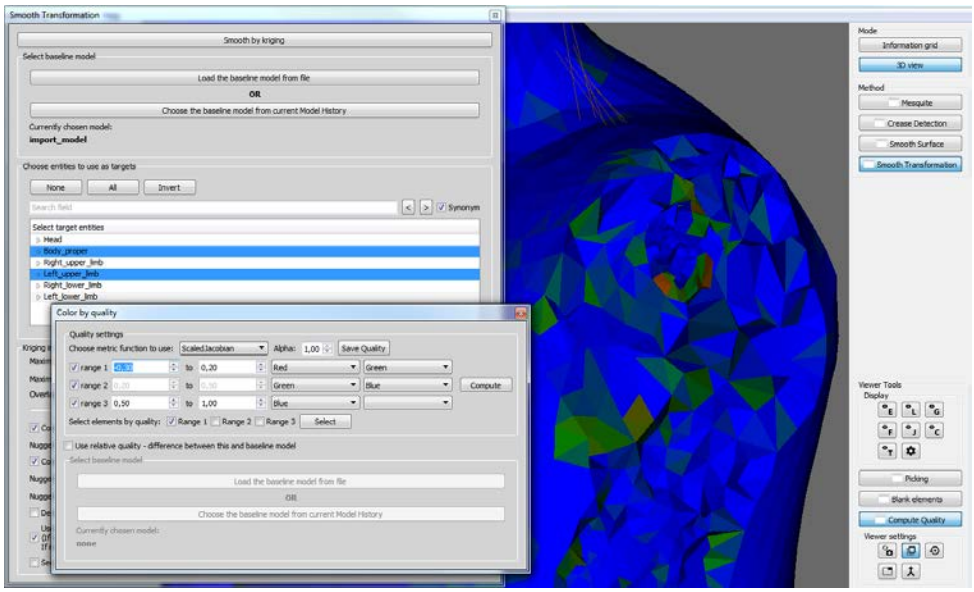


Figure 65: After transformation smoothing

3. And finally you can export the positionned model back to the FE code format in the Project menu

7.3 Positioning with spine posture prediction

1. Import the GHBM model using the `GHBMC_M50_v4-1-1_description_LSDyna.pmr` and `GHBMC_M50-O_Main.dyn` files (see Project menu)

2. In the module parameters (Module parameters), set bone collision **off** for *Physics based Positioning*

3. Start the *Positioning* module and do some spine positioning using the physiological spine predictor :

   - Fix both legs (Fixed bone controller)
   - Enable the *Spine* predictor from the Positioning predictors
   - For source posture select **Seating erect**, for target select **Forward flexed**, keep posture value to **0**, keep align on gravity **on** and *Update*
   - Set time step to **0.01**, run the positioning until it is stable, if necessary reduce the simulation time-step (Simulation control)

4. You should get something similar to the following screenshot:

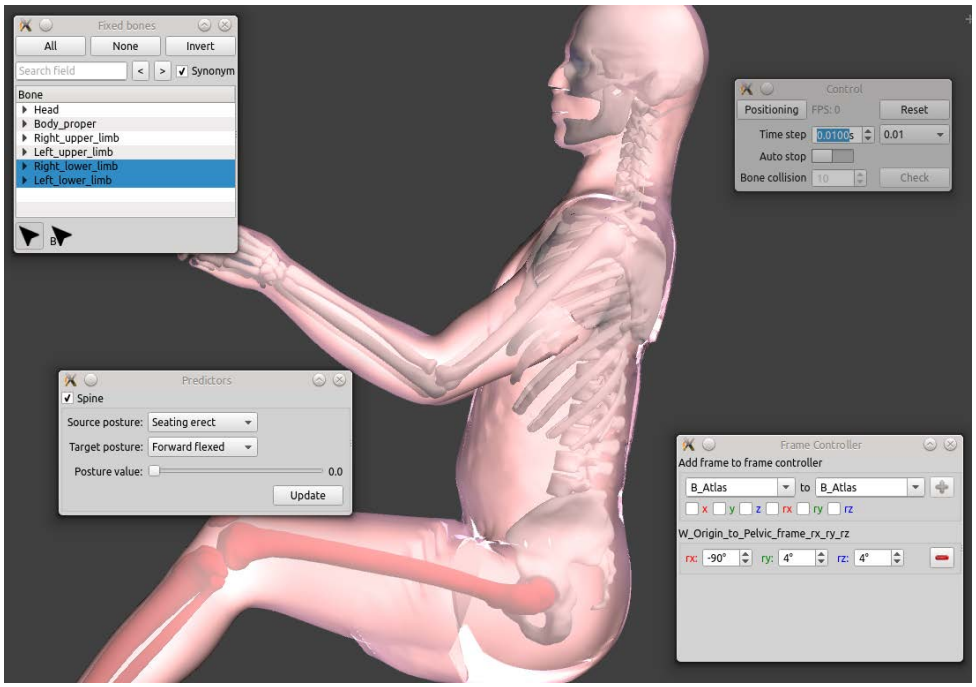


Figure 66: Spine posture predictor on GHBM model, fixed bones, and automatically added pelvis orientation controller

5. Now let's do some more spine positioning :

   (a) Be sure the simulation is stopped

   (b) Fix the pelvis to keep its posture orientation

   (c) Deactivate the *Spine* predictor

   (d) Open the Spine controller, select **C1**, **T1**, **T12** and **L5** as spline control points

   (e) Activate the **mouse spine interactor**

   (f) Adjust the Display settings, align a view on the back of the GHBM to be confortable to edit the spline and give it a scoliosis shape, you can obtain some thing like :

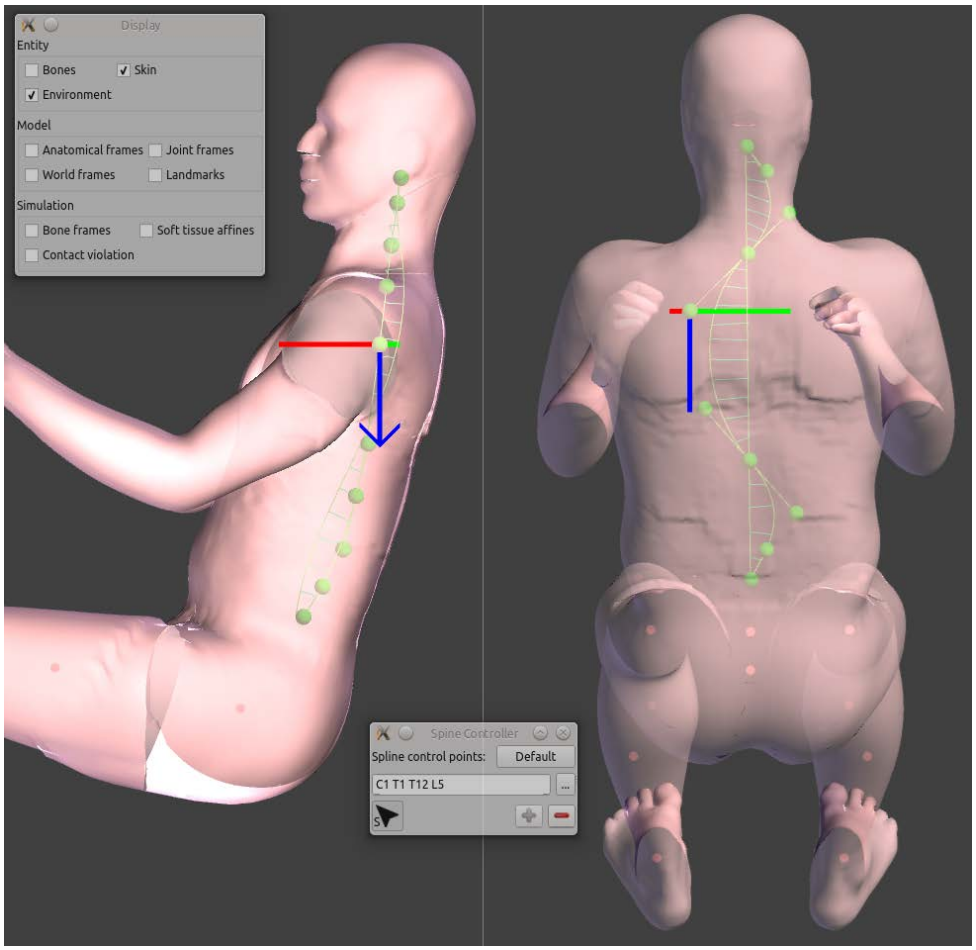


Figure 67: Scoliosis spine shape

(g) Run the positioning until it is stable, if necessary reduce the simulation time-step

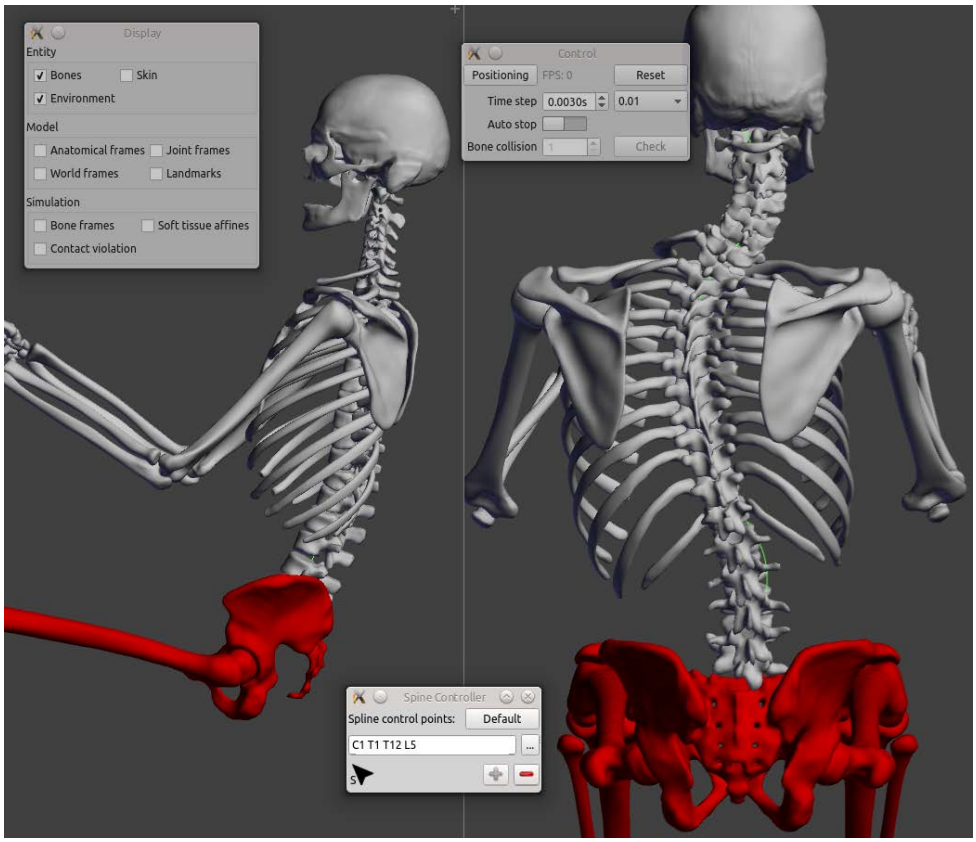


Figure 68: Scoliosis spine shape reached

7.4 Model Contour Deformation Target Based Positioning

A) IMPORT THE GHBM MODEL

1. import a HBM from its vendor format files (see Project menu)
2. check and explore the imported model with the Check Module

B) TARGET BASED POSITIONING (IMPORTING TARGET FROM INTERACTIVE MODULE)

1. Import the target file using Target Menu
2. Go to Contour Deformation Module
3. Press Positioning button on the side bar.

This will open a new window having controls for re-positioning the HBM

1. Press "Load ContourCL file" button to open the contourCL.xml. This step will cause yellow colored control lines to appear with the model display. These control lines provide a guiding structure for the generation of Contours around the HBM. It is also used in the Target Based Repositioning for traversal around the various Body Regions and to find the joint angles.
2. Press "Use Landmark Targets" switch to go into re-positioning based on Targets mode.
3. The target processing is an iterative process. The body regions are traversed along the contourCL structure. In order to start this iterative process follow the instructions below:
   (a) Press "Process Landmark Targets button". Please note as of now only Landmark Targets are supported.

(b) The immutable fields Joint,Side, Type and Angle will get populated automatically. Please note, currently only flexion-extension is supported in this Mode.

(c) Press "Generate Contours" button in order to generate the contours around the HBM.

(d) Press "Deform Contours" button in order to deform the contours according to the computed Joint,Side,Type and Angle computed in the Step 6-b.

(e) Press "Reposition" button. This will run the contour based repositioning algo to deform the model, based on information synthesized from Landmark Targets.

(f) Press "Continue" button to move on to the next target for the remaining body regions, until all the body regions are processed.

(g) If the computed angle is zero, stop.

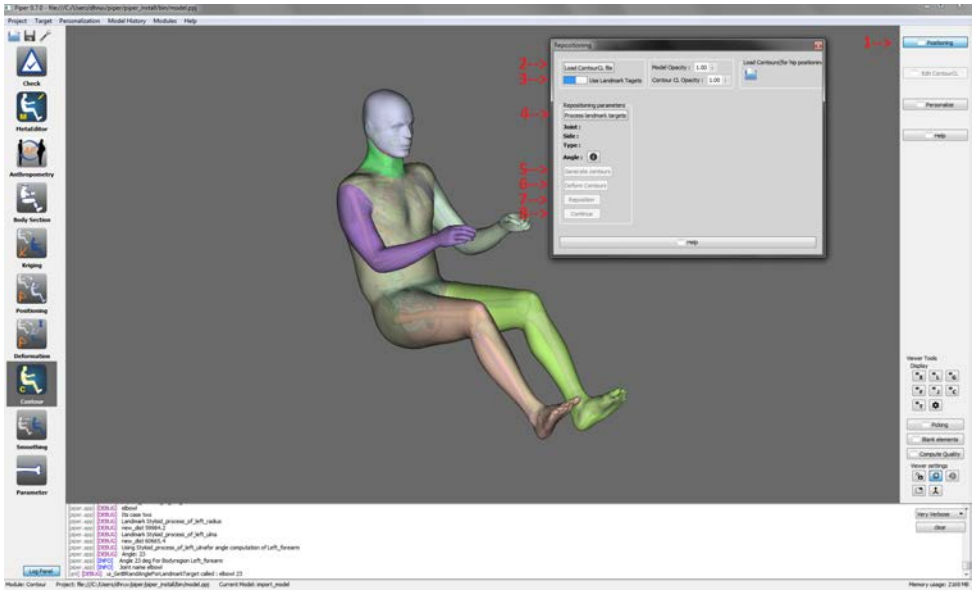


Figure 69: Workflow For Contour Deformation Positioning Using Landmark Targets

7.5 Model Contour Deformation Target Based Personalization

7.5.1 Generate Target Files

1. Open the Anthropometry module.
2. Click on "Create Anthropometry" button on the right menu.
3. Select "Generate a new regression file".
4. Select "BMI" "WEIGHT" "STATURE" as predictors, and define a population of two bins. One representing males and the second one representing females. Please refer to Generate a new regression file for details about how to create a regression file.
5. Click on "Define sample information".
6. Select "MeanOnly" as sample type.
7. Set a name to the meanOnly output target file. **This is the file you will need for the Target Based Personalization (Importing target from anthropometry module).**
8. Select "Fixed predictor" as sample input type.
9. A list of predictor that you selected for the regression file should appear at the bottom of the window. Set the folowing values: BMI = 25, WEIGHT = 730 and STATURE = 1750.

10. Select "Normal" as sample output type.

11. Click on the green arrow to generate the regression and target file.

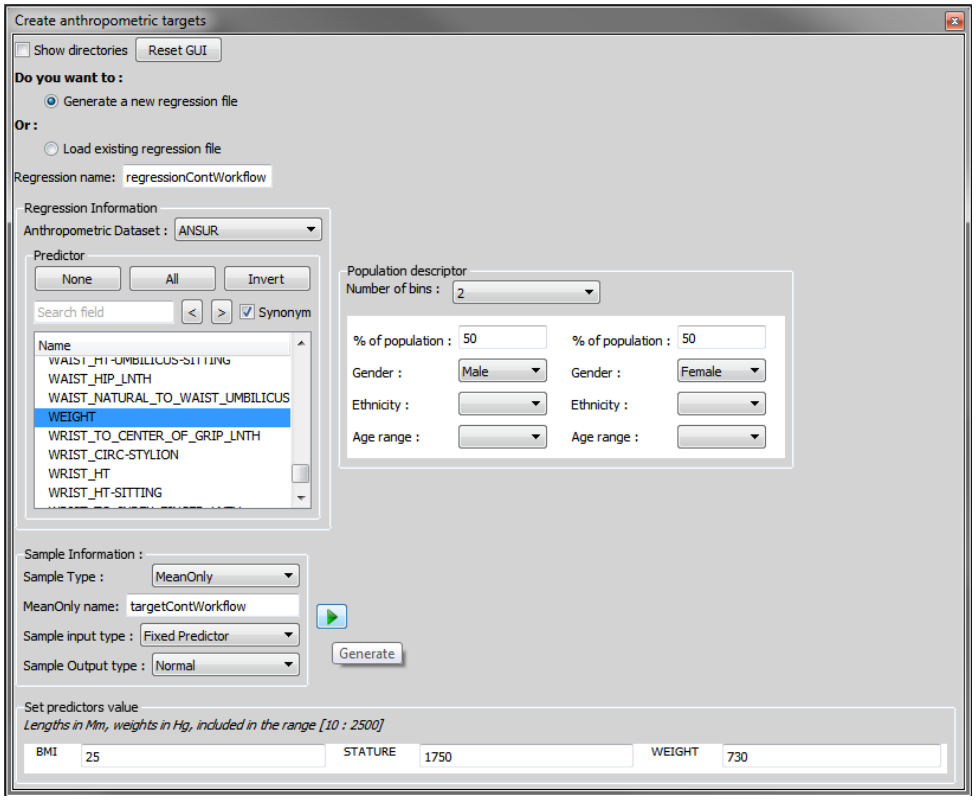


Figure 70: Input data in the GUI

7.5.2 Import GHBM Model

1. Import a HBM from its vendor format files (see Project menu).

2. Check and explore the imported model with the Check Module.

7.5.3 Target Based Personalization (Importing target from anthropometry module)

1. Open the Contour Deformation Module.

2. Open the Personalization window by clicking the "Personalize" button on the right panel.

3. Click the "Load Personalization Contours" button and select the Personalization_Contours.txt file to load the personalizaiton contours. You will find it in the metadata folder of the GHBM model.

4. Click the "Load Body Data" button and select the BodyData.txt file to load the personalization ratios. You will find it in the metadata folder of the GHBM model.

5. Click the "Load Anthropometric Target File" button and select the target file you created in the Generate Target Files part (step 7).

6. Choose "Anthropometric target values" in the "Options".

7. Click on the "Personalize" button to personalize the model.

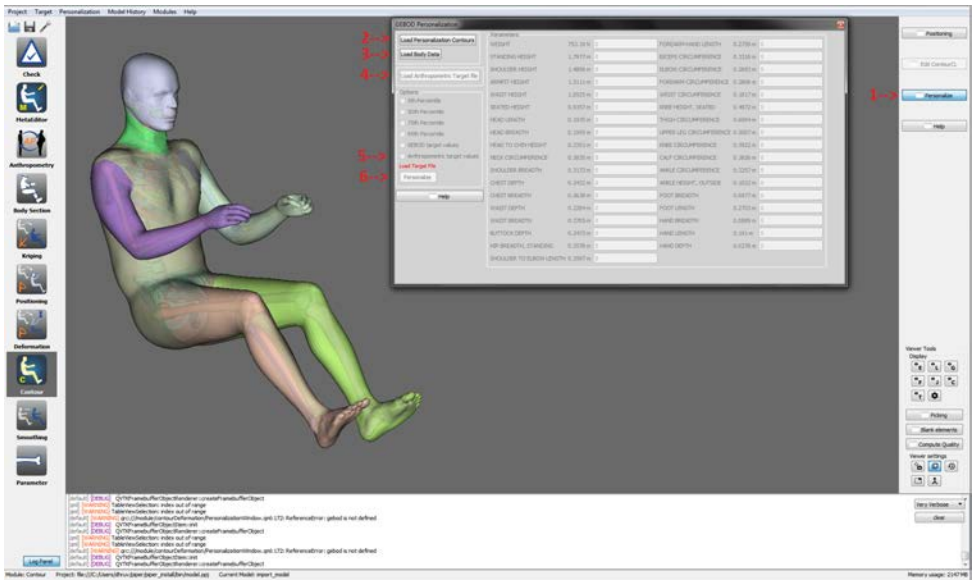


Figure 71: Workflow For Contour Deformation Personalization Using Anthropometric Targets

7.6 Export Partial parts of the mesh to Obj format a re-import it to Piper

This workflow come after a position modification or a personalization of the model within piper that went wrong.

If one of the points are not in correct location, or if you want to smooth a part of the model after a transformation, this workflow will show you how to edit a part of the Piper mesh with your favorite 3D engine software. Note that you can use the scritps 'exportSelectedGeometryToDyna' & 'updateNodesfromDyna' the same way as this workflow to work with the LsDyna file format.

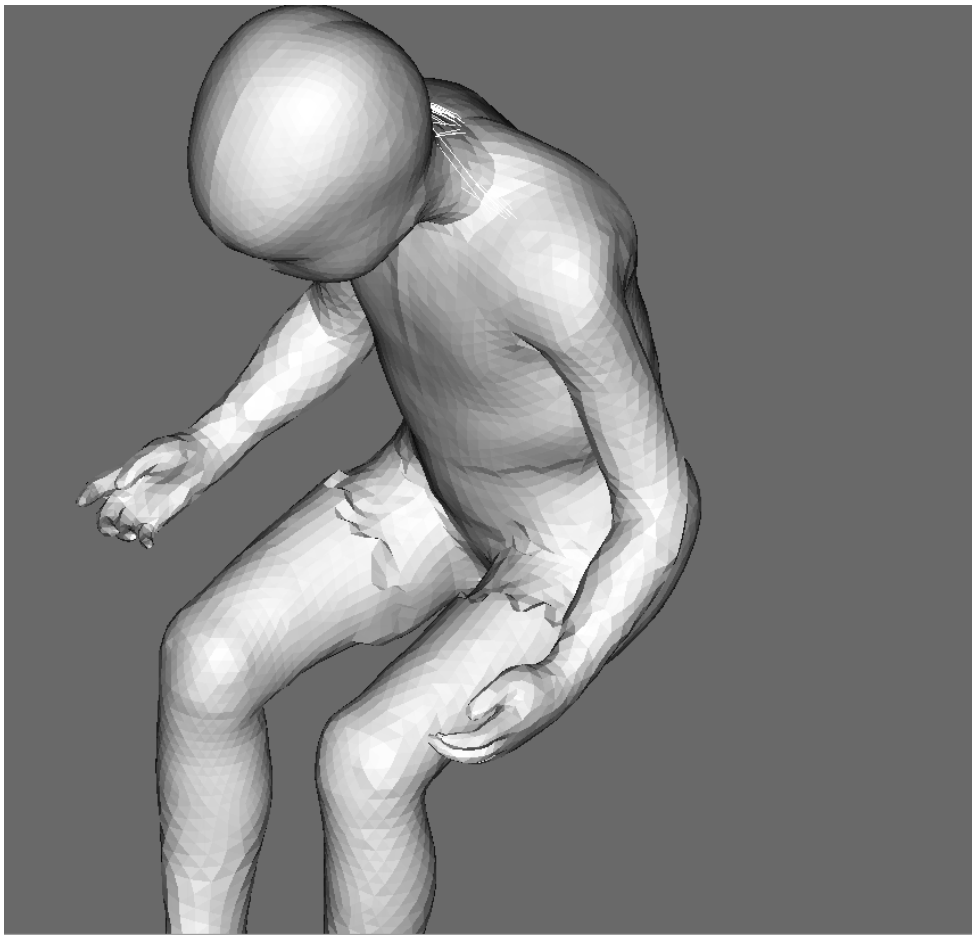


Figure 72: Example of disorted model

7.6.1 Export to Obj file

1. Open the picking window from any module. Make sure that the option "visualize full model" is unchecked in the Entity Display window. Export supports only selection on model (non blanked) entities.

2. Click on "Box element Picker"

3. Select the part of the mesh you want to export with the box picking tool.

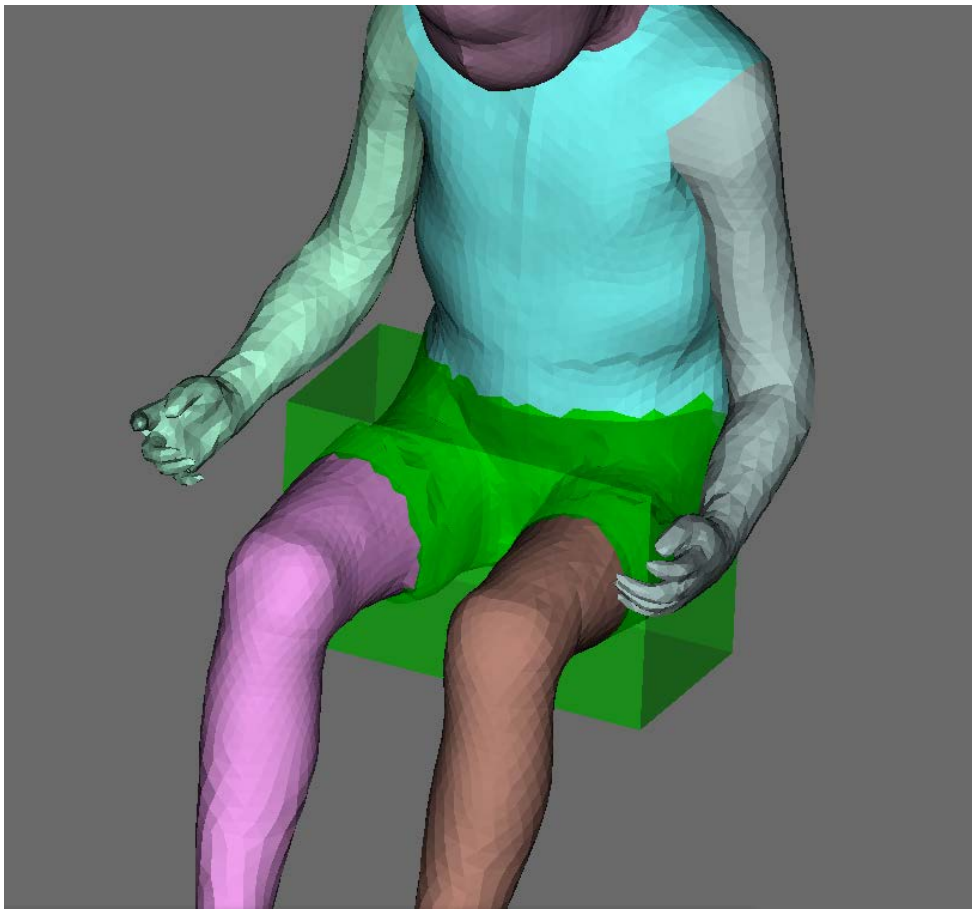


Figure 73: Select the disorted parts

1. Click on "Scripting" tab from the upper menu and select "run script"
2. Select the script "ExportSelectedGeometryToObj" located in: "piper/share/example/scripting/" folder.
3. As arguments, put the path of the output .obj file you want to generate, and click on "Ok"

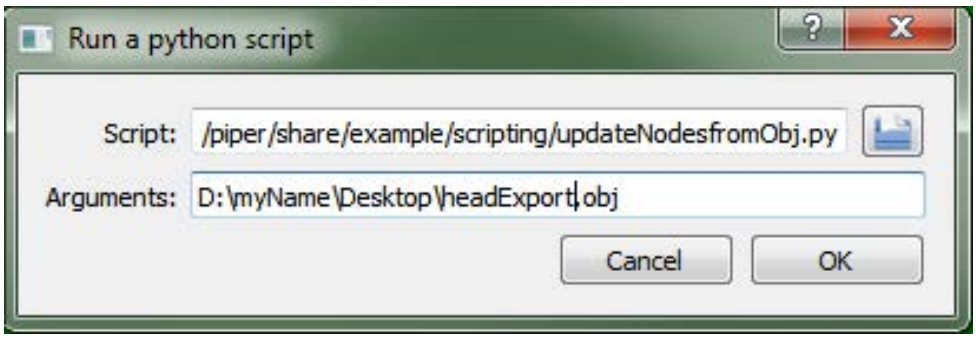


Figure 74: Example of how to use the script menu in Piper

7.6.2 Edit the obj file

Now you can edit the obj file within a 3rd party 3d software. The tests for this workflow have been done with Meshlab that you can find: here

1. Open Meshlab and import the obj file you just extracted from piper.

2. Do a modification on the mesh, like applying a smoothing algorithm.

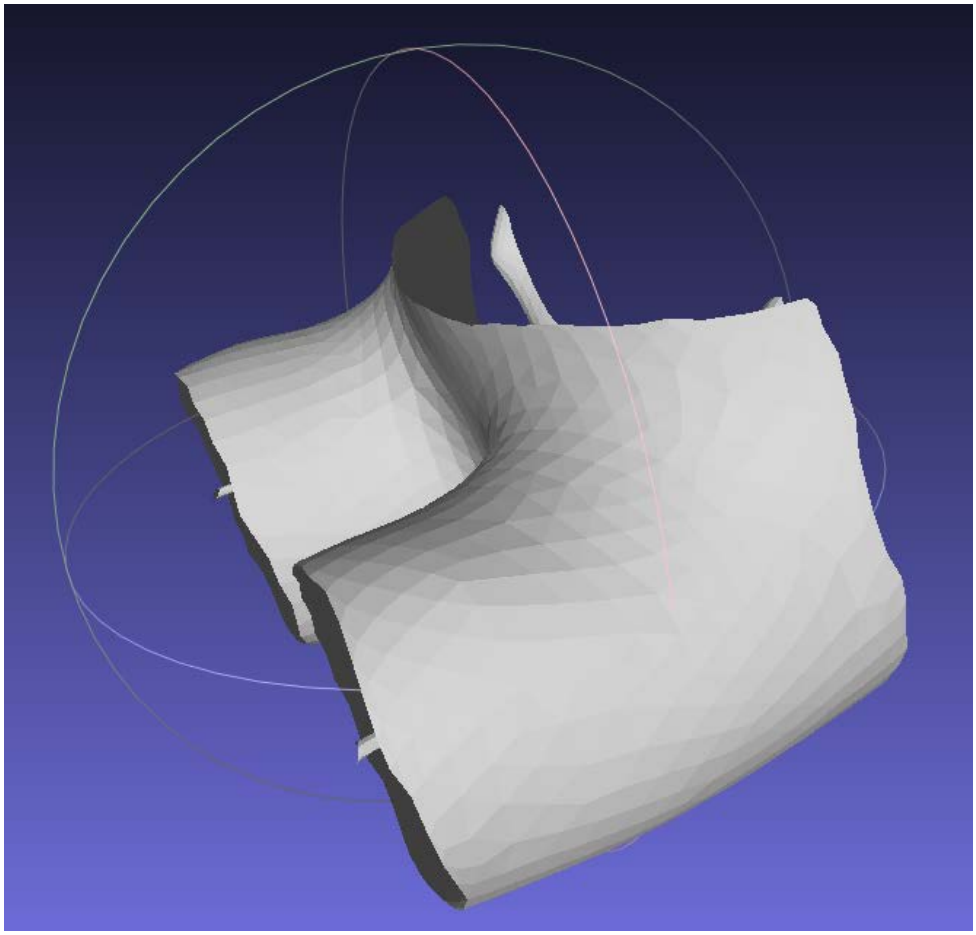


Figure 75: Example of how to use the script menu in Piper

1. When it's done, export the model as an .obj and overwrite the one you exported previously from PIPER application.

7.6.3 Import the obj back to Piper

1. Go back to the Piper application.

2. Click on the "Scripting" tab again.

3. Select the script "UpdateNodesFromObj" within the same script folder.

4. Give the path for your obj file and a name to be stored in the model history as an argument. example "c↩ :/path/to/obj/file.obj modelWithSmoothedHips"

5. Click on "Ok"

The selected part of the mesh that you extracted and modified in an other 3d application is now updated accordingly to the modifications you've done on it.

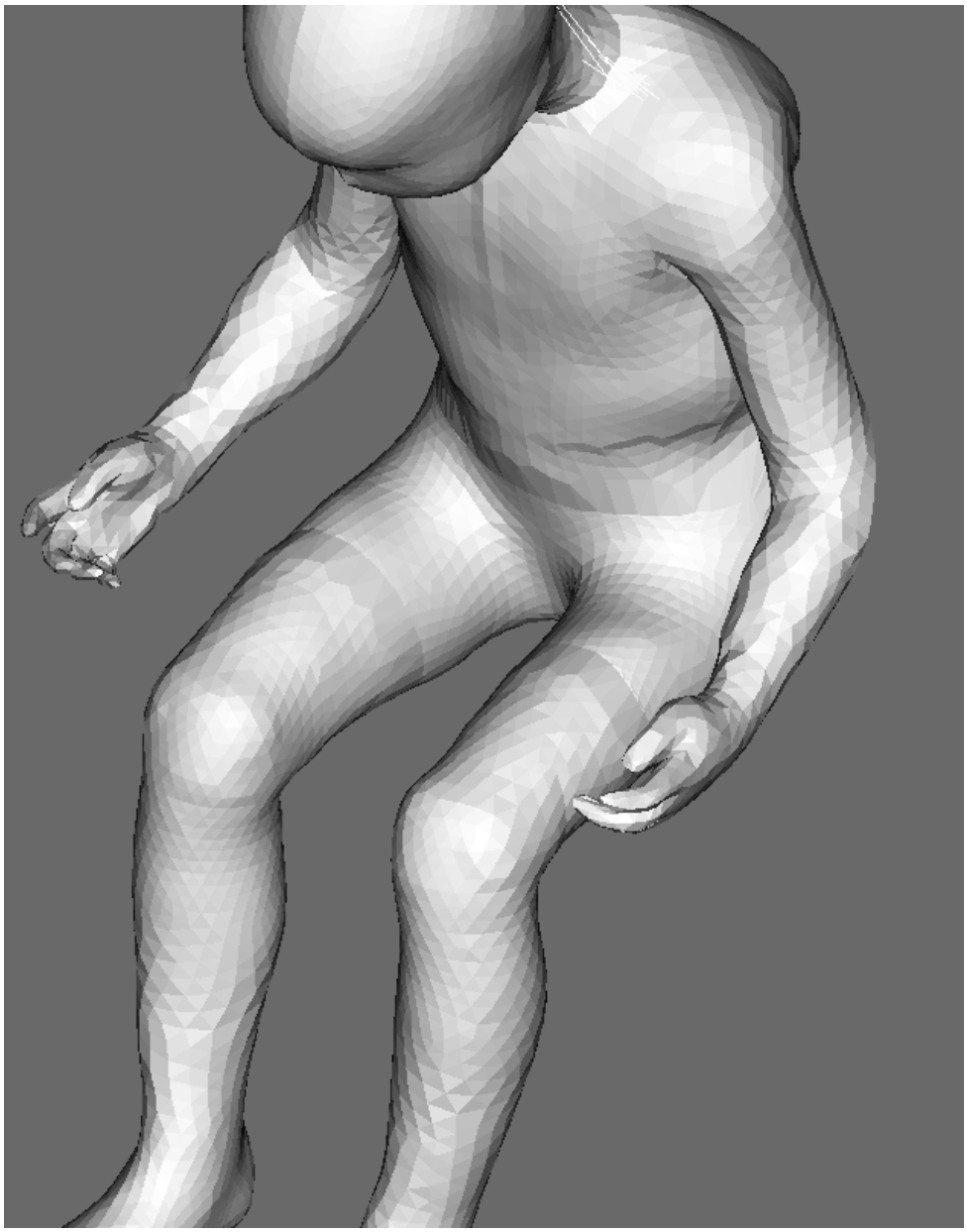


Figure 76: Smoothing part reimported on Piper

If you wish you can apply a 3D mesh smoothing algorithm on the model currently displayed in the Piper application. To do that you can read the folowing workflow: Pre-crash positioning and smoothing

8 Scripting and Batch mode

8.1 Overview

The *scripting* capabilities of the PIPER application let the user insert custom operation in his workflow by Writing a script. It is also possible to run complete workflows from scripts, this is the so called Batch mode. The scripts are written in `Python 2.7`, see PIPER Python packages for a reference of the PIPER specific python API.

8.2 Running a script

From the *Scripting* menu any python script can be run at any time. If the script expects parameters, they can be specified in a space separated list. A list of standard script is provided with the application (they are located in `share/python/scripting`). Additionnaly the user can set a custom script folder from which the python scripts are listed.

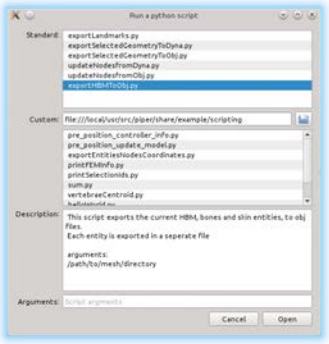


Figure 77: Running a python script with arguments

8.3 Writing a script

Using the provided `piper` python packages, the user can access and modify the current application *model*, *metadata* and *targets*. The following example are available in the `share/example/scripting` folder.

8.3.1 Basic functionalities

The traditional *Hello world !* example (in `helloWorld.py`):

```
1 # Simple hello-world script
2
3 # access PIPER application functions
4 import piper.app
5
6 piper.app.logInfo("Hello World !")
```

A script which manipulates its arguments (in `sum.py`) :

```
1 # Compute the sum a + b
2 #
3 # arguments:
4 #  a b
5
6 import sys
7
8 # access PIPER application functions
9 import piper.app
10
11 def computeSum(a, b):
12     piper.app.logInfo("a+b = {0}".format(a+b))
13
14 if (3 != len(sys.argv)):
15     raise RuntimeError("Invalid arguments")
16 else :
17     computeSum(float(sys.argv[1]), float(sys.argv[2]))
18
```

8.3.2 Accessing model metadata and geometry

The following example shows how to access the current model and metadata, and also query the anatomy database. The script computes the vertebrae centroids and print the result in the application log (in `vertebrae`↩

Centroid.py):

```
1 # Compute the centroid of each vertebra and print it in the application log
2
3 # numpy is a standard package for matrix manipulation
4 import numpy
5
6 # access PIPER human body model functionnalities
7 import piper.hbm
8 # access anatomy database functionnalities
9 import piper.anatomyDB
10 # access PIPER application functions
11 import piper.app
12
13 model = piper.app.Context.instance().project().model()
14
15 # if the model is empty, we can do nothing
16 if (model.empty()):
17     piper.app.logWarning("Model is empty")
18 else:
19     # get metadata and fem from the current project
20     metadata = piper.app.Context.instance().project().model().metadata()
21     fem = piper.app.Context.instance().project().model().fem()
22     for (name,entity) in metadata.entities().iteritems() :
23         # check if this entity is a vertebrae
24         if piper.anatomyDB.isEntityPartOf(name, "Vertebral_column", True):
25             piper.app.logInfo("{0}: {1}".format(name, computeCentroid(entity, fem)))
26
27 # compute the centroid of an entity
28 def computeCentroid(entity, fem):
29     c = numpy.zeros((3,1)) # initialize the centroid, notice the (3,1) shape for a vector
30     envelopNodesId = entity.getEnvelopNodes(fem)
31     for id in envelopNodesId :
32          c += fem.getNode(id).get()
33     return (c / len(envelopNodesId)).flatten()
```

The following example shows how to export the currently selected nodes and elements to an external .obj file (in exportSelectedGeometryToObj.py).

```
1 # This script exports the current selected geometry (nodes + triangles)
2 # It also writes a json file which stores correspondances between obj vertices and piper nodes id
3 # for further update with updateNodesFromObj.py script
4 #
5 # arguments:
6 #  /path/to/mesh.obj
7
8 # author: Thomas Lemaire date: 2017
9
10 import sys
11 import json
12
13 import piper.hbm
14 import piper.app
15
16 def writeObj(fem, nodeIds, elem2DIds, objFilePath):
17     # gather node ids from nodes and elements
18     allNodeIds = set()
19     for id in nodeIds:
20         allNodeIds.add(id)
21     for elemId in elem2DIds:
22         elem = fem.getElement2D(elemId)
23         if not elem.getType() in {piper.hbm.ELT_TRI, piper.hbm.ELT_QUAD}:
24             piper.app.logWarning("Unsupported: elem: {0} type: {1}".format(elemId, elem.getType()))
25             continue
26         ids = elem.get()
27         for id in ids:
28             allNodeIds.add(id)
29     # in an obj file, vertices are referenced by their index, starting at 1, to describe faces
30     nodeIdToObjIndex = dict()
31     objIndexToNodeId = dict()
32     with open(objFilePath, 'w') as objFile:
33         piper.app.logInfo("Writting obj file {0} ...".format(objFilePath))
34         objFile.write("# File generated by the PIPER application\n")
35         currentIndex = 1 # obj convention starts at 1
36         for nodeId in allNodeIds:
37             nodeIdToObjIndex[nodeId] = currentIndex
38             objIndexToNodeId[currentIndex] = nodeId
39             coord = fem.getNode(nodeId).get().flatten()
40             objFile.write("v {0} {1} {2}\n".format(coord[0], coord[1], coord[2]))
41             currentIndex+=1
42         for elemId in elem2DIds:
43             elem = fem.getElement2D(elemId)
44             ids = elem.get()
45             if elem.getType() == piper.hbm.ELT_QUAD:
```

```
46                 idsQuad = elem.get()
47                 objFile.write("f {0} {1} {2}\n".format(nodeIdToObjIndex[idsQuad[0]], nodeIdToObjIndex[
    idsQuad[1]], nodeIdToObjIndex[idsQuad[2]]))
48                 objFile.write("f {0} {1} {2}\n".format(nodeIdToObjIndex[idsQuad[0]], nodeIdToObjIndex[
    idsQuad[2]], nodeIdToObjIndex[idsQuad[3]]))
49                 continue
50             elif elem.getType() == piper.hbm.ELT_TRI:
51                 objFile.write("f {0} {1} {2}\n".format(nodeIdToObjIndex[ids[0]], nodeIdToObjIndex[ids[1]],
    nodeIdToObjIndex[ids[2]]))
52             else:
53                 continue
54     objIdFilePath = objFilePath+".json"
55     with open(objIdFilePath, 'w') as objIdFile:
56         piper.app.logInfo("Writting obj id file {0} ...".format(objIdFilePath))
57         json.dump(objIndexToNodeId, objIdFile)
58
59 model = piper.app.Context.instance().project().model()
60
61 # if the model is empty, we can do nothing
62 if model.empty():
63     piper.app.logWarning("Model is empty")
64 elif 2 != len(sys.argv) :
65     raise RuntimeError("Invalid arguments")
66 else :
67     modelVTK = model.fem().getFEModelVTK()
68     nodeIds = modelVTK.getSelectedNodes()
69     elem2DIds = modelVTK.getSelectedElements(2)
70     piper.app.logInfo("nb selected nodes: {0}".format(len(nodeIds)))
71     piper.app.logInfo("nb selected elements: {0}".format(len(elem2DIds)))
72     writeObj(model.fem(), nodeIds, elem2DIds, sys.argv[1])
73
74
```

The following example shows how to export entities nodes *id in FE code* and coordinate (in `exportEntities↩NodesCoordinates.py`)

```
1 # Export nodes coordinates from a model in history
2 #
3 # arguments:
4 #   /path/to/export/directory [history name]
5
6 import os
7 import os.path
8 import shutil
9 import exceptions
10
11 import piper.hbm
12 import piper.app
13
14 model = piper.app.Context.instance().project().model()
15
16 # if the model is empty, we can do nothing
17 if model.empty():
18     piper.app.logWarning("Model is empty")
19 elif not ( len(sys.argv) in {2,3} ) :
20     raise RuntimeError("Invalid arguments")
21 else :
22     exportDirectory = sys.argv[1]
23
24     # setup model in history
25     originalHistory = None
26     if len(sys.argv) == 3:
27         originalHistory = model.history().getCurrent()
28         model.history().setCurrentModel(sys.argv[2])
29
30     shutil.rmtree(exportDirectory, ignore_errors=True)
31     os.makedirs(exportDirectory)
32     metadata = model.metadata()
33     fem = model.fem()
34     piperIdToIdKey = fem.computePiperIdToIdKeyMapNode();
35     for (name,entity) in metadata.entities().iteritems() :
36         ids = entity.getEntityNodesIds(fem)
37         datFilePath = os.path.join(exportDirectory, name+".dat")
38         piper.app.logInfo("Export {0} nodes from entity {1} to {2}".format(len(ids), name, datFilePath))
39         with open(datFilePath, 'w') as datFile:
40             datFile.write("# Entity {0}\n".format(name))
41             datFile.write("# Id x y z\n")
42             for nodeId in ids:
43                 coord = fem.getNode(nodeId).get().flatten()
44                 datFile.write("{0} {1} {2} {3}\n".format(piperIdToIdKey[nodeId].id, coord[0], coord[1],
    coord[2]))
45
46     # restore model in history
47     if not originalHistory is None:
48         model.history().setCurrentModel(originalHistory)
```

TODO: examples with target creation, TODO: example with data preparation saved to (octave) file, running octave, reading output and preparing targets TODO: doc for piper.app, piper.hbm and piper.anatomydb python packages

8.4 Batch mode

The *batch mode* enable the user to prepare *scripts* to run at once a sequence of operations. The functions available in batch mode includes Project read/write, import/export, target read/write and modification, module parameters modification and access to PIPER Modules Overview.

The batch mode is started with:

```
$ piper --batch my_script.py
```

More options are available, refer to "`piper --help`" output.

8.4.1 Examples

8.4.1.1 Target creation

```
1 import math
2
3 import piper.app
4 import piper.hbm
5 import piper.test
6
7 target = piper.hbm.TargetList()
8
9 target.add(piper.hbm.FixedBoneTarget("Pelvic_skeleton"))
10
11 leftASISTarget = piper.hbm.LandmarkTarget()
12 leftASISTarget.setName("Pelvis_position_1")
13 leftASISTarget.setUnit(piper.hbm.Length_m)
14 leftASISTarget.landmark = "Left_ASIS"
15 leftASISTarget.setValue({0:0.3, 1:1.2, 2:-0.25})
16 target.add(leftASISTarget)
17
18 piper.test.expect_eq(piper.hbm.Length_m, leftASISTarget.unit(), "Error during target creation")
19
20 hipTarget = piper.hbm.JointTarget()
21 hipTarget.setName("Hip_position")
22 hipTarget.setUnit(piper.hbm.Length_dm)
23 hipTarget.joint="Left_hip_joint"
24 hipTarget.setValue({4:math.radians(45)})
25 target.add(hipTarget)
26
27 piper.test.expect_eq(piper.hbm.Length_dm, hipTarget.unit(), "Error during target creation")
28
29 headTarget = piper.hbm.FrameToFrameTarget()
30 headTarget.setName("Head_position")
31 headTarget.frameSource = "W_Origin"
32 headTarget.frameTarget = "B_Skull"
33 headTarget.setUnit(piper.hbm.Length_mm)
34 headTarget.setValue({0:200})
35 target.add(headTarget)
36
37 piper.test.expect_eq(4, target.size(), "Error during target creation")
```

8.4.1.2 Physics based positioning

```
1 import os.path
2 import math
3
4 import piper.app
5 import piper.hbm
6
7 project = piper.app.Project()
8
9 # save Child
10 chilpProjectFile = os.path.join(piper.app.tempDirectoryPath(), "child.ppj")
11 #chilpProjectFile = os.path.join("/local2/build/piper-release/child.ppj")
12 project.read(chilpProjectFile)
13
14 project.moduleParameter().setInt("PhysPosi","autoStopNbIterationMax", 100)
15 project.moduleParameter().setFloat("PhysPosi", "autoStopVelocity", 1e-7)
16 project.moduleParameter().setFloat("PhysPosi","voxelSize", 0.01)
```

```
17
18 target = piper.hbm.TargetList()
19
20 target.add(piper.hbm.FixedBoneTarget("Pelvic_skeleton"))
21
22 hipTarget = piper.hbm.JointTarget()
23 hipTarget.setName("Hip_position")
24 hipTarget.joint="Left_hip_joint"
25 hipTarget.setValue({4:math.radians(45)})
26 target.add(hipTarget)
27
28 headTarget = piper.hbm.FrameToFrameTarget()
29 headTarget.setName("Head_position")
30 headTarget.setUnit(piper.hbm.Length_mm)
31 headTarget.frameSource = "W_Origin"
32 headTarget.frameTarget = "B_Skull"
33 headTarget.setValue({0:200})
34 target.add(headTarget)
35
36 piper.app.physPosiDeform(project, target)
```

8.5 PIPER Python packages

The PIPER specific functions are made available in batch thanks to several Python packages :

- application Context, Project manipulation, logging functions and more: Python package piper.app
- access HBM data, Metadata, Target and more: Python package piper.hbm
- access anatomy database queries and anatomical frames computation: Python package piper.anatomyDB

9 Known Limitations

9.1 Application

- There is a single list of targets in the application

9.2 Modules

9.2.1 Pre Position

- The gui of the joint controller only offers to control rotational degrees of freedom, no translation.
- There is no way to save spine spline positioning target
- There are no constraints on the volume conservation
- Make some processes parallel (initialization, collisions, ...)
- Store the target stiffness in the target file to facilitate repetitive tasks

9.2.2 Anthropometry personalization

- For the population definition, make sure that if you defined a criteria for a bin, to define the same criteria for the rest of the bins.
- While defining information for the population definition, make sure the informations concerning the different bins are differentiable. For exemple if you define two bins, with the same informations: 50% population of males both, it will generate a script execution error.
- There is not output indicating how plausible the selection of predictor (w.r.t. the current database used). Unrealistic input may lead to unrealistic prediction without warning.

9.2.3 Metadata editor

- Created joints are not appended to the existing ones
- Currently cannot edit existing entities, only create new ones

9.2.4 Contour Deformation Positioning

- There are limits of the angles by which each joint in the full GHBMC model is positioned. The restriction is because of the current classification of Skin Entities in the model.
- Currently positioning at Shoulder Joint and Thorax region is not included in this release.

9.2.5 Contour Deformation GEBOD Personalization

- Please select the parameter values such that they lie above the 5th percentile values of the human body model.
  Please generate the the GEBOD target values again from the Anthropometry module.
- Only those ANSUR target parameters which correspond to 35 GEBOD parameters used to personalize the full GHBMC Model.

9.2.6 Anatomy DB

- classes should be implemented to facilitate checking the landmark consistency (are points symmetric, distal, cranial, medial).
- documentation still needs to be updated

9.3 Perspectives/Ideas

- a new module for motion learning / motion library (able to interpolate/cache common motion)

10 Support and Contact

- If you find a bug, please submit a new report here

11 Installation

Windows

The provided software archive contains all the necessary dependencies. It is a windows 64 version (tested on windows 7 and 10). It can be started using “Runpiper.bat”. No installation or administrative rights are needed.

Linux

On linux we rely on the distribution package management system. To run the PIPER application, the required packages are:

```
python2.7, python-numpy, libxml2-utils, freeglut3
```

Some functionnalities rely on octave scripts and a few octave toolboxes, to be able to use them, install the additionnal packages:

```
octave, liboctave-dev, unzip
```

And then install the octave packages, start octave and proceed with:

```
$ octave-cli
octave:1> pkg install -forge io
octave:2> pkg install -forge statistics
octave:3> pkg list
Package Name  | Version | Installation directory
--------------+---------+-----------------------
          io  |   2.4.2 | /home/thomas/octave/io-2.4.2
  statistics  |   1.2.4 | /home/thomas/octave/statistics-1.2.4
```

12 ChangeLog

v1.0.0 - 05/2017

- [pre-posi] some gui enhancement (landmark visibility, reload button always active)
- [pre-posi] fix controllers were not destroyed on reload
- [fine-posi] fix bad affine density value used
- [shape] fix skin young modules value was not set properly from the gui
- [kriging] support for domain decomposition of the model through named metadata

v0.99.0 - 04/2017

- [shape] new module to shape HBM skin (beta quality)
- [pre-posi] spine predictor have two lateral flexion angles
- [pre-posi] voxel size can be scaled with HBM height
- [pre-posi] custom affine frames can be defined in the metadata
- [pre-posi] add simulation of articular capsules, ligaments, cartilage, when appropriate entities are defined
- [pre-posi] add a distance constraint between tibia and patella
- [pre-posi] add relative motion in the frame controller
- [pre-posi] bone and capsule collisions can be enabled/disabled directly in the module, disabling it speeds up the simulation
- [application] ability to interrupt lengthy process (like Position modules loading)
- [anatomyDB] add some articular capsules, ligaments and meniscus
- [pre-posi] FIX wrong bone collision in rare cases
- [pre-posi] FIX update of some nodes in bone entities
- [application] Environment can now be visualized and transformed (scale, translation and rotation)
- [framework] metadata can be exported and imported.
- [framework] only pmr file is now required to imported HBM from FE file or graphic mesh format (obj).
- [bodysection] refactoring module: anthropometric model can be edited. Handles of type of dimension with relative or absolute taget value
- [kriging] skin transformation now possible using surface distance instead of euclidean
- [kriging] kriging with intermediate bone and skin target

v0.8.0 - 01/2017

- [pre-posi] improve stability of the spine controller and the spine predictor
- [pre-posi] add control on the user target stiffness
- [pre-posi] add a clipping plane to inspect the 3D model
- [pre-posi] fix flesh attachment bug
- [framework] add generic tooltip
- [framework] the application remembers your last used folders
- [framework] python scripts can be run at any time in the workflow

v0.7.0 - 11/2016

- [application] add landmarks and nodes exporter to simple ascii files
- [pre-posi] lazy simulation loading, do not reload simulation if it is already ready
- [pre-posi] updated spine predictor, with or without pelvis orientation
- [pre-posi] spine target is reached progressively to improve simulation stability
- [pre-posi] let the user control more simulation parameters (timestep, number of vertices for bone collision)
- [smoothing] allowed selecting multiple entities as targets for optimization by all smoothing techniques
- [3D viewer] Element blanking to allow exploration of inside of the model
- [3D viewer] Display settings: setting camera focal point, resetting camera, switching projection type
- [3D viewer] Highlighting element edges and normals
- [3D viewer] Camera and display settings are now persistent across all modules
- [3D viewer] Improved stability (due to safer handling of interaction between "computational" and rendering threads)

v0.6 - 10/2016 - alpha version

- [pre-posi] new limited mouse interaction with the model
- [pre-posi] new spine predictor
- [pre-posi] new spine controller
- [pre-posi] new anatomical landmark controllers
- [fine-posi] slightly better affine frame spreading in the model
- [pre-posi] soft and hard joint simulation
- [pre-posi] loading of targets
- [pre-posi] bone collision
- [pre-posi] automatic stop of the positioning process
- [app] model-modifying actions now create a history of node coordinates to which the user can roll-back
- [bodysection] new module for scaling the model based on body section dimension targets
- [anthropo-perso] new module for generating anthropometric measurement targets based on statistical regression analysis of a specified database

- [smoothing] crease detection on surfaces
- [smoothing] transformation smoothing using kriging in a box
- [smoothing] surface smoothing using windowed sinc FIR filter
- [smoothing] mechanism for loading baseline models either from file or from history
- [3D viewer] VTK-based 3D viewer - model exploration, visualization of metadata
- [3D viewer] Picking tools: rubber band, box and single-object picking
- [3D viewer] Coloring elements based on quality
- [app] uniform module layout, use of tool windows to gather specific parts of the GUI

v0.5 - 12/2015

- [smoothing] new smoothing module based on the Mesquite library
- [fine-posi] new physics based module for positioned module deformation, it is based on the [pre-posi]
- [pre-posi] control over frame orientation, ability to remove controllers
- [pre-posi] save some targets
- [laplacian-smoothing] this module is removed
- [app] modules can have parameters
- [framework] target data structure as an input/output for modules, can be saved/loaded to/from an xml file
- [framework] environment can be loaded from FE code files
- [framework] piper application save/load project files which contain the FEModel, targets and environment.
- [iitd] new module to define contours metadata and position hip and knee, personalize the hbm.
- [mesh-optim] new module to analyze mesh quality and improve mesh quality for 3D elements meshes.

v0.4 - 09/2015

- [pre-posi] positioning of the frames, more display options
- [pre-posi] support contact
- [libhbm] support contact
- [libhbm] split parser files in a format specific file and a model specific file

v0.3 - 07/2015

- same functionalities as 0.2 but using double precision for floating point numbers: it consumes more memory but sofa-positioning is more stable.

v0.2 - 07/2015

- package for both windows and linux
- new module physics-positioning

v0.1 - 05/2015

- initial version of the PIPER application for windows
- it includes the following modules: check, 3d-display, krigging, sofa-smoothing, scaling parameter

13 License

PIPER software framework and application

This work has received funding from the European Union Seventh Framework Program ([FP7/2007-2013]) under grant agreement 605544 [PIPER project]).

PIPER documentation

PIPER data

14 Acknowledgements

The PIPER application makes use of the following external libraries and tools :

- boost
- python
- numpy
- qt
- SOFA
- VTK
- eigen
- Spectra
- mesquite

- tetgen
- TinySpline
- TinyXML
- SWIG
- SQLite
- xmllint
- Google Test
- Doxygen
- LaTeX

15 Appendix: List of entities

Entity name	Synonyms	Classes	PartOf	Region
Skin		Skin		Body
Skin_of_abdomen		Skin	Skin_of_trunk	Trunk
Skin_of_body_↩ proper		Skin	Skin	Body_proper
Skin_of_head		Skin	Skin	Head
Skin_of_left_arm		Skin	Skin_of_left_free↩ _upper_limb	Left_arm
Skin_of_left_foot		Skin	Skin_of_left_free↩ _lower_limb	Left_foot
Skin_of_left_↩ forearm		Skin	Skin_of_left_free↩ _upper_limb	Left_forearm
Skin_of_left_free↩ _lower_limb		Skin	Skin_of_left_↩ lower_limb	Left_free_lower_↩ limb
Skin_of_left_free↩ _upper_limb		Skin	Skin_of_left_↩ upper_limb	Left_free_upper_↩ limb
Skin_of_left_hand		Skin	Skin_of_left_free↩ _upper_limb	Left_hand
Skin_of_left_leg		Skin	Skin_of_left_free↩ _lower_limb	Left_leg
Skin_of_left_↩ lower_limb		Skin	Skin	Left_lower_limb
Skin_of_left_↩ pectoral_girdle		Skin	Skin_of_left_↩ upper_limb	Left_pectoral_↩ girdle
Skin_of_left_↩ pelvic_girdle		Skin	Skin_of_left_↩ lower_limb	Left_pelvic_girdle
Skin_of_left_thigh		Skin	Skin_of_left_free↩ _lower_limb	Left_thigh
Skin_of_left_↩ upper_limb		Skin	Skin	Left_upper_limb
Skin_of_neck		Skin	Skin_of_body_↩ proper	Neck
Skin_of_right_arm		Skin	Skin_of_right_↩ free_upper_limb	Right_arm

Skin_of_right_foot		Skin	Skin_of_right_↩ free_lower_limb	Right_foot
Skin_of_right_↩ forearm		Skin	Skin_of_right_↩ free_upper_limb	Right_forearm
Skin_of_right_↩ free_lower_limb		Skin	Skin_of_right_↩ lower_limb	Right_free_lower↩ _limb
Skin_of_right_↩ free_upper_limb		Skin	Skin_of_right_↩ upper_limb	Right_free_upper↩ _limb
Skin_of_right_hand		Skin	Skin_of_right_↩ free_upper_limb	Right_arm
Skin_of_right_leg		Skin	Skin_of_right_↩ free_lower_limb	Right_leg
Skin_of_right_↩ lower_limb		Skin	Skin	Right_lower_limb
Skin_of_right_↩ pectoral_girdle		Skin	Skin_of_right_↩ upper_limb	Right_pectoral_↩ girdle
Skin_of_right_↩ pelvic_girdle		Skin	Skin_of_right_↩ lower_limb	Right_pelvic_girdle
Skin_of_right_thigh		Skin	Skin_of_right_↩ free_lower_limb	Right_thigh
Skin_of_right_↩ upper_limb		Skin	Skin	Right_upper_limb
Skin_of_thorax		Skin	Skin_of_trunk	Trunk
Skin_of_trunk		Skin	Skin_of_body_↩ proper	Trunk
Atlas	C1	Bone	Vertebral_column	Neck
Atypical_rib		Bone	Skeleton_of_rib_↩ cage	
Auditory_ossicle		Bone	Skull	
Axis	C2	Bone	Vertebral_column	Neck
Calcaneus		Bone		
Capitate		Bone		
Carpal_bone		Bone		
Cervical_rib		Bone	Skeleton_of_rib_↩ cage	
Cervical_vertebra		Bone	Vertebral_column	
Clavicle		Bone		
Coccyx		Bone	Pelvic_skeleton	
Cuboid_bone		Bone		
Cuneiform_bone		Bone		
Distal_carpal_bone		Bone		
Distal_phalanx_↩ of_big_toe		Bone		
Distal_phalanx_↩ of_fourth_toe		Bone		
Distal_phalanx_↩ of_index_finger		Bone		
Distal_phalanx_↩ of_left_big_toe		Bone		
Distal_phalanx_↩ of_left_fourth_toe		Bone		

Distal_phalanx_↩ of_left_index_finger		Bone		
Distal_phalanx_↩ of_left_little_finger		Bone		
Distal_phalanx_↩ of_left_little_toe		Bone		
Distal_phalanx_↩ of_left_middle_↩ finger		Bone		
Distal_phalanx_↩ of_left_ring_finger		Bone		
Distal_phalanx_↩ of_left_second_toe		Bone		
Distal_phalanx_↩ of_left_third_toe		Bone		
Distal_phalanx_↩ of_left_thumb		Bone		
Distal_phalanx_↩ of_little_finger		Bone		
Distal_phalanx_↩ of_little_toe		Bone		
Distal_phalanx_↩ of_middle_finger		Bone		
Distal_phalanx_↩ of_right_big_toe		Bone		
Distal_phalanx_↩ of_right_fourth_toe		Bone		
Distal_phalanx_↩ of_right_index_↩ finger		Bone		
Distal_phalanx_↩ of_right_little_↩ finger		Bone		
Distal_phalanx_↩ of_right_little_toe		Bone		
Distal_phalanx_↩ of_right_middle_↩ finger		Bone		
Distal_phalanx_↩ of_right_ring_finger		Bone		
Distal_phalanx_↩ of_right_second_↩ toe		Bone		
Distal_phalanx_↩ of_right_third_toe		Bone		
Distal_phalanx_↩ of_right_thumb		Bone		
Distal_phalanx_↩ of_ring_finger		Bone		
Distal_phalanx_↩ of_second_toe		Bone		
Distal_phalanx_↩ of_third_toe		Bone		

Distal_phalanx_↩ of_thumb		Bone		
Eighth_rib		Bone	Skeleton_of_rib_↩ cage	
Eighth_thoracic_↩ vertebra	T8, T08	Bone	Vertebral_column	Trunk
Eleventh_rib		Bone	Skeleton_of_rib_↩ cage	
Eleventh_↩ thoracic_vertebra	T11	Bone	Vertebral_column	Trunk
Epipteric_bone		Bone		
Ethmoid		Bone	Skull	
False_rib		Bone	Skeleton_of_rib_↩ cage	
Femur		Bone	Skeleton_of_free↩ _lower_limb	
Fibula		Bone	Skeleton_of_free↩ _lower_limb	
Fifth_cervical_↩ vertebra	C5	Bone	Vertebral_column	Neck
Fifth_lumbar_↩ vertebra	L5	Bone	Vertebral_column	Trunk
Fifth_metacarpal↩ _bone		Bone		
Fifth_metatarsal_↩ bone		Bone		
Fifth_rib		Bone	Skeleton_of_rib_↩ cage	
Fifth_thoracic_↩ vertebra	T5, T05	Bone	Vertebral_column	Trunk
First_lumbar_↩ vertebra	L1	Bone	Vertebral_column	Trunk
First_metacarpal↩ _bone		Bone		
First_metatarsal_↩ bone		Bone		
First_rib		Bone	Skeleton_of_rib_↩ cage	
First_thoracic_↩ vertebra	T1, T01	Bone	Vertebral_column	Trunk
Floating_rib		Bone	Skeleton_of_rib_↩ cage	
Fourth_cervical_↩ vertebra	C4	Bone	Vertebral_column	Neck
Fourth_lumbar_↩ vertebra	L4	Bone	Vertebral_column	Trunk
Fourth_↩ metacarpal_bone		Bone		
Fourth_↩ metatarsal_bone		Bone		
Fourth_rib		Bone	Skeleton_of_rib_↩ cage	

Fourth_thoracic_↩ vertebra	T4, T04	Bone	Vertebral_column	Trunk
Frontal_bone		Bone	Skull	
Hamate		Bone		
Hip_bone		Bone	Skeleton_of_↩ pelvic_girdle	
Humerus		Bone	Skeleton_of_↩ upper_limb	
Hyoid_bone		Bone		Head
Incus		Bone		
Inferior_nasal_↩ concha		Bone		
Intermediate_↩ cuneiform_bone		Bone		
Interparietal_bone		Bone		
Lacrimal_bone		Bone	Skull	Head
Lateral_↩ cuneiform_bone		Bone		
Left_calcaneus		Bone	Skeleton_of_left_↩ foot	
Left_capitate		Bone		
Left_clavicle	Clavicle_L	Bone		Left_pectoral_↩ girdle
Left_cuboid_bone		Bone		Left_free_lower_↩ limb
Left_eighth_rib	Rib_r08L	Bone	Skeleton_of_rib_↩ cage	
Left_eleventh_rib	Rib_r11L	Bone	Skeleton_of_rib_↩ cage	
Left_femur	Femur_L	Bone		Left_thigh
Left_fibula	Fibula_L	Bone	Skeleton_of_left_↩ leg	Left_leg
Left_fifth_↩ metacarpal_bone		Bone		
Left_fifth_↩ metatarsal_bone		Bone		Left_foot
Left_fifth_rib	Rib_r05L	Bone	Skeleton_of_rib_↩ cage	
Left_first_↩ metacarpal_bone		Bone		
Left_first_↩ metatarsal_bone		Bone		Left_foot
Left_first_rib	Rib_r01L	Bone	Skeleton_of_rib_↩ cage	
Left_fourth_↩ metacarpal_bone		Bone		
Left_fourth_↩ metatarsal_bone		Bone		Left_foot
Left_fourth_rib	Rib_r04L	Bone	Skeleton_of_rib_↩ cage	
Left_hamate		Bone		
Left_hip_bone		Bone	Pelvic_skeleton	

Left_humerus	Humerus_L	Bone		Left_arm
Left_incus		Bone		Head
Left_inferior_↩ nasal_concha		Bone		Head
Left_↩ intermediate_↩ cuneiform_bone		Bone		Left_foot
Left_lacrimal_bone		Bone		Head
Left_lateral_↩ cuneiform_bone		Bone		Left_foot
Left_lunate		Bone		
Left_malleus		Bone		Head
Left_maxilla		Bone		Head
Left_medial_↩ cuneiform_bone		Bone		Left_foot
Left_nasal_bone		Bone		Head
Left_ninth_rib	Rib_r09L	Bone	Skeleton_of_rib_↩ cage	
Left_palatine_bone		Bone		Head
Left_parietal_bone		Bone		Head
Left_patella	Patella_L	Bone		Left_free_lower_↩ limb
Left_pisiform		Bone		
Left_radius	Radius_L	Bone	Skeleton_of_left_↩ forearm	Left_forearm
Left_scaphoid		Bone		
Left_scapula	Scapula_L	Bone		Left_pectoral_↩ girdle
Left_second_↩ metacarpal_bone		Bone		
Left_second_↩ metatarsal_bone		Bone		Left_foot
Left_second_rib	Rib_r02L	Bone	Skeleton_of_rib_↩ cage	
Left_seventh_rib	Rib_r07L	Bone	Skeleton_of_rib_↩ cage	
Left_sixth_rib	Rib_r06L	Bone	Skeleton_of_rib_↩ cage	
Left_stapes		Bone		Head
Left_talus		Bone		Left_foot
Left_temporal_↩ bone		Bone		Head
Left_tenth_rib	Rib_r10L	Bone	Skeleton_of_rib_↩ cage	
Left_third_↩ metacarpal_bone		Bone		
Left_third_↩ metatarsal_bone		Bone		Left_foot
Left_third_rib	Rib_r03L	Bone	Skeleton_of_rib_↩ cage	

Left_tibia	Tibia_L	Bone	Skeleton_of_left_↩ leg	Left_leg
Left_trapezium		Bone		
Left_trapezoid		Bone		
Left_triquetral		Bone		
Left_twelfth_rib	Rib_r12L	Bone	Skeleton_of_rib_↩ cage	
Left_ulna	Ulna_L	Bone	Skeleton_of_left_↩ forearm	Left_forearm
Left_zygomatic_↩ bone		Bone		Head
Lumbar_rib		Bone	Skeleton_of_rib_↩ cage	
Lumbar_vertebra		Bone	Vertebral_column	
Lunate		Bone		
Malleus		Bone		Head
Mandible		Bone	Skull	
Maxilla		Bone	Skull	
Medial_↩ cuneiform_bone		Bone		
Metacarpal_bone		Bone		
Metatarsal_bone		Bone	Set_of_↩ metatarsal_bones	
Middle_phalanx_↩ of_fourth_toe		Bone		
Middle_phalanx_↩ of_index_finger		Bone		
Middle_phalanx_↩ of_left_fourth_toe		Bone		Left_foot
Middle_phalanx_↩ of_left_index_finger		Bone		
Middle_phalanx_↩ of_left_little_finger		Bone		
Middle_phalanx_↩ of_left_little_toe		Bone		Left_foot
Middle_phalanx_↩ of_left_middle_↩ finger		Bone		
Middle_phalanx_↩ of_left_ring_finger		Bone		
Middle_phalanx_↩ of_left_second_toe		Bone		Left_foot
Middle_phalanx_↩ of_left_third_toe		Bone		Left_foot
Middle_phalanx_↩ of_little_finger		Bone		
Middle_phalanx_↩ of_little_toe		Bone		
Middle_phalanx_↩ of_middle_finger		Bone		
Middle_phalanx_↩ of_right_fourth_toe		Bone		Right_foot

Middle_phalanx_↩ of_right_index_↩ finger		Bone		
Middle_phalanx_↩ of_right_little_↩ finger		Bone		
Middle_phalanx_↩ of_right_little_toe		Bone		Right_foot
Middle_phalanx_↩ of_right_middle_↩ finger		Bone		
Middle_phalanx_↩ of_right_ring_finger		Bone		
Middle_phalanx_↩ of_right_second_↩ toe		Bone		Right_foot
Middle_phalanx_↩ of_right_third_toe		Bone		Right_foot
Middle_phalanx_↩ of_ring_finger		Bone		
Middle_phalanx_↩ of_second_toe		Bone		
Middle_phalanx_↩ of_third_toe		Bone		
Nasal_bone		Bone	Skull	
Navicular_bone_↩ of_foot		Bone		
Navicular_bone_↩ of_left_foot		Bone		Left_foot
Navicular_bone_↩ of_right_foot		Bone		Right_foot
Ninth_rib		Bone	Skeleton_of_rib_↩ cage	
Ninth_thoracic_↩ vertebra	T9, T09	Bone	Vertebral_column	Trunk
Occipital_bone		Bone	Skull	
Palatine_bone		Bone	Skull	
Parietal_bone		Bone	Skull	
Patella		Bone	Skeleton_of_free↩ _lower_limb	
Pelvic_skeleton		Bone		Trunk
Phalanx_of_big_↩ toe		Bone		
Phalanx_of_finger		Bone		
Phalanx_of_↩ fourth_toe		Bone		
Phalanx_of_↩ index_finger		Bone		
Phalanx_of_little↩ _finger		Bone		
Phalanx_of_little↩ _toe		Bone		

Phalanx_of_middle_finger		Bone		
Phalanx_of_ring_finger		Bone		
Phalanx_of_second_toe		Bone		
Phalanx_of_third_toe		Bone		
Phalanx_of_thumb		Bone		
Phalanx_of_toe		Bone	Skeleton_of_foot	
Pisiform		Bone		
Preinterparietal_bone		Bone		
Proximal_carpal_bone		Bone		
Proximal_phalanx_of_big_toe		Bone		
Proximal_phalanx_of_fourth_toe		Bone		
Proximal_phalanx_of_index_finger		Bone		
Proximal_phalanx_of_left_big_toe		Bone		Left_foot
Proximal_phalanx_of_left_fourth_toe		Bone		Left_foot
Proximal_phalanx_of_left_index_finger		Bone		
Proximal_phalanx_of_left_little_finger		Bone		
Proximal_phalanx_of_left_little_toe		Bone		Left_foot
Proximal_phalanx_of_left_middle_finger		Bone		
Proximal_phalanx_of_left_ring_finger		Bone		
Proximal_phalanx_of_left_second_toe		Bone		Left_foot
Proximal_phalanx_of_left_third_toe		Bone		Left_foot

Proximal_↩ phalanx_of_left_↩ thumb		Bone		
Proximal_↩ phalanx_of_little↩ _finger		Bone		
Proximal_↩ phalanx_of_little↩ _toe		Bone		
Proximal_↩ phalanx_of_↩ middle_finger		Bone		
Proximal_↩ phalanx_of_right↩ _big_toe		Bone		Right_foot
Proximal_↩ phalanx_of_right↩ _fourth_toe		Bone		Right_foot
Proximal_↩ phalanx_of_right↩ _index_finger		Bone		
Proximal_↩ phalanx_of_right↩ _little_finger		Bone		
Proximal_↩ phalanx_of_right↩ _little_toe		Bone		Right_foot
Proximal_↩ phalanx_of_right↩ _middle_finger		Bone		
Proximal_↩ phalanx_of_right↩ _ring_finger		Bone		
Proximal_↩ phalanx_of_right↩ _second_toe		Bone		Right_foot
Proximal_↩ phalanx_of_right↩ _third_toe		Bone		Right_foot
Proximal_↩ phalanx_of_right↩ _thumb		Bone		
Proximal_↩ phalanx_of_ring_↩ finger		Bone		
Proximal_↩ phalanx_of_↩ second_toe		Bone		
Proximal_↩ phalanx_of_third↩ _toe		Bone		

Proximal_↩ phalanx_of_thumb		Bone		
Radius		Bone		
Rib		Bone		
Right_calcaneus		Bone	Skeleton_of_↩ right_foot	
Right_capitate		Bone		
Right_clavicle	Clavicle_R	Bone		Right_pectoral_↩ girdle
Right_cuboid_bone		Bone		Right_foot
Right_eighth_rib	Rib_r08R	Bone	Skeleton_of_rib_↩ cage	
Right_eleventh_rib	Rib_r11R	Bone	Skeleton_of_rib_↩ cage	
Right_femur	Femur_R	Bone		Right_thigh
Right_fibula	Fibula_R	Bone	Skeleton_of_↩ right_leg	Right_leg
Right_fifth_↩ metacarpal_bone		Bone		
Right_fifth_↩ metatarsal_bone		Bone		Right_foot
Right_fifth_rib	Rib_r05R	Bone	Skeleton_of_rib_↩ cage	
Right_first_↩ metacarpal_bone		Bone		
Right_first_↩ metatarsal_bone		Bone		Right_foot
Right_first_rib	Rib_r01R	Bone	Skeleton_of_rib_↩ cage	
Right_fourth_↩ metacarpal_bone		Bone		
Right_fourth_↩ metatarsal_bone		Bone		Right_foot
Right_fourth_rib	Rib_r04R	Bone	Skeleton_of_rib_↩ cage	
Right_hamate		Bone		
Right_hip_bone		Bone	Pelvic_skeleton	
Right_humerus	Humerus_R	Bone		Right_arm
Right_incus		Bone		Head
Right_inferior_↩ nasal_concha		Bone		Head
Right_↩ intermediate_↩ cuneiform_bone		Bone		Right_foot
Right_lacrimal_↩ bone		Bone		Head
Right_lateral_↩ cuneiform_bone		Bone		Right_foot
Right_lunate		Bone		
Right_malleus		Bone		Head
Right_maxilla		Bone		Head

Right_medial_↩ cuneiform_bone		Bone		Right_foot
Right_nasal_bone		Bone		Head
Right_ninth_rib	Rib_r09R	Bone	Skeleton_of_rib_↩ cage	
Right_palatine_↩ bone		Bone		Head
Right_parietal_↩ bone		Bone		Head
Right_patella	Patella_R	Bone		Right_free_lower↩ _limb
Right_pisiform		Bone		
Right_radius	Radius_R	Bone	Skeleton_of_↩ right_forearm	Right_forearm
Right_scaphoid		Bone		
Right_scapula	Scapula_R	Bone		Right_pectoral_↩ girdle
Right_second_↩ metacarpal_bone		Bone		
Right_second_↩ metatarsal_bone		Bone		Right_foot
Right_second_rib	Rib_r02R	Bone	Skeleton_of_rib_↩ cage	
Right_seventh_rib	Rib_r07R	Bone	Skeleton_of_rib_↩ cage	
Right_sixth_rib	Rib_r06R	Bone	Skeleton_of_rib_↩ cage	
Right_stapes		Bone		Head
Right_talus		Bone		Right_foot
Right_temporal_↩ bone		Bone		Head
Right_tenth_rib	Rib_r10R	Bone	Skeleton_of_rib_↩ cage	
Right_third_↩ metacarpal_bone		Bone		
Right_third_↩ metatarsal_bone		Bone		Right_foot
Right_third_rib	Rib_r03R	Bone	Skeleton_of_rib_↩ cage	
Right_tibia	Tibia_R	Bone	Skeleton_of_↩ right_leg	Right_leg
Right_trapezium		Bone		
Right_trapezoid		Bone		
Right_triquetral		Bone		
Right_twelfth_rib	Rib_r12R	Bone	Skeleton_of_rib_↩ cage	
Right_ulna	Ulna_R	Bone	Skeleton_of_↩ right_forearm	Right_forearm
Right_zygomatic↩ _bone		Bone		Head
Sacrum		Bone	Pelvic_skeleton	
Scaphoid		Bone		

Scapula		Bone		
Second_lumbar_↩ vertebra	L2	Bone	Vertebral_column	Trunk
Second_↩ metacarpal_bone		Bone		
Second_↩ metatarsal_bone		Bone		
Second_rib		Bone	Skeleton_of_rib_↩ cage	
Second_thoracic↩ _vertebra	T2, T02	Bone	Vertebral_column	Trunk
Sesamoid_bone		Bone		
Sesamoid_bone↩ _of_foot		Bone	Skeleton_of_foot	
Sesamoid_bone↩ _of_hand		Bone		
Sesamoid_bone↩ _of_left_foot		Bone		Left_foot
Sesamoid_bone↩ _of_left_hand		Bone		
Sesamoid_bone↩ _of_right_foot		Bone		Right_foot
Sesamoid_bone↩ _of_right_hand		Bone		
Set_of_left_↩ metatarsal_bones		Bone		Left_foot
Set_of_left_↩ tarsal_bones		Bone		Left_foot
Set_of_↩ metatarsal_bones		Bone		
Set_of_↩ phalanges_of_↩ big_toe		Bone		
Set_of_↩ phalanges_of_↩ fourth_toe		Bone		
Set_of_↩ phalanges_of_↩ left_big_toe		Bone		Left_foot
Set_of_↩ phalanges_of_↩ left_fourth_toe		Bone		Left_foot
Set_of_↩ phalanges_of_↩ left_little_toe		Bone		Left_foot
Set_of_↩ phalanges_of_↩ left_second_toe		Bone		Left_foot
Set_of_↩ phalanges_of_↩ left_third_toe		Bone		Left_foot

Set_of_↩ phalanges_of_↩ little_toe		Bone		
Set_of_↩ phalanges_of_↩ right_big_toe		Bone		Right_foot
Set_of_↩ phalanges_of_↩ right_fourth_toe		Bone		Right_foot
Set_of_↩ phalanges_of_↩ right_little_toe		Bone		Right_foot
Set_of_↩ phalanges_of_↩ right_second_toe		Bone		Right_foot
Set_of_↩ phalanges_of_↩ right_third_toe		Bone		Right_foot
Set_of_↩ phalanges_of_↩ second_toe		Bone		
Set_of_↩ phalanges_of_↩ third_toe		Bone		
Set_of_↩ phalanges_of_toe		Bone		
Set_of_right_↩ metatarsal_bones		Bone		Right_foot
Set_of_right_↩ tarsal_bones		Bone		Right_foot
Set_of_tarsal_↩ bones		Bone		
Seventh_cervical↩ _vertebra	C7	Bone	Vertebral_column	Neck
Seventh_rib		Bone	Skeleton_of_rib_↩ cage	
Seventh_thoracic↩ _vertebra	T7, T07	Bone	Vertebral_column	Trunk
Sixth_cervical_↩ vertebra	C6	Bone	Vertebral_column	Neck
Sixth_rib		Bone	Skeleton_of_rib_↩ cage	
Sixth_thoracic_↩ vertebra	T6, T06	Bone	Vertebral_column	Trunk
Skeleton		Bone		Body
Skeleton_of_foot		Bone		
Skeleton_of_free↩ _lower_limb		Bone		
Skeleton_of_left_↩ foot	Left_foot_skeleton	Bone		Left_foot
Skeleton_of_left_↩ forearm	Left_forearm_↩ skeleton, Forearm_L	Bone		Left_forearm

Skeleton_of_left_↩ free_lower_limb		Bone		Left_free_lower_↩ limb
Skeleton_of_left_↩ hand	Left_hand_skeleton	Bone		Left_hand
Skeleton_of_left_↩ leg		Bone		Left_leg
Skeleton_of_left_↩ lower_limb		Bone		Left_lower_limb
Skeleton_of_left_↩ pelvic_girdle		Bone		Left_pelvic_girdle
Skeleton_of_left_↩ upper_limb		Bone		Left_upper_limb
Skeleton_of_↩ lower_limb		Bone		
Skeleton_of_↩ pelvic_girdle		Bone		
Skeleton_of_rib_↩ cage		Bone		Trunk
Skeleton_of_↩ right_foot	Right_foot_↩ skeleton	Bone		Right_foot
Skeleton_of_↩ right_forearm	Right_forearm_↩ skeleton, Forearm_R	Bone		Right_forearm
Skeleton_of_↩ right_free_lower_↩ limb		Bone		Right_free_lower↩ _limb
Skeleton_of_↩ right_hand	Right_hand_↩ skeleton	Bone		Right_hand
Skeleton_of_↩ right_leg		Bone		Right_leg
Skeleton_of_↩ right_lower_limb		Bone		Right_lower_limb
Skeleton_of_↩ right_pelvic_girdle		Bone		Right_pelvic_girdle
Skeleton_of_↩ right_upper_limb		Bone		Right_upper_limb
Skeleton_of_↩ upper_limb		Bone		
Skull		Bone		Head
Sphenoid_bone		Bone	Skull	
Stapes		Bone		Head
Sternum		Bone	Skeleton_of_rib_↩ cage	
Sutural_bone		Bone	Skull	
Talus		Bone		
Tarsal_bone		Bone	Set_of_tarsal_↩ bones	
Temporal_bone		Bone	Skull	
Tenth_rib		Bone	Skeleton_of_rib_↩ cage	
Tenth_thoracic_↩ vertebra	T10	Bone	Vertebral_column	Trunk

Third_cervical_↩ vertebra	C3	Bone	Vertebral_column	Neck
Third_lumbar_↩ vertebra	L3	Bone	Vertebral_column	Trunk
Third_↩ metacarpal_bone		Bone		
Third_metatarsal↩ _bone		Bone		
Third_rib		Bone	Skeleton_of_rib_↩ cage	
Third_thoracic_↩ vertebra	T3, T03	Bone	Vertebral_column	Trunk
Thoracic_vertebra		Bone	Vertebral_column	
Tibia		Bone	Skeleton_of_free↩ _lower_limb	
Trabecular_bone↩ _of_lateral_↩ cuneiform_bone		Bone		
Trabecular_bone↩ _of_left_lateral_↩ cuneiform_bone		Bone		Left_foot
Trabecular_bone↩ _of_right_lateral_↩ cuneiform_bone		Bone		Left_foot
Trapezium		Bone		
Trapezoid		Bone		
Triquetral		Bone		
True_rib		Bone	Skeleton_of_rib_↩ cage	
Twelfth_rib		Bone	Skeleton_of_rib_↩ cage	
Twelfth_thoracic_↩ vertebra	T12	Bone	Vertebral_column	Trunk
Typical_rib		Bone	Skeleton_of_rib_↩ cage	
Ulna		Bone		
Variant_cervical_↩ vertebra		Bone	Vertebral_column	
Variant_rib		Bone	Skeleton_of_rib_↩ cage	
Vertebra		Bone	Vertebral_column	
Vertebral_column	Backbone, Spinal_column, Spine	Bone		Body_proper
Vomer		Bone	Skull	
Zygomatic_bone		Bone	Skull	
Articular_capsule↩ _of_left_↩ acromioclavicular↩ _joint		Articular_capsule		

Articular_capsule↩ _of_left_elbow_↩ joint		Articular_capsule		
Articular_capsule↩ _of_left_↩ glenohumeral_joint		Articular_capsule		
Articular_capsule↩ _of_left_hip_joint		Articular_capsule		
Articular_capsule↩ _of_left_knee_joint		Articular_capsule		
Articular_capsule↩ _of_left_↩ sternoclavicular_↩ joint		Articular_capsule		
Articular_capsule↩ _of_right_↩ acromioclavicular↩ _joint		Articular_capsule		
Articular_capsule↩ _of_right_elbow_↩ joint		Articular_capsule		
Articular_capsule↩ _of_right_↩ glenohumeral_joint		Articular_capsule		
Articular_capsule↩ _of_right_hip_joint		Articular_capsule		
Articular_capsule↩ _of_right_knee_↩ joint		Articular_capsule		
Articular_capsule↩ _of_right_↩ sternoclavicular_↩ joint		Articular_capsule		
Left_anterior_↩ cruciate_ligament		Ligament		
Left_lateral_↩ collateral_ligament		Ligament		
Left_medial_↩ collateral_ligament		Ligament		
Left_posterior_↩ cruciate_ligament		Ligament		
Right_anterior_↩ cruciate_ligament		Ligament		
Right_lateral_↩ collateral_ligament		Ligament		
Right_medial_↩ collateral_ligament		Ligament		
Right_posterior_↩ cruciate_ligament		Ligament		
Left_lateral_↩ meniscus		Cartilage		
Left_medial_↩ meniscus		Cartilage		

Right_lateral_↩ meniscus		Cartilage		
Right_medial_↩ meniscus		Cartilage		

16 Appendix: List of landmarks

Landmark name	Synonyms	Description	PartOf	Bibliography
Acromial_edge_L	LMK_SAE_L	Point located beetween SAA and SAT on the edge of the acromion ; used_for_piper_↩ landmarks_left	Left_scapula	
Acromial_edge_R	LMK_SAE_R	Point located beetween SAA and SAT on the edge of the acromion ; used_for_piper_↩ landmarks_right	Right_scapula	
Acromial_tip_L	LMK_SAT_L	Lateral angle of the anterior tip of the acromion (so not the flat tip) ; used_for_piper_↩ landmarks_left	Left_scapula	
Acromial_tip_R	LMK_SAT_R	Lateral angle of the anterior tip of the acromion (so not the flat tip) ; used_for_piper_↩ landmarks_right	Right_scapula	
Acromioclavicular↩ _joint_L	LMK_SAJ_L	Closest point to CAJ on the Scapula ; used_for_piper_↩ landmarks_left	Left_scapula	
Acromioclavicular↩ _joint_R	LMK_SAJ_R	Closest point to CAJ on the Scapula ; used_for_piper_↩ landmarks_right	Right_scapula	
Angulus_↩ acromialis_of_↩ left_scapula	Angulus_↩ acromialis_of_↩ scapula_L, Scapula_SAA_L, AA_L	most laterodorsal point of the scapula ; used_for_piper↩ _landmarks_left	Left_scapula	

Angulus_↩ acromialis_of_↩ right_scapula	Angulus_↩ acromialis_of_↩ scapula_R, Scapula_SAA_R, AA_R	most laterodorsal point of the scapula ; used_for_piper↩ _landmarks_right	Right_scapula	
Angulus_inferior↩ _of_left_scapula	Angulus_inferior↩ _of_scapula_L, Scapula_SIA_L, AI_L	most caudal point of the scapula ; used_for_piper_↩ landmarks_left	Left_scapula	
Angulus_inferior↩ _of_right_scapula	Angulus_inferior↩ _of_scapula_R, Scapula_SIA_R, AI_R	most caudal point of the scapula ; used_for_piper_↩ landmarks_right	Right_scapula	
Anterior_aspect_↩ of_rib_2_R	LMK_RA2_R	Anterior aspect of the rib	Right_second_rib	
Anterior_aspect_↩ of_rib_3_R	LMK_RA3_R	Anterior aspect of the rib	Right_third_rib	
Anterior_aspect_↩ of_rib_4_R	LMK_RA4_R	Anterior aspect of the rib	Right_fourth_rib	
Anterior_aspect_↩ of_rib_5_R	LMK_RA5_R	Anterior aspect of the rib	Right_fifth_rib	
Anterior_↩ concavity_of_the↩ _Clavicle_L	LMK_CAA_L	Anterior concavity of the Clavicle ; used_for_piper_↩ landmarks_left	Left_clavicle	
Anterior_↩ concavity_of_the↩ _Clavicle_R	LMK_CAA_R	Anterior concavity of the Clavicle ; used_for_piper_↩ landmarks_right	Right_clavicle	
Anterior_↩ extremity_of_the↩ _left_eighth_rib	Ins_r08L, Extremity_head_↩ of_the_eighth_↩ left_rib, Anterior_Left_↩ Rib_8	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_left	Left_eighth_rib	
Anterior_↩ extremity_of_the↩ _left_eleventh_rib	Ins_r11L, Extremity_head_↩ of_the_eleventh_↩ left_rib, Anterior_Left_↩ Rib_11	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_left	Left_eleventh_rib	
Anterior_↩ extremity_of_the↩ _left_fifth_rib	Ins_r05L, Extremity_head_↩ of_the_fifth_left_↩ rib, Anterior_Left_↩ Rib_5	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_left	Left_fifth_rib	

Anterior_↩ extremity_of_the↩ _left_first_rib	Ins_r01L, Extremity_head_↩ of_the_first_left_↩ rib, Anterior_Left_↩ Rib_1	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_left	Left_first_rib	
Anterior_↩ extremity_of_the↩ _left_fourth_rib	Ins_r04L, Extremity_head_↩ of_the_fourth_↩ left_rib, Anterior_Left_↩ Rib_4	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_left	Left_fourth_rib	
Anterior_↩ extremity_of_the↩ _left_ninth_rib	Ins_r09L, Extremity_head_↩ of_the_ninth_left↩ _rib, Anterior_Left_↩ Rib_9	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_left	Left_ninth_rib	
Anterior_↩ extremity_of_the↩ _left_second_rib	Ins_r02L, Extremity_head_↩ of_the_second_↩ left_rib, Anterior_Left_↩ Rib_2	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_left	Left_second_rib	
Anterior_↩ extremity_of_the↩ _left_seventh_rib	Ins_r07L, Extremity_head_↩ of_the_seventh_↩ left_rib, Anterior_Left_↩ Rib_7	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_left	Left_seventh_rib	
Anterior_↩ extremity_of_the↩ _left_sixth_rib	Ins_r06L, Extremity_head_↩ of_the_sixth_left↩ _rib, Anterior_Left_↩ Rib_6	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_left	Left_sixth_rib	
Anterior_↩ extremity_of_the↩ _left_tenth_rib	Ins_r10L, Extremity_head_↩ of_the_tenth_left↩ _rib, Anterior_Left_↩ Rib_10	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_left	Left_tenth_rib	
Anterior_↩ extremity_of_the↩ _left_third_rib	Ins_r03L, Extremity_head_↩ of_the_third_left_↩ rib, Anterior_Left_↩ Rib_3	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_left	Left_third_rib	

Anterior_↩ extremity_of_the↩ _left_twelfth_rib	Ins_r12L, Extremity_head_↩ of_the_twelfth_↩ left_rib, Anterior_Left_↩ Rib_12	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_left	Left_twelfth_rib	
Anterior_↩ extremity_of_the↩ _right_eighth_rib	Ins_r08R, Extremity_head_↩ of_the_eighth_↩ right_rib, Anterior_Right_↩ Rib_8	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_right	Right_eighth_rib	
Anterior_↩ extremity_of_the↩ _right_eleventh_rib	Ins_r11R, Extremity_head_↩ of_the_eleventh_↩ right_rib, Anterior_Right_↩ Rib_11	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_right	Right_eleventh_rib	
Anterior_↩ extremity_of_the↩ _right_fifth_rib	Ins_r05R, Extremity_head_↩ of_the_fifth_right↩ _rib, Anterior_Right_↩ Rib_5	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_right	Right_fifth_rib	
Anterior_↩ extremity_of_the↩ _right_first_rib	Ins_r01R, Extremity_head_↩ of_the_first_right↩ _rib, Anterior_Right_↩ Rib_1	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_right	Right_first_rib	
Anterior_↩ extremity_of_the↩ _right_fourth_rib	Ins_r04R, Extremity_head_↩ of_the_fourth_↩ right_rib, Anterior_Right_↩ Rib_4	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_right	Right_fourth_rib	
Anterior_↩ extremity_of_the↩ _right_ninth_rib	Ins_r09R, Extremity_head_↩ of_the_ninth_↩ right_rib, Anterior_Right_↩ Rib_9	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_right	Right_ninth_rib	
Anterior_↩ extremity_of_the↩ _right_second_rib	Ins_r02R, Extremity_head_↩ of_the_second_↩ right_rib, Anterior_Right_↩ Rib_2	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_right	Right_second_rib	

Anterior_↩ extremity_of_the↩ _right_seventh_rib	Ins_r07R, Extremity_head_↩ of_the_seventh_↩ right_rib, Anterior_Right_↩ Rib_7	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_right	Right_seventh_rib	
Anterior_↩ extremity_of_the↩ _right_sixth_rib	Ins_r06R, Extremity_head_↩ of_the_sixth_↩ right_rib, Anterior_Right_↩ Rib_6	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_right	Right_sixth_rib	
Anterior_↩ extremity_of_the↩ _right_tenth_rib	Ins_r10R, Extremity_head_↩ of_the_tenth_↩ right_rib, Anterior_Right_↩ Rib_10	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_right	Right_tenth_rib	
Anterior_↩ extremity_of_the↩ _right_third_rib	Ins_r03R, Extremity_head_↩ of_the_third_↩ right_rib, Anterior_Right_↩ Rib_3	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_right	Right_third_rib	
Anterior_↩ extremity_of_the↩ _right_twelfth_rib	Ins_r12R, Extremity_head_↩ of_the_twelfth_↩ right_rib, Anterior_Right_↩ Rib_12	point at the anterior extremity of the rib ; used_for_piper_↩ landmarks_right	Right_twelfth_rib	
Anterior_↩ midpoint_of_the_↩ body_of_C3		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Third_cervical_↩ vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_C4		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Fourth_cervical_↩ vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_C5		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Fifth_cervical_↩ vertebra	

Anterior_↩ midpoint_of_the_↩ body_of_C6		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Sixth_cervical_↩ vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_C7		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Seventh_cervical↩ _vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_L1		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	First_lumbar_↩ vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_L2		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Second_lumbar_↩ vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_L3		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Third_lumbar_↩ vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_L4		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Fourth_lumbar_↩ vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_L5		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Fifth_lumbar_↩ vertebra	

Anterior_↩ midpoint_of_the_↩ body_of_T1		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	First_thoracic_↩ vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_T10		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Tenth_thoracic_↩ vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_T11		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Eleventh_↩ thoracic_vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_T12		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Twelfth_thoracic_↩ vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_T2		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Second_thoracic↩ _vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_T3		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Third_thoracic_↩ vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_T4		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Fourth_thoracic_↩ vertebra	

Anterior_↩ midpoint_of_the_↩ body_of_T5		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Fifth_thoracic_↩ vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_T6		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Sixth_thoracic_↩ vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_T7		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Seventh_thoracic↩ _vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_T8		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Eighth_thoracic_↩ vertebra	
Anterior_↩ midpoint_of_the_↩ body_of_T9		the anterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Ninth_thoracic_↩ vertebra	
Anterior_nasal_↩ spine	LMK_SNS	Lower spine apex of the nasal cavity ; used_for_piper_↩ landmarks	Skull	
Anterior_point_of↩ _the_inferior_↩ plate_of_C2	LMK_C2Lower↩ FrontEnd	Anterior point of the inferior end-plate ; used_for_piper_↩ landmarks	Axis	
Anterior_point_of↩ _the_inferior_↩ plate_of_C3	LMK_C3Lower↩ FrontEnd	Anterior point of the inferior end-plate	Third_cervical_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_C4	LMK_C4Lower↩ FrontEnd	Anterior point of the inferior end-plate	Fourth_cervical_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_C5	LMK_C5Lower↩ FrontEnd	Anterior point of the inferior end-plate	Fifth_cervical_↩ vertebra	

Anterior_point_of↩ _the_inferior_↩ plate_of_C6	LMK_C6Lower↩ FrontEnd	Anterior point of the inferior end-plate	Sixth_cervical_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_C7	LMK_C7Lower↩ FrontEnd	Anterior point of the inferior end-plate	Seventh_cervical↩ _vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_L1	LMK_L1Lower↩ FrontEnd	Anterior point of the inferior end-plate	First_lumbar_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_L2	LMK_L2Lower↩ FrontEnd	Anterior point of the inferior end-plate	Second_lumbar_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_L3	LMK_L3Lower↩ FrontEnd	Anterior point of the inferior end-plate	Third_lumbar_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_L4	LMK_L4Lower↩ FrontEnd	Anterior point of the inferior end-plate	Fourth_lumbar_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_L5	LMK_L5Lower↩ FrontEnd	Anterior point of the inferior end-plate	Fifth_lumbar_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_T1	LMK_T01Lower↩ FrontEnd	Anterior point of the inferior end-plate	First_thoracic_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_T10	LMK_T10Lower↩ FrontEnd	Anterior point of the inferior end-plate	Tenth_thoracic_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_T11	LMK_T11Lower↩ FrontEnd	Anterior point of the inferior end-plate	Eleventh_↩ thoracic_vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_T12	LMK_T12Lower↩ FrontEnd	Anterior point of the inferior end-plate	Twelfth_thoracic_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_T2	LMK_T02Lower↩ FrontEnd	Anterior point of the inferior end-plate	Second_thoracic↩ _vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_T3	LMK_T03Lower↩ FrontEnd	Anterior point of the inferior end-plate	Third_thoracic_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_T4	LMK_T04Lower↩ FrontEnd	Anterior point of the inferior end-plate	Fourth_thoracic_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_T5	LMK_T05Lower↩ FrontEnd	Anterior point of the inferior end-plate	Fifth_thoracic_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_T6	LMK_T06Lower↩ FrontEnd	Anterior point of the inferior end-plate	Sixth_thoracic_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_T7	LMK_T07Lower↩ FrontEnd	Anterior point of the inferior end-plate	Seventh_thoracic↩ _vertebra	

Anterior_point_of↩ _the_inferior_↩ plate_of_T8	LMK_T08Lower↩ FrontEnd	Anterior point of the inferior end-plate	Eighth_thoracic_↩ vertebra	
Anterior_point_of↩ _the_inferior_↩ plate_of_T9	LMK_T09Lower↩ FrontEnd	Anterior point of the inferior end-plate	Ninth_thoracic_↩ vertebra	
Anterior_point_of↩ _the_sacral_plate	VS, Most_anterior_↩ point_sacral_plate	; used_for_piper↩ _landmarks	Pelvic_skeleton	
Anterior_point_of↩ _the_superior_↩ plate_of_C3	VC3, LMK_C3↩ UpperFrontEnd		Third_cervical_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_C4	VC4, LMK_C4↩ UpperFrontEnd		Fourth_cervical_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_C5	VC5, LMK_C5↩ UpperFrontEnd		Fifth_cervical_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_C6	VC6, LMK_C6↩ UpperFrontEnd		Sixth_cervical_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_C7	VC7, LMK_C7↩ UpperFrontEnd		Seventh_cervical↩ _vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_L1	VL1, LMK_L1↩ UpperFrontEnd		First_lumbar_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_L2	VL2, LMK_L2↩ UpperFrontEnd		Second_lumbar_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_L3	VL3, LMK_L3↩ UpperFrontEnd		Third_lumbar_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_L4	VL4, LMK_L4↩ UpperFrontEnd		Fourth_lumbar_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_L5	VL5, LMK_L5↩ UpperFrontEnd		Fifth_lumbar_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_T1	VT01, LMK_T01↩ UpperFrontEnd		First_thoracic_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_T10	VT10, LMK_T10↩ UpperFrontEnd		Tenth_thoracic_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_T11	VT11, LMK_T11↩ UpperFrontEnd		Eleventh_↩ thoracic_vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_T12	VT12, LMK_T12↩ UpperFrontEnd		Twelfth_thoracic_↩ vertebra	

Anterior_point_of↩ _the_superior_↩ plate_of_T2	VT02, LMK_T02↩ UpperFrontEnd		Second_thoracic↩ _vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_T3	VT03, LMK_T03↩ UpperFrontEnd		Third_thoracic_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_T4	VT04, LMK_T04↩ UpperFrontEnd		Fourth_thoracic_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_T5	VT05, LMK_T05↩ UpperFrontEnd		Fifth_thoracic_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_T6	VT06, LMK_T06↩ UpperFrontEnd		Sixth_thoracic_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_T7	VT07, LMK_T07↩ UpperFrontEnd		Seventh_thoracic↩ _vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_T8	VT08, LMK_T08↩ UpperFrontEnd		Eighth_thoracic_↩ vertebra	
Anterior_point_of↩ _the_superior_↩ plate_of_T9	VT09, LMK_T09↩ UpperFrontEnd		Ninth_thoracic_↩ vertebra	
Anterior_↩ tuberosity_of_C2	LMK_C2AntArch	Point on the anterior tuberosity of the vertebral body just underneath the Odontoid ; used_for_piper_↩ landmarks	Axis	
Apex_of_the_↩ styloid_process_↩ of_the_left_fibula	FAX_L, Fibula_FAX_L		Left_fibula	
Apex_of_the_↩ styloid_process_↩ of_the_right_fibula	FAX_R, Fibula_FAX_R		Right_fibula	
Center_Sacral_↩ Plate		Center of the sacral plate ; used_for_↩ piper_landmarks	Sacrum	
Center_atlanto_↩ occipital_joint		the middle of left and right atlanto occipital joints ; used_for_piper_↩ landmarks	Skull	
Center_↩ depression_of_↩ left_radial_head	Center_↩ depression_of_↩ radial_head_L, Radius_RHE_L	bottom of the depresion of the proximal radial head ; used_for_piper_↩ landmarks_left	Left_radius	

Center_↩ depression_of_↩ right_radial_head	Center_↩ depression_of_↩ radial_head_R, Radius_RHE_R	bottom of the depresion of the proximal radial head ; used_for_piper_↩ landmarks_right	Right_radius	
Center_lower_↩ plate_of_C1		Center of the lower plate of the first cervical vertebra ; used_for_piper_↩ landmarks	Atlas	
Center_lower_↩ plate_of_C2		Center of the lower plate of the second cervical vertebra ; used_for_piper_↩ landmarks	Axis	
Center_lower_↩ plate_of_C3	LMK_C3LowerMid	Center of the lower plate of the third cervical vertebra ; used_for_piper_↩ landmarks	Third_cervical_↩ vertebra	
Center_lower_↩ plate_of_C4	LMK_C4LowerMid	Center of the lower plate of the fourth cervical vertebra ; used_for_piper_↩ landmarks	Fourth_cervical_↩ vertebra	
Center_lower_↩ plate_of_C5	LMK_C5LowerMid	Center of the lower plate of the fifth cervical vertebra ; used_for_piper_↩ landmarks	Fifth_cervical_↩ vertebra	
Center_lower_↩ plate_of_C6	LMK_C6LowerMid	Center of the lower plate of the sixth cervical vertebra ; used_for_piper_↩ landmarks	Sixth_cervical_↩ vertebra	
Center_lower_↩ plate_of_C7	LMK_C7LowerMid	Center of the lower plate of the seventh cervical vertebra ; used_for_piper_↩ landmarks	Seventh_cervical↩ _vertebra	
Center_lower_↩ plate_of_L1	LMK_L1LowerMid	Center of the lower plate of the first lumbar vertebra ; used_for_piper_↩ landmarks	First_lumbar_↩ vertebra	
Center_lower_↩ plate_of_L2	LMK_L2LowerMid	Center of the lower plate of the second lumbar vertebra ; used_for_piper_↩ landmarks	Second_lumbar_↩ vertebra	

Center_lower_↩ plate_of_L3	LMK_L3LowerMid	Center of the lower plate of the third lumbar vertebra ; used_for_piper_↩ landmarks	Third_lumbar_↩ vertebra	
Center_lower_↩ plate_of_L4	LMK_L4LowerMid	Center of the lower plate of the fourth lumbar vertebra ; used_for_piper_↩ landmarks	Fourth_lumbar_↩ vertebra	
Center_lower_↩ plate_of_L5	LMK_L5LowerMid	Center of the lower plate of the fifth lumbar vertebra ; used_for_piper_↩ landmarks	Fifth_lumbar_↩ vertebra	
Center_lower_↩ plate_of_T1	LMK_T01LowerMid	Center of the lower plate of the first thoracic vertebra ; used_for_piper_↩ landmarks	First_thoracic_↩ vertebra	
Center_lower_↩ plate_of_T10	LMK_T10LowerMid	Center of the lower plate of the tenth thoracic vertebra ; used_for_piper_↩ landmarks	Tenth_thoracic_↩ vertebra	
Center_lower_↩ plate_of_T11	LMK_T11LowerMid	Center of the lower plate of the eleventh thoracic vertebra ; used_for_piper_↩ landmarks	Eleventh_↩ thoracic_vertebra	
Center_lower_↩ plate_of_T12	LMK_T12LowerMid	Center of the lower plate of the twelfth thoracic vertebra ; used_for_piper_↩ landmarks	Twelfth_thoracic_↩ vertebra	
Center_lower_↩ plate_of_T2	LMK_T02LowerMid	Center of the lower plate of the second thoracic vertebra ; used_for_piper_↩ landmarks	Second_thoracic↩ _vertebra	
Center_lower_↩ plate_of_T3	LMK_T03LowerMid	Center of the lower plate of the third thoracic vertebra ; used_for_piper_↩ landmarks	Third_thoracic_↩ vertebra	
Center_lower_↩ plate_of_T4	LMK_T04LowerMid	Center of the lower plate of the fourth thoracic vertebra ; used_for_piper_↩ landmarks	Fourth_thoracic_↩ vertebra	

Center_lower_↩ plate_of_T5	LMK_T05LowerMid	Center of the lower plate of the fifth thoracic vertebra ; used_for_piper_↩ landmarks	Fifth_thoracic_↩ vertebra	
Center_lower_↩ plate_of_T6	LMK_T06LowerMid	Center of the lower plate of the sixth thoracic vertebra ; used_for_piper_↩ landmarks	Sixth_thoracic_↩ vertebra	
Center_lower_↩ plate_of_T7	LMK_T07LowerMid	Center of the lower plate of the seventh thoracic vertebra ; used_for_piper_↩ landmarks	Seventh_thoracic↩ _vertebra	
Center_lower_↩ plate_of_T8	LMK_T08LowerMid	Center of the lower plate of the eighth thoracic vertebra ; used_for_piper_↩ landmarks	Eighth_thoracic_↩ vertebra	
Center_lower_↩ plate_of_T9	LMK_T09LowerMid	Center of the lower plate of the nineth thoracic vertebra ; used_for_piper_↩ landmarks	Ninth_thoracic_↩ vertebra	
Center_of_lateral↩ _edge_of_↩ posterior_surface↩ _of_left_calcaneus	FCL_L	Center of the lateral edge of the posterior surface of the calcaneus	Skeleton_of_left_↩ foot	
Center_of_lateral↩ _edge_of_↩ posterior_surface↩ _of_right_↩ calcaneus	FCL_R	Center of the lateral edge of the posterior surface of the calcaneus	Skeleton_of_↩ right_foot	
Center_of_↩ medial_edge_of_↩ posterior_surface↩ _of_left_calcaneus	FCM_L	Center of the medial edge of the posterior surface of the calcaneus	Skeleton_of_left_↩ foot	
Center_of_↩ medial_edge_of_↩ posterior_surface↩ _of_right_↩ calcaneus	FCM_R	Center of the medial edge of the posterior surface of the calcaneus	Skeleton_of_↩ right_foot	
Center_of_↩ posterior_↩ calcaneal_face_↩ of_left_foot	FCC_L	Middle of the medial and lateral edges of the calcaneus	Skeleton_of_left_↩ foot	
Center_of_↩ posterior_↩ calcaneal_face_↩ of_right_foot	FCC_R	Middle of the medial and lateral edges of the calcaneus	Skeleton_of_↩ right_foot	

Center_upper_↩ plate_of_C2		Center of the upper plate of the second cervical vertebra ; used_for_piper_↩ landmarks	Axis	
Center_upper_↩ plate_of_C3	LMK_C3UpperMid	Center of the upper plate of the third cervical vertebra ; used_for_piper_↩ landmarks	Third_cervical_↩ vertebra	
Center_upper_↩ plate_of_C4	LMK_C4UpperMid	Center of the upper plate of the fourth cervical vertebra ; used_for_piper_↩ landmarks	Fourth_cervical_↩ vertebra	
Center_upper_↩ plate_of_C5	LMK_C5UpperMid	Center of the upper plate of the fifth cervical vertebra ; used_for_piper_↩ landmarks	Fifth_cervical_↩ vertebra	
Center_upper_↩ plate_of_C6	LMK_C6UpperMid	Center of the upper plate of the sixth cervical vertebra ; used_for_piper_↩ landmarks	Sixth_cervical_↩ vertebra	
Center_upper_↩ plate_of_C7	LMK_C7UpperMid	Center of the upper plate of the seventh cervical vertebra ; used_for_piper_↩ landmarks	Seventh_cervical↩ _vertebra	
Center_upper_↩ plate_of_L1	LMK_L1UpperMid	Center of the upper plate of the first lumbar vertebra ; used_for_piper_↩ landmarks	First_lumbar_↩ vertebra	
Center_upper_↩ plate_of_L2	LMK_L2UpperMid	Center of the upper plate of the second lumbar vertebra ; used_for_piper_↩ landmarks	Second_lumbar_↩ vertebra	
Center_upper_↩ plate_of_L3	LMK_L3UpperMid	Center of the upper plate of the third lumbar vertebra ; used_for_piper_↩ landmarks	Third_lumbar_↩ vertebra	
Center_upper_↩ plate_of_L4	LMK_L4UpperMid	Center of the upper plate of the fourth lumbar vertebra ; used_for_piper_↩ landmarks	Fourth_lumbar_↩ vertebra	

Center_upper_↩ plate_of_L5	LMK_L5UpperMid	Center of the upper plate of the fifth lumbar vertebra ; used_for_piper_↩ landmarks	Fifth_lumbar_↩ vertebra	
Center_upper_↩ plate_of_T1	LMK_T01UpperMid	Center of the upper plate of the first thoracic vertebra ; used_for_piper_↩ landmarks	First_thoracic_↩ vertebra	
Center_upper_↩ plate_of_T10	LMK_T10UpperMid	Center of the upper plate of the tenth thoracic vertebra ; used_for_piper_↩ landmarks	Tenth_thoracic_↩ vertebra	
Center_upper_↩ plate_of_T11	LMK_T11UpperMid	Center of the upper plate of the eleventh thoracic vertebra ; used_for_piper_↩ landmarks	Eleventh_↩ thoracic_vertebra	
Center_upper_↩ plate_of_T12	LMK_T12UpperMid	Center of the upper plate of the twelfth thoracic vertebra ; used_for_piper_↩ landmarks	Twelfth_thoracic_↩ vertebra	
Center_upper_↩ plate_of_T2	LMK_T02UpperMid	Center of the upper plate of the second thoracic vertebra ; used_for_piper_↩ landmarks	Second_thoracic↩ _vertebra	
Center_upper_↩ plate_of_T3	LMK_T03UpperMid	Center of the upper plate of the third thoracic vertebra ; used_for_piper_↩ landmarks	Third_thoracic_↩ vertebra	
Center_upper_↩ plate_of_T4	LMK_T04UpperMid	Center of the upper plate of the fourth thoracic vertebra ; used_for_piper_↩ landmarks	Fourth_thoracic_↩ vertebra	
Center_upper_↩ plate_of_T5	LMK_T05UpperMid	Center of the upper plate of the fifth thoracic vertebra ; used_for_piper_↩ landmarks	Fifth_thoracic_↩ vertebra	
Center_upper_↩ plate_of_T6	LMK_T06UpperMid	Center of the upper plate of the sixth thoracic vertebra ; used_for_piper_↩ landmarks	Sixth_thoracic_↩ vertebra	

Center_upper_↩ plate_of_T7	LMK_T07UpperMid	Center of the upper plate of the seventh thoracic vertebra ; used_for_piper_↩ landmarks	Seventh_thoracic↩ _vertebra	
Center_upper_↩ plate_of_T8	LMK_T08UpperMid	Center of the upper plate of the eighth thoracic vertebra ; used_for_piper_↩ landmarks	Eighth_thoracic_↩ vertebra	
Center_upper_↩ plate_of_T9	LMK_T09UpperMid	Center of the upper plate of the nineth thoracic vertebra ; used_for_piper_↩ landmarks	Ninth_thoracic_↩ vertebra	
Clavicle_CAE_L	Anterior_↩ convexity_of_the↩ _left_clavicle	; used_for_piper↩ _landmarks_left	Left_clavicle	
Clavicle_CAE_R	Anterior_↩ convexity_of_the↩ _right_clavicle	; used_for_piper↩ _landmarks_right	Right_clavicle	
Clavicle_CSJ_L	Sternoclavicular_↩ joint_of_the_left_↩ clavicle	; used_for_piper↩ _landmarks_left	Left_clavicle	
Clavicle_CSJ_R	Sternoclavicular_↩ joint_of_the_right↩ _clavicle	; used_for_piper↩ _landmarks_right	Right_clavicle	
Corner_of_the_↩ eye_L	LMK_SLC_L	Undepressed skin surface point at the lateral junction of the upper and lower eyelids		
Corner_of_the_↩ eye_L_on_skull		corner of the left eye but on skull ; used_for_piper_↩ landmarks	Skull	
Corner_of_the_↩ eye_R	LMK_SLC_R	Undepressed skin surface point at the lateral junction of the upper and lower eyelids		
Corner_of_the_↩ eye_R_on_skull		corner of the right eye but on skull ; used_for_piper_↩ landmarks	Skull	
Crest_tubercle_L	LMK_ICT_L, Crest_tubercle_Left	Widening of the anterior part of the iliac crest ; used_for_piper_↩ landmarks	Left_hip_bone	

Crest_tubercle_R	LMK_ICT_R, Crest_tubercle_↩ Right	Widening of the anterior part of the iliac crest ; used_for_piper_↩ landmarks	Right_hip_bone	
DC1		; used_for_piper↩ _landmarks	Atlas	
DC2			Axis	
Distal_phalanx_↩ of_the_big_toe_↩ of_the_left_foot	Foot_BD6_L	Distal phalanx of the big toe (6=big toe 10=fifth toe): at the center of the proximal epiphysis just distal to the joint edge	Skeleton_of_left_↩ foot	
Distal_phalanx_↩ of_the_big_toe_↩ of_the_right_foot	BD6_R, Foot_BD6_R	Distal phalanx of the big toe (6=big toe 10=fifth toe): at the center of the proximal epiphysis just distal to the joint edge	Skeleton_of_↩ right_foot	
Distal_phalanx_↩ of_the_fifth_toe_↩ of_the_left_foot	Foot_BD10_L	Distal phalanx of the second toe (6=big toe 10=fifth toe): at the center of the proximal epiphysis just distal to the joint edge	Skeleton_of_left_↩ foot	
Distal_phalanx_↩ of_the_fifth_toe_↩ of_the_right_foot	BD10_R, Foot_BD10_R	Distal phalanx of the second toe (6=big toe 10=fifth toe): at the center of the proximal epiphysis just distal to the joint edge	Skeleton_of_↩ right_foot	
Distal_phalanx_↩ of_the_second_↩ toe_of_the_left_↩ foot	Foot_BD7_L	Distal phalanx of the second toe (6=big toe 10=fifth toe): at the center of the proximal epiphysis just distal to the joint edge	Skeleton_of_left_↩ foot	
Distal_phalanx_↩ of_the_second_↩ toe_of_the_right↩ _foot	BD7_R, Foot_BD7_R	Distal phalanx of the second toe (6=big toe 10=fifth toe): at the center of the proximal epiphysis just distal to the joint edge	Skeleton_of_↩ right_foot	

Dorsal_point_of_↩ left_↩ acromioclavicular↩ _joint	LMK_CAJ_L, Dorsal_point_of_↩ acromioclavicular↩ _joint_L, Clavicle_CAJ_L, AC_L	Most dorsal point on the acromioclavicular joint (shared with the scapula) ; used_for_piper_↩ landmarks_left	Left_clavicle	
Dorsal_point_of_↩ right_↩ acromioclavicular↩ _joint	LMK_CAJ_R, Dorsal_point_of_↩ acromioclavicular↩ _joint_R, Clavicle_CAJ_R, AC_R	Most dorsal point on the acromioclavicular joint (shared with the scapula) ; used_for_piper_↩ landmarks_right	Right_clavicle	
Dorsal_tubercule↩ _of_left_radius	Dorsal_tubercule↩ _of_radius_L, Radius_RDT_L	Ridge between radioscaphoid fossa and radioulna fossa ; used_for_piper_↩ landmarks_left	Left_radius	
Dorsal_tubercule↩ _of_right_radius	Dorsal_tubercule↩ _of_radius_R, Radius_RDT_R	Ridge between radioscaphoid fossa and radioulna fossa ; used_for_piper_↩ landmarks_right	Right_radius	
External_↩ occipital_↩ protuberance	LMK_SOP, Skull_SOP	Largest protuberance located at the posterorior aspect of the occipital bone ; used_for_↩ piper_landmarks	Skull	FMA
Femur_FAM_L			Left_femur	
Femur_FAM_R			Right_femur	
Femur_FBE_L			Left_femur	
Femur_FBE_R			Right_femur	
Femur_FCH1_L			Left_femur	
Femur_FCH1_R			Right_femur	
Femur_FCH2_L			Left_femur	
Femur_FCH2_R			Right_femur	
Femur_FCH3_L			Left_femur	
Femur_FCH3_R			Right_femur	
Femur_FCH4_L			Left_femur	
Femur_FCH4_R			Right_femur	
Femur_FCH5_L			Left_femur	
Femur_FCH5_R			Right_femur	
Femur_FCH6_L			Left_femur	
Femur_FCH6_R			Right_femur	
Femur_FLC_L			Left_femur	

Femur_FLC_R			Right_femur	
Femur_FLG_L			Left_femur	
Femur_FLG_R			Right_femur	
Femur_FMC_L			Left_femur	
Femur_FMC_R			Right_femur	
Femur_FMG_L			Left_femur	
Femur_FMG_R			Right_femur	
Femur_FMS_L			Left_femur	
Femur_FMS_R			Right_femur	
Femur_FPS_L			Left_femur	
Femur_FPS_R			Right_femur	
Femur_FTA_L			Left_femur	
Femur_FTA_R			Right_femur	
Femur_FTP_L			Left_femur	
Femur_FTP_R			Right_femur	
Femur_FUE_L			Left_femur	
Femur_FUE_R			Right_femur	
Fibula_FAL_L			Left_fibula	
Fibula_FAL_R			Right_fibula	
Fibula_FNE_L			Left_fibula	
Fibula_FNE_R			Right_fibula	
Foot_BC10_L			Skeleton_of_left_↩ foot	
Foot_BC10_R			Skeleton_of_↩ right_foot	
Foot_BC7_L			Skeleton_of_left_↩ foot	
Foot_BC7_R			Skeleton_of_↩ right_foot	
Foot_BC8_L			Skeleton_of_left_↩ foot	
Foot_BC8_R			Skeleton_of_↩ right_foot	
Foot_BC9_L			Skeleton_of_left_↩ foot	
Foot_BC9_R			Skeleton_of_↩ right_foot	
Foot_BD8_L			Skeleton_of_left_↩ foot	
Foot_BD8_R			Skeleton_of_↩ right_foot	
Foot_BD9_L			Skeleton_of_left_↩ foot	
Foot_BD9_R			Skeleton_of_↩ right_foot	
Foot_BP10_L			Skeleton_of_left_↩ foot	
Foot_BP10_R			Skeleton_of_↩ right_foot	

Foot_BP6_L			Skeleton_of_left_↩ foot	
Foot_BP6_R			Skeleton_of_↩ right_foot	
Foot_BP7_L			Skeleton_of_left_↩ foot	
Foot_BP7_R			Skeleton_of_↩ right_foot	
Foot_BP8_L			Skeleton_of_left_↩ foot	
Foot_BP8_R			Skeleton_of_↩ right_foot	
Foot_BP9_L			Skeleton_of_left_↩ foot	
Foot_BP9_R			Skeleton_of_↩ right_foot	
Foot_CL10_L			Skeleton_of_left_↩ foot	
Foot_CL10_R			Skeleton_of_↩ right_foot	
Foot_CL7_L			Skeleton_of_left_↩ foot	
Foot_CL7_R			Skeleton_of_↩ right_foot	
Foot_CL8_L			Skeleton_of_left_↩ foot	
Foot_CL8_R			Skeleton_of_↩ right_foot	
Foot_CL9_L			Skeleton_of_left_↩ foot	
Foot_CL9_R			Skeleton_of_↩ right_foot	
Foot_CM10_L			Skeleton_of_left_↩ foot	
Foot_CM10_R			Skeleton_of_↩ right_foot	
Foot_CM7_L			Skeleton_of_left_↩ foot	
Foot_CM7_R			Skeleton_of_↩ right_foot	
Foot_CM8_L			Skeleton_of_left_↩ foot	
Foot_CM8_R			Skeleton_of_↩ right_foot	
Foot_CM9_L			Skeleton_of_left_↩ foot	
Foot_CM9_R			Skeleton_of_↩ right_foot	
Foot_DL10_L			Skeleton_of_left_↩ foot	
Foot_DL10_R			Skeleton_of_↩ right_foot	

Foot_DL6_L			Skeleton_of_left_↩ foot	
Foot_DL6_R			Skeleton_of_↩ right_foot	
Foot_DL7_L			Skeleton_of_left_↩ foot	
Foot_DL7_R			Skeleton_of_↩ right_foot	
Foot_DL8_L			Skeleton_of_left_↩ foot	
Foot_DL8_R			Skeleton_of_↩ right_foot	
Foot_DL9_L			Skeleton_of_left_↩ foot	
Foot_DL9_R			Skeleton_of_↩ right_foot	
Foot_DM10_L			Skeleton_of_left_↩ foot	
Foot_DM10_R			Skeleton_of_↩ right_foot	
Foot_DM6_L			Skeleton_of_left_↩ foot	
Foot_DM6_R			Skeleton_of_↩ right_foot	
Foot_DM7_L			Skeleton_of_left_↩ foot	
Foot_DM7_R			Skeleton_of_↩ right_foot	
Foot_DM8_L			Skeleton_of_left_↩ foot	
Foot_DM8_R			Skeleton_of_↩ right_foot	
Foot_DM9_L			Skeleton_of_left_↩ foot	
Foot_DM9_R			Skeleton_of_↩ right_foot	
Foot_FGA_L			Skeleton_of_left_↩ foot	
Foot_FGA_R			Skeleton_of_↩ right_foot	
Foot_FM1_L			Skeleton_of_left_↩ foot	
Foot_FM1_R			Skeleton_of_↩ right_foot	
Foot_FM2_L			Skeleton_of_left_↩ foot	
Foot_FM2_R			Skeleton_of_↩ right_foot	
Foot_FM3_L			Skeleton_of_left_↩ foot	
Foot_FM3_R			Skeleton_of_↩ right_foot	

Foot_FM4_L			Skeleton_of_left_↩ foot	
Foot_FM4_R			Skeleton_of_↩ right_foot	
Foot_FM5_L			Skeleton_of_left_↩ foot	
Foot_FM5_R			Skeleton_of_↩ right_foot	
Foot_FMT_R			Skeleton_of_↩ right_foot	
Foot_FNT_L			Skeleton_of_left_↩ foot	
Foot_FNT_R			Skeleton_of_↩ right_foot	
Foot_FPT_L			Skeleton_of_left_↩ foot	
Foot_FPT_R			Skeleton_of_↩ right_foot	
Foot_FST_L			Skeleton_of_left_↩ foot	
Foot_FST_R			Skeleton_of_↩ right_foot	
Foot_MT_L			Skeleton_of_left_↩ foot	
Foot_MT_R			Skeleton_of_↩ right_foot	
Foot_PL10_L			Skeleton_of_left_↩ foot	
Foot_PL10_R			Skeleton_of_↩ right_foot	
Foot_PL6_L			Skeleton_of_left_↩ foot	
Foot_PL6_R			Skeleton_of_↩ right_foot	
Foot_PL7_L			Skeleton_of_left_↩ foot	
Foot_PL7_R			Skeleton_of_↩ right_foot	
Foot_PL8_L			Skeleton_of_left_↩ foot	
Foot_PL8_R			Skeleton_of_↩ right_foot	
Foot_PL9_L			Skeleton_of_left_↩ foot	
Foot_PL9_R			Skeleton_of_↩ right_foot	
Foot_PM10_L			Skeleton_of_left_↩ foot	
Foot_PM10_R			Skeleton_of_↩ right_foot	
Foot_PM6_L			Skeleton_of_left_↩ foot	

Foot_PM6_R			Skeleton_of_↩ right_foot	
Foot_PM7_L			Skeleton_of_left_↩ foot	
Foot_PM7_R			Skeleton_of_↩ right_foot	
Foot_PM8_L			Skeleton_of_left_↩ foot	
Foot_PM8_R			Skeleton_of_↩ right_foot	
Foot_PM9_L			Skeleton_of_left_↩ foot	
Foot_PM9_R			Skeleton_of_↩ right_foot	
Glabella	LMK_SGL	Most forward projection of the frontal bone in the midline at the level of the brow ridges ; used_for_piper_↩ landmarks	Skull	
Greater_tubercle↩ _of_the_left_↩ humerus	HGT_L, Humerus_HGT_L	; used_for_piper↩ _landmarks_left	Left_humerus	
Greater_tubercle↩ _of_the_right_↩ humerus	HGT_R, Humerus_HGT_R	; used_for_piper↩ _landmarks_right	Right_humerus	
Hand_BC2_L			Skeleton_of_left_↩ hand	
Hand_BC2_R			Skeleton_of_↩ right_hand	
Hand_BC3_L			Skeleton_of_left_↩ hand	
Hand_BC3_R			Skeleton_of_↩ right_hand	
Hand_BC4_L			Skeleton_of_left_↩ hand	
Hand_BC4_R			Skeleton_of_↩ right_hand	
Hand_BC5_L			Skeleton_of_left_↩ hand	
Hand_BC5_R			Skeleton_of_↩ right_hand	
Hand_BD1_L			Skeleton_of_left_↩ hand	
Hand_BD1_R			Skeleton_of_↩ right_hand	
Hand_BD2_L			Skeleton_of_left_↩ hand	
Hand_BD2_R			Skeleton_of_↩ right_hand	
Hand_BD3_L			Skeleton_of_left_↩ hand	

Hand_BD3_R			Skeleton_of_↩ right_hand	
Hand_BD4_L			Skeleton_of_left_↩ hand	
Hand_BD4_R			Skeleton_of_↩ right_hand	
Hand_BD5_L			Skeleton_of_left_↩ hand	
Hand_BD5_R			Skeleton_of_↩ right_hand	
Hand_BP1_L			Skeleton_of_left_↩ hand	
Hand_BP1_R			Skeleton_of_↩ right_hand	
Hand_BP2_L			Skeleton_of_left_↩ hand	
Hand_BP2_R			Skeleton_of_↩ right_hand	
Hand_BP3_L			Skeleton_of_left_↩ hand	
Hand_BP3_R			Skeleton_of_↩ right_hand	
Hand_BP4_L			Skeleton_of_left_↩ hand	
Hand_BP4_R			Skeleton_of_↩ right_hand	
Hand_BP5_L			Skeleton_of_left_↩ hand	
Hand_BP5_R			Skeleton_of_↩ right_hand	
Hand_CL2_L			Skeleton_of_left_↩ hand	
Hand_CL2_R			Skeleton_of_↩ right_hand	
Hand_CL3_L			Skeleton_of_left_↩ hand	
Hand_CL3_R			Skeleton_of_↩ right_hand	
Hand_CL4_L			Skeleton_of_left_↩ hand	
Hand_CL4_R			Skeleton_of_↩ right_hand	
Hand_CL5_L			Skeleton_of_left_↩ hand	
Hand_CL5_R			Skeleton_of_↩ right_hand	
Hand_CM2_L			Skeleton_of_left_↩ hand	
Hand_CM2_R			Skeleton_of_↩ right_hand	
Hand_CM3_L			Skeleton_of_left_↩ hand	

Hand_CM3_R			Skeleton_of_↩ right_hand	
Hand_CM4_L			Skeleton_of_left_↩ hand	
Hand_CM4_R			Skeleton_of_↩ right_hand	
Hand_CM5_L			Skeleton_of_left_↩ hand	
Hand_CM5_R			Skeleton_of_↩ right_hand	
Hand_DL1_L			Skeleton_of_left_↩ hand	
Hand_DL1_R			Skeleton_of_↩ right_hand	
Hand_DL2_L			Skeleton_of_left_↩ hand	
Hand_DL2_R			Skeleton_of_↩ right_hand	
Hand_DL3_L			Skeleton_of_left_↩ hand	
Hand_DL3_R			Skeleton_of_↩ right_hand	
Hand_DL4_L			Skeleton_of_left_↩ hand	
Hand_DL4_R			Skeleton_of_↩ right_hand	
Hand_DL5_L			Skeleton_of_left_↩ hand	
Hand_DL5_R			Skeleton_of_↩ right_hand	
Hand_DM1_L			Skeleton_of_left_↩ hand	
Hand_DM1_R			Skeleton_of_↩ right_hand	
Hand_DM2_L			Skeleton_of_left_↩ hand	
Hand_DM2_R			Skeleton_of_↩ right_hand	
Hand_DM3_L			Skeleton_of_left_↩ hand	
Hand_DM3_R			Skeleton_of_↩ right_hand	
Hand_DM4_L			Skeleton_of_left_↩ hand	
Hand_DM4_R			Skeleton_of_↩ right_hand	
Hand_DM5_L			Skeleton_of_left_↩ hand	
Hand_DM5_R			Skeleton_of_↩ right_hand	
Hand_HHH_L			Skeleton_of_left_↩ hand	

Hand_HHH_R			Skeleton_of_↩ right_hand	
Hand_HL2_L			Skeleton_of_left_↩ hand	
Hand_HL2_R			Skeleton_of_↩ right_hand	
Hand_HL3_L			Skeleton_of_left_↩ hand	
Hand_HL3_R			Skeleton_of_↩ right_hand	
Hand_HL4_L			Skeleton_of_left_↩ hand	
Hand_HL4_R			Skeleton_of_↩ right_hand	
Hand_HM1_L			Skeleton_of_left_↩ hand	
Hand_HM1_R			Skeleton_of_↩ right_hand	
Hand_HM2_L			Skeleton_of_left_↩ hand	
Hand_HM2_R			Skeleton_of_↩ right_hand	
Hand_HM3_L			Skeleton_of_left_↩ hand	
Hand_HM3_R			Skeleton_of_↩ right_hand	
Hand_HM4_L			Skeleton_of_left_↩ hand	
Hand_HM4_R			Skeleton_of_↩ right_hand	
Hand_HM5_L			Skeleton_of_left_↩ hand	
Hand_HM5_R			Skeleton_of_↩ right_hand	
Hand_HNT_L			Skeleton_of_left_↩ hand	
Hand_HNT_R			Skeleton_of_↩ right_hand	
Hand_HPI_L			Skeleton_of_left_↩ hand	
Hand_HPI_R			Skeleton_of_↩ right_hand	
Hand_MB1_L			Skeleton_of_left_↩ hand	
Hand_MB1_R			Skeleton_of_↩ right_hand	
Hand_MB2_L			Skeleton_of_left_↩ hand	
Hand_MB2_R			Skeleton_of_↩ right_hand	
Hand_MB3_L			Skeleton_of_left_↩ hand	

Hand_MB3_R			Skeleton_of_↩ right_hand	
Hand_MB4_L			Skeleton_of_left_↩ hand	
Hand_MB4_R			Skeleton_of_↩ right_hand	
Hand_MB5_L			Skeleton_of_left_↩ hand	
Hand_MB5_R			Skeleton_of_↩ right_hand	
Hand_MSL_L			Skeleton_of_left_↩ hand	
Hand_MSL_R			Skeleton_of_↩ right_hand	
Hand_MSM_L			Skeleton_of_left_↩ hand	
Hand_MSM_R			Skeleton_of_↩ right_hand	
Hand_PL1_L			Skeleton_of_left_↩ hand	
Hand_PL1_R			Skeleton_of_↩ right_hand	
Hand_PL2_L			Skeleton_of_left_↩ hand	
Hand_PL2_R			Skeleton_of_↩ right_hand	
Hand_PL3_L			Skeleton_of_left_↩ hand	
Hand_PL3_R			Skeleton_of_↩ right_hand	
Hand_PL4_L			Skeleton_of_left_↩ hand	
Hand_PL4_R			Skeleton_of_↩ right_hand	
Hand_PL5_L			Skeleton_of_left_↩ hand	
Hand_PL5_R			Skeleton_of_↩ right_hand	
Hand_PM1_L			Skeleton_of_left_↩ hand	
Hand_PM1_R			Skeleton_of_↩ right_hand	
Hand_PM2_L			Skeleton_of_left_↩ hand	
Hand_PM2_R			Skeleton_of_↩ right_hand	
Hand_PM3_L			Skeleton_of_left_↩ hand	
Hand_PM3_R			Skeleton_of_↩ right_hand	
Hand_PM4_L			Skeleton_of_left_↩ hand	

Hand_PM4_R			Skeleton_of_↩ right_hand	
Hand_PM5_L			Skeleton_of_left_↩ hand	
Hand_PM5_R			Skeleton_of_↩ right_hand	
Head_center_of_↩ left_femur	head_center_↩ femur_L, Femur_FCH_L	; used_for_piper↩ _landmarks_left	Left_femur	
Head_center_of_↩ left_humerus	Head_center_of_↩ humerus_L, GH_L	glenohumeral center of rotation ; used_for_piper_↩ landmarks_left	Left_humerus	
Head_center_of_↩ right_femur	head_center_↩ femur_R, Femur_FCH_R	; used_for_piper↩ _landmarks_right	Right_femur	
Head_center_of_↩ right_humerus	Head_center_of_↩ humerus_R, GH_R	glenohumeral center of rotation ; used_for_piper_↩ landmarks_right	Right_humerus	
Head_of_the_left↩ _talus	FHE_L	The head of the talus is located on the most anterior aspect of the bone. It has a relatively sharp superior edge.	Skeleton_of_left_↩ foot	
Head_of_the_↩ right_talus	FHE_R	The head of the talus is located on the most anterior aspect of the bone. It has a relatively sharp superior edge.	Skeleton_of_↩ right_foot	
Humerus_HDT_L			Left_humerus	
Humerus_HDT_R			Right_humerus	
Humerus_HLT_L			Left_humerus	
Humerus_HLT_R			Right_humerus	
Humerus_HML_L			Left_humerus	
Humerus_HML_R			Right_humerus	
Humerus_HMU_L			Left_humerus	
Humerus_HMU_R			Right_humerus	
Ilium_IPP_L	Ilium_pubic_↩ spine_left	; used_for_piper↩ _landmarks	Pelvic_skeleton	
Ilium_IPP_R	Ilium_pubic_↩ spine_right	; used_for_piper↩ _landmarks	Pelvic_skeleton	
Incisive	LMK_JIN	Protuberance between the two lower incisors ; used_for_piper_↩ landmarks	Mandible	

Inferior_angle_of↩ _left_scapula	SIA_L		Left_scapula	
Inferior_angle_of↩ _right_scapula	SIA_R		Right_scapula	
Inferior_crest_of↩ _the_jaw	LMK_JIC	Inferior edge of the jaw bone ; used_for_piper_↩ landmarks	Mandible	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_L1	LMK_LV1	; used_for_piper↩ _landmarks	First_lumbar_↩ vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_L2	LMK_LV2	; used_for_piper↩ _landmarks	Second_lumbar_↩ vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_L3	LMK_LV3	; used_for_piper↩ _landmarks	Third_lumbar_↩ vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_L4	LMK_LV4	; used_for_piper↩ _landmarks	Fourth_lumbar_↩ vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_L5	LMK_LV5	; used_for_piper↩ _landmarks	Fifth_lumbar_↩ vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_T1	LMK_TV01	; used_for_piper↩ _landmarks	First_thoracic_↩ vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_T10	LMK_TV10	; used_for_piper↩ _landmarks	Tenth_thoracic_↩ vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_T11	LMK_TV11	; used_for_piper↩ _landmarks	Eleventh_↩ thoracic_vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_T12	LMK_TV12	; used_for_piper↩ _landmarks	Twelfth_thoracic_↩ vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_T2	LMK_TV02	; used_for_piper↩ _landmarks	Second_thoracic↩ _vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_T3	LMK_TV03	; used_for_piper↩ _landmarks	Third_thoracic_↩ vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_T4	LMK_TV04	; used_for_piper↩ _landmarks	Fourth_thoracic_↩ vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_T5	LMK_TV05	; used_for_piper↩ _landmarks	Fifth_thoracic_↩ vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_T6	LMK_TV06	; used_for_piper↩ _landmarks	Sixth_thoracic_↩ vertebra	

Inferior_point_of↩ _tip_of_spinous_↩ process_of_T7	LMK_TV07	; used_for_piper↩ _landmarks	Seventh_thoracic↩ _vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_T8	LMK_TV08	; used_for_piper↩ _landmarks	Eighth_thoracic_↩ vertebra	
Inferior_point_of↩ _tip_of_spinous_↩ process_of_T9	LMK_TV09	; used_for_piper↩ _landmarks	Ninth_thoracic_↩ vertebra	
Infraorbital_↩ Foramen_L	LMK_SIF_L	Center of the small depressed Foramen found under the orbit	Skull	
Infraorbital_↩ Foramen_R	LMK_SIF_R	Center of the small depressed Foramen found under the orbit	Skull	
LMK_C0_SZR_L	Skull_SZR_L		Skull	
LMK_C0_SZR_R	Skull_SZR_R		Skull	
LMK_C3Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_C3	; used_for_piper↩ _landmarks	Third_cervical_↩ vertebra	
LMK_C3Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_C3	; used_for_piper↩ _landmarks	Third_cervical_↩ vertebra	
LMK_C4Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_C4	; used_for_piper↩ _landmarks	Fourth_cervical_↩ vertebra	
LMK_C4Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_C4	; used_for_piper↩ _landmarks	Fourth_cervical_↩ vertebra	
LMK_C5Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_C5	; used_for_piper↩ _landmarks	Fifth_cervical_↩ vertebra	
LMK_C5Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_C5	; used_for_piper↩ _landmarks	Fifth_cervical_↩ vertebra	
LMK_C6Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_C6	; used_for_piper↩ _landmarks	Sixth_cervical_↩ vertebra	
LMK_C6Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_C6	; used_for_piper↩ _landmarks	Sixth_cervical_↩ vertebra	
LMK_C7Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_C7	; used_for_piper↩ _landmarks	Seventh_cervical↩ _vertebra	
LMK_C7Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_C7	; used_for_piper↩ _landmarks	Seventh_cervical↩ _vertebra	
LMK_L1Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_L1	; used_for_piper↩ _landmarks	First_lumbar_↩ vertebra	

LMK_L1Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_L1	; used_for_piper↩ _landmarks	First_lumbar_↩ vertebra	
LMK_L2Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_L2	; used_for_piper↩ _landmarks	Second_lumbar_↩ vertebra	
LMK_L2Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_L2	; used_for_piper↩ _landmarks	Second_lumbar_↩ vertebra	
LMK_L3Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_L3	; used_for_piper↩ _landmarks	Third_lumbar_↩ vertebra	
LMK_L3Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_L3	; used_for_piper↩ _landmarks	Third_lumbar_↩ vertebra	
LMK_L4Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_L4	; used_for_piper↩ _landmarks	Fourth_lumbar_↩ vertebra	
LMK_L4Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_L4	; used_for_piper↩ _landmarks	Fourth_lumbar_↩ vertebra	
LMK_L5Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_L5	; used_for_piper↩ _landmarks	Fifth_lumbar_↩ vertebra	
LMK_L5Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_L5	; used_for_piper↩ _landmarks	Fifth_lumbar_↩ vertebra	
LMK_T01Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_T1	; used_for_piper↩ _landmarks	First_thoracic_↩ vertebra	
LMK_T01Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_T1	; used_for_piper↩ _landmarks	First_thoracic_↩ vertebra	
LMK_T02Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_T2	; used_for_piper↩ _landmarks	Second_thoracic↩ _vertebra	
LMK_T02Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_T2	; used_for_piper↩ _landmarks	Second_thoracic↩ _vertebra	
LMK_T03Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_T3	; used_for_piper↩ _landmarks	Third_thoracic_↩ vertebra	
LMK_T03Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_T3	; used_for_piper↩ _landmarks	Third_thoracic_↩ vertebra	
LMK_T04Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_T4	; used_for_piper↩ _landmarks	Fourth_thoracic_↩ vertebra	
LMK_T04Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_T4	; used_for_piper↩ _landmarks	Fourth_thoracic_↩ vertebra	

LMK_T05Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_T5	; used_for_piper↩ _landmarks	Fifth_thoracic_↩ vertebra	
LMK_T05Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_T5	; used_for_piper↩ _landmarks	Fifth_thoracic_↩ vertebra	
LMK_T06Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_T6	; used_for_piper↩ _landmarks	Sixth_thoracic_↩ vertebra	
LMK_T06Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_T6	; used_for_piper↩ _landmarks	Sixth_thoracic_↩ vertebra	
LMK_T07Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_T7	; used_for_piper↩ _landmarks	Seventh_thoracic↩ _vertebra	
LMK_T07Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_T7	; used_for_piper↩ _landmarks	Seventh_thoracic↩ _vertebra	
LMK_T08Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_T8	; used_for_piper↩ _landmarks	Eighth_thoracic_↩ vertebra	
LMK_T08Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_T8	; used_for_piper↩ _landmarks	Eighth_thoracic_↩ vertebra	
LMK_T09Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_T9	; used_for_piper↩ _landmarks	Ninth_thoracic_↩ vertebra	
LMK_T09Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_T9	; used_for_piper↩ _landmarks	Ninth_thoracic_↩ vertebra	
LMK_T10Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_T10	; used_for_piper↩ _landmarks	Tenth_thoracic_↩ vertebra	
LMK_T10Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_T10	; used_for_piper↩ _landmarks	Tenth_thoracic_↩ vertebra	
LMK_T11Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_T11	; used_for_piper↩ _landmarks	Eleventh_↩ thoracic_vertebra	
LMK_T11Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_T11	; used_for_piper↩ _landmarks	Eleventh_↩ thoracic_vertebra	
LMK_T12Trans↩ Proc_L	Tip_of_left_↩ transverse_↩ process_of_T12	; used_for_piper↩ _landmarks	Twelfth_thoracic_↩ vertebra	
LMK_T12Trans↩ Proc_R	Tip_of_right_↩ transverse_↩ process_of_T12	; used_for_piper↩ _landmarks	Twelfth_thoracic_↩ vertebra	
Lateral_aspect_↩ of_olecranon_of_↩ the_left_ulna	UOL_L, Ulna_UOL_L		Left_ulna	

Lateral_aspect_↩ of_olecranon_of_↩ the_right_ulna	UOL_R, Ulna_UOL_R		Right_ulna	
Lateral_aspect_↩ of_rib_4_L	LMK_RL4_L	Lateral aspect of the rib	Left_fourth_rib	
Lateral_aspect_↩ of_rib_4_R	LMK_RL4_R	Lateral aspect of the rib	Right_fourth_rib	
Lateral_aspect_↩ of_rib_5_L	LMK_RL5_L	Lateral aspect of the rib	Left_fifth_rib	
Lateral_aspect_↩ of_rib_5_R	LMK_RL5_R	Lateral aspect of the rib	Right_fifth_rib	
Lateral_aspect_↩ of_rib_6_L	LMK_RL6_L	Lateral aspect of the rib	Left_sixth_rib	
Lateral_aspect_↩ of_rib_6_R	LMK_RL6_R	Lateral aspect of the rib	Right_sixth_rib	
Lateral_aspect_↩ of_rib_7_L	LMK_RL7_L	Lateral aspect of the rib	Left_seventh_rib	
Lateral_aspect_↩ of_rib_7_R	LMK_RL7_R	Lateral aspect of the rib	Right_seventh_rib	
Lateral_↩ epicondyle_of_↩ left_femur	LLATEPIC, Femur_FLE_L, FEL_L	most lateral point on lateral epicondyle ; used_for_piper_↩ landmarks_left	Left_femur	Kepple1997
Lateral_↩ epicondyle_of_↩ left_humerus	Lateral_↩ epicondyle_of_↩ humerus_L, Humerus_HLE_L, EL_L	most caudal point on lateral epicondyle ; used_for_piper_↩ landmarks_left	Left_humerus	FMA
Lateral_↩ epicondyle_of_↩ right_femur	RLATEPIC, Femur_FLE_R, FEL_R	most lateral point on lateral epicondyle ; used_for_piper_↩ landmarks_right	Right_femur	FMA, Kepple1997
Lateral_↩ epicondyle_of_↩ right_humerus	Lateral_↩ epicondyle_of_↩ humerus_R, Humerus_HLE_R, EL_R	most caudal point on lateral epicondyle ; used_for_piper_↩ landmarks_right	Right_humerus	FMA
Lateral_left_rib_1		lateral point on the left first rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_left	Left_first_rib	

Lateral_left_rib_10		lateral point on the left tenth rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_left	Left_tenth_rib	
Lateral_left_rib_2		lateral point on the left second rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_left	Left_second_rib	
Lateral_left_rib_3		lateral point on the left third rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_left	Left_third_rib	
Lateral_left_rib_4		lateral point on the left fourth rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_left	Left_fourth_rib	
Lateral_left_rib_5		lateral point on the left fifth rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_left	Left_fifth_rib	
Lateral_left_rib_6		lateral point on the left sixth rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_left	Left_sixth_rib	
Lateral_left_rib_7		lateral point on the left seventh rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_left	Left_seventh_rib	

Lateral_left_rib_8		lateral point on the left eighth rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_left	Left_eighth_rib	
Lateral_left_rib_9		lateral point on the left ninth rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_left	Left_ninth_rib	
Lateral_point_↩ lateral_condyle_↩ of_left_tibia	Lateral_point_↩ lateral_condyle_↩ of_tibia_L, LC_L	; used_for_piper↩ _landmarks_left	Left_tibia	
Lateral_point_↩ lateral_condyle_↩ of_right_tibia	Lateral_point_↩ lateral_condyle_↩ of_tibia_R, LC_R	; used_for_piper↩ _landmarks_right	Right_tibia	
Lateral_point_on↩ _distal_phalanx_↩ of_the_left_↩ auricular	Hand_HL5_L, HL5_L		Skeleton_of_left_↩ hand	
Lateral_point_on↩ _distal_phalanx_↩ of_the_left_thumb	HL1_L, Hand_HL1_L		Skeleton_of_left_↩ hand	
Lateral_point_on↩ _distal_phalanx_↩ of_the_right_↩ auricular	Hand_HL5_R, HL5_R		Skeleton_of_left_↩ hand	
Lateral_point_on↩ _distal_phalanx_↩ of_the_right_thumb	HL1_R, Hand_HL1_R		Skeleton_of_left_↩ hand	
Lateral_point_on↩ _greater_↩ trochanter_of_↩ the_left_femur	FTC_L, Femur_FTC_L	Most lateral point at mid heifght of the greater trochanter ; used_for_piper_↩ landmarks_left	Left_femur	
Lateral_point_on↩ _greater_↩ trochanter_of_↩ the_right_femur	FTC_R, Femur_FTC_R	Most lateral point at mid heifght of the greater trochanter ; used_for_piper_↩ landmarks_right	Right_femur	
Lateral_right_rib↩ _1		lateral point on the right first rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_right	Right_first_rib	

Lateral_right_rib↩ _10		lateral point on the right tenth rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_right	Right_tenth_rib	
Lateral_right_rib↩ _2		lateral point on the right second rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_right	Right_second_rib	
Lateral_right_rib↩ _3		lateral point on the right third rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_right	Right_third_rib	
Lateral_right_rib↩ _4		lateral point on the right fourth rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_right	Right_fourth_rib	
Lateral_right_rib↩ _5		lateral point on the right fifth rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_right	Right_fifth_rib	
Lateral_right_rib↩ _6		lateral point on the right sixth rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_right	Right_sixth_rib	
Lateral_right_rib↩ _7		lateral point on the right seventh rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_right	Right_seventh_rib	

Lateral_right_rib↩ _8		lateral point on the right eighth rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_right	Right_eighth_rib	
Lateral_right_rib↩ _9		lateral point on the right ninth rib (frontal plane between spine and sternum); used_for_piper_↩ landmarks_right	Right_ninth_rib	
Left_Femur_↩ Lesser_Trochanter		in the middle ; used_for_piper_↩ landmarks_left	Left_femur	
Left_acetabular_↩ center	LHJCPEL, Cotyle_L, Ilium_IAC_L	Center of the Left acetabular cup ; used_for_piper_↩ landmarks	Left_hip_bone	Kepple1997
Left_anterior_↩ inferior_iliac_spine	LAIIS, anterior_inferior_↩ iliac_spine_L	Left Anterior Inferior Iliac Spine	Pelvic_skeleton	FMA, Kepple1997
Left_anterior_↩ superior_iliac_↩ spine	LASIS, anterior_superior↩ _iliac_spine_L, Ilium_IAS_L, ASIS_L	Left Anterior Superior Iliac Spine ; used_for_piper↩ _landmarks	Left_hip_bone	FMA, Kepple1997
Left_clavicle_↩ acromioclavicular↩ _joint		center of articular face of acromioclavicular joint on clavicle	Left_clavicle	
Left_clavicle_↩ sternoclavicular_↩ joint		center of articular face of sternoclavicular joint on clavicle	Left_clavicle	
Left_ischial_↩ tuberosity	ischial_tuberosity↩ _L, Ilium_IIT_L	; used_for_piper↩ _landmarks	Left_hip_bone	
Left_jaw_angle	LMK_JAN_L, Jaw_Angle_L	Lower posterior corner of the body and ramus of the jaw ; used_for_↩ piper_landmarks	Mandible	
Left_mastoid_↩ process	LMK_SMP_L, Skull_SMP_L	The mastoid process is found at the posterolateral aspect of the skull. It is a large protuberance for the origin of the sternoc... <Preview truncated at 128 characters> ; used_for_piper_↩ landmarks	Skull	FMA

Left_midpoint_of↩ _the_body_of_C3		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Third_cervical_↩ vertebra	
Left_midpoint_of↩ _the_body_of_C4		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Fourth_cervical_↩ vertebra	
Left_midpoint_of↩ _the_body_of_C5		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Fifth_cervical_↩ vertebra	
Left_midpoint_of↩ _the_body_of_C6		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Sixth_cervical_↩ vertebra	
Left_midpoint_of↩ _the_body_of_C7		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Seventh_cervical↩ _vertebra	
Left_midpoint_of↩ _the_body_of_L1		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	First_lumbar_↩ vertebra	
Left_midpoint_of↩ _the_body_of_L2		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Second_lumbar_↩ vertebra	

Left_midpoint_of↩ _the_body_of_L3		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Third_lumbar_↩ vertebra	
Left_midpoint_of↩ _the_body_of_L4		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Fourth_lumbar_↩ vertebra	
Left_midpoint_of↩ _the_body_of_L5		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Fifth_lumbar_↩ vertebra	
Left_midpoint_of↩ _the_body_of_T1		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	First_thoracic_↩ vertebra	
Left_midpoint_of↩ _the_body_of_T10		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Tenth_thoracic_↩ vertebra	
Left_midpoint_of↩ _the_body_of_T11		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Eleventh_↩ thoracic_vertebra	
Left_midpoint_of↩ _the_body_of_T12		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Twelfth_thoracic_↩ vertebra	

Left_midpoint_of↩ _the_body_of_T2		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Second_thoracic↩ _vertebra	
Left_midpoint_of↩ _the_body_of_T3		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Third_thoracic_↩ vertebra	
Left_midpoint_of↩ _the_body_of_T4		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Fourth_thoracic_↩ vertebra	
Left_midpoint_of↩ _the_body_of_T5		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Fifth_thoracic_↩ vertebra	
Left_midpoint_of↩ _the_body_of_T6		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Sixth_thoracic_↩ vertebra	
Left_midpoint_of↩ _the_body_of_T7		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Seventh_thoracic↩ _vertebra	
Left_midpoint_of↩ _the_body_of_T8		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Eighth_thoracic_↩ vertebra	

Left_midpoint_of↩ _the_body_of_T9		the left point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Ninth_thoracic_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _C2	LMK_C2Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Axis	
Left_point_of_the↩ _inferior_plate_of↩ _C3	LMK_C3Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Third_cervical_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _C4	LMK_C4Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Fourth_cervical_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _C5	LMK_C5Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Fifth_cervical_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _C6	LMK_C6Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Sixth_cervical_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _C7	LMK_C7Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Seventh_cervical↩ _vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _L1	LMK_L1Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	First_lumbar_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _L2	LMK_L2Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Second_lumbar_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _L3	LMK_L3Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Third_lumbar_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _L4	LMK_L4Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Fourth_lumbar_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _L5	LMK_L5Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Fifth_lumbar_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _T1	LMK_T01Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	First_thoracic_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _T10	LMK_T10Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Tenth_thoracic_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _T11	LMK_T11Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Eleventh_↩ thoracic_vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _T12	LMK_T12Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Twelfth_thoracic_↩ vertebra	

Left_point_of_the↩ _inferior_plate_of↩ _T2	LMK_T02Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Second_thoracic↩ _vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _T3	LMK_T03Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Third_thoracic_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _T4	LMK_T04Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Fourth_thoracic_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _T5	LMK_T05Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Fifth_thoracic_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _T6	LMK_T06Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Sixth_thoracic_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _T7	LMK_T07Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Seventh_thoracic↩ _vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _T8	LMK_T08Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Eighth_thoracic_↩ vertebra	
Left_point_of_the↩ _inferior_plate_of↩ _T9	LMK_T09Lower↩ LeftEnd	Left point of the inferior end-plate (largest dimension)	Ninth_thoracic_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_C3	LMK_C3Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Third_cervical_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_C4	LMK_C4Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Fourth_cervical_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_C5	LMK_C5Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Fifth_cervical_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_C6	LMK_C6Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Sixth_cervical_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_C7	LMK_C7Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Seventh_cervical↩ _vertebra	
Left_point_of_the↩ _superior_plate_↩ of_L1	LMK_L1Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	First_lumbar_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_L2	LMK_L2Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Second_lumbar_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_L3	LMK_L3Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Third_lumbar_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_L4	LMK_L4Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Fourth_lumbar_↩ vertebra	

Left_point_of_the↩ _superior_plate_↩ of_L5	LMK_L5Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Fifth_lumbar_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_T1	LMK_T01Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	First_thoracic_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_T10	LMK_T10Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Tenth_thoracic_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_T11	LMK_T11Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Eleventh_↩ thoracic_vertebra	
Left_point_of_the↩ _superior_plate_↩ of_T12	LMK_T12Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Twelfth_thoracic_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_T2	LMK_T02Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Second_thoracic↩ _vertebra	
Left_point_of_the↩ _superior_plate_↩ of_T3	LMK_T03Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Third_thoracic_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_T4	LMK_T04Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Fourth_thoracic_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_T5	LMK_T05Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Fifth_thoracic_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_T6	LMK_T06Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Sixth_thoracic_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_T7	LMK_T07Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Seventh_thoracic↩ _vertebra	
Left_point_of_the↩ _superior_plate_↩ of_T8	LMK_T08Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Eighth_thoracic_↩ vertebra	
Left_point_of_the↩ _superior_plate_↩ of_T9	LMK_T09Upper↩ LeftEnd	Left point of the superior end-plate (largest dimension)	Ninth_thoracic_↩ vertebra	
Left_point_on_↩ the_middle_↩ section_of_the_↩ sternum		is used to define the width of the sternum ; used_for_piper_↩ landmarks	Sternum	
Left_porion		; used_for_piper↩ _landmarks	Skull	
Left_posterior_↩ superior_iliac_↩ spine	LPSIS, posterior↩ _superior_iliac_↩ spine_L, Ilium_IPS_L, PSIS_L	Left Posterior Superior Iliac Spine ; used_for_piper↩ _landmarks	Left_hip_bone	FMA, Kepple1997

Left_scapula_↩ acromioclavicular↩ _joint		center of articular face of acromioclavicular joint on scapula	Left_scapula	
Left_sternum_↩ sternoclavicular_↩ joint	SC1	center of articular face of sternoclavicular joint on sternum ; used_for_piper_↩ landmarks	Sternum	
Left_tibial_↩ tuberosity		; used_for_piper↩ _landmarks_left	Left_tibia	FMA
Left_tip_of_the_↩ spinous_process↩ _of_C1	LMK_C2Spinous↩ _L, CV2_2		Axis	
Left_tip_of_the_↩ spinous_process↩ _of_C3	LMK_C3Spinous↩ _L, CV3_2		Third_cervical_↩ vertebra	
Left_tip_of_the_↩ spinous_process↩ _of_C4	LMK_C4Spinous↩ _L, CV4_2		Fourth_cervical_↩ vertebra	
Left_tip_of_the_↩ spinous_process↩ _of_C5	LMK_C5Spinous↩ _L, CV5_2		Fifth_cervical_↩ vertebra	
Left_tip_of_the_↩ spinous_process↩ _of_C6	LMK_C6Spinous↩ _L, CV6_2		Sixth_cervical_↩ vertebra	
Lower_articular_↩ facet_of_C1_L	LMK_C1Lower↩ LatFacet_L	Lowest most lateral point ont the articular facet ; used_for_piper_↩ landmarks	Atlas	
Lower_articular_↩ facet_of_C1_R	LMK_C1Lower↩ LatFacet_R	Lowest most lateral point ont the articular facet ; used_for_piper_↩ landmarks	Atlas	
Lower_articular_↩ facet_of_C2_L	LMK_C2Lower↩ LatFacet_L	Lowest point ont the articular facet ; used_for_piper_↩ landmarks	Axis	
Lower_articular_↩ facet_of_C2_R	LMK_C2Lower↩ LatFacet_R	Lowest point ont the articular facet ; used_for_piper_↩ landmarks	Axis	
Manubriosternal↩ _joint	LMK_SME	Palpated edge of the manubriosternal joint. Located at the level of the second pair of costal cartilage ; used_for_piper_↩ landmarks	Sternum	

Medial_↩ epicondyle_of_↩ left_femur	LMEDEPIC, Femur_FME_L, FEM_L	most medial point on medial epicondyle ; used_for_piper_↩ landmarks_left	Left_femur	Kepple1997
Medial_↩ epicondyle_of_↩ left_humerus	Medial_↩ epicondyle_of_↩ humerus_L, Humerus_HME_L, EM_L	most caudal point on medial epicondyle ; used_for_piper_↩ landmarks_left	Left_humerus	FMA
Medial_↩ epicondyle_of_↩ right_femur	RMEDEPIC, Femur_FME_R, FEM_R	most medial point on medial epicondyle ; used_for_piper_↩ landmarks_right	Right_femur	FMA, Kepple1997
Medial_↩ epicondyle_of_↩ right_humerus	Medial_↩ epicondyle_of_↩ humerus_R, Humerus_HME_R, EM_R	most caudal point on medial epicondyle ; used_for_piper_↩ landmarks_right	Right_humerus	FMA
Medial_point_↩ medial_condyle_↩ of_left_tibia	LTIMEDPLAT, Medial_point_↩ medial_condyle_↩ of_tibia_L, MC_L	most medial point on medial tibial plateau ; used_for_piper_↩ landmarks_left	Left_tibia	Kepple1997
Medial_point_↩ medial_condyle_↩ of_right_tibia	RTIMEDPLAT, Medial_point_↩ medial_condyle_↩ of_tibia_R, LMK_RTIMEDPL↩ AT, MC_R	most medial point on medial tibial plateau ; used_for_piper_↩ landmarks_right	Right_tibia	Kepple1997
Medial_point_of_↩ the_upper_↩ articular_facet_of↩ _C2_L	LMK_C2Upper↩ MedFacet_L	Medial point of the upper articular facet of C2	Axis	
Medial_point_of_↩ the_upper_↩ articular_facet_of↩ _C2_R	LMK_C2Upper↩ MedFacet_R	Medial point of the upper articular facet of C2	Axis	
Mental_↩ protuberance_of↩ _the_jaw	LMK_JMP	Protuberance at the anterior face of the jaw bone (upper ridge not apex below)	Mandible	
Midpoint_of_↩ intercondylar_↩ line_of_the_left_↩ femur		middle of the intercondylar line at the end of the notch ; used_for_piper_↩ landmarks_left	Left_femur	

Midpoint_of_↩ intercondylar_↩ line_of_the_right↩ _femur		middle of the intercondylar line at the end of the notch ; used_for_piper_↩ landmarks_right	Right_femur	
Most_posterior_↩ point_of_tip_of_↩ spinous_process↩ _of_C2	LMK_CV2, CV2	; used_for_piper↩ _landmarks	Axis	
Most_posterior_↩ point_of_tip_of_↩ spinous_process↩ _of_C3	LMK_CV3, CV3		Third_cervical_↩ vertebra	
Most_posterior_↩ point_of_tip_of_↩ spinous_process↩ _of_C4	LMK_CV4, CV4		Fourth_cervical_↩ vertebra	
Most_posterior_↩ point_of_tip_of_↩ spinous_process↩ _of_C5	LMK_CV5, CV5		Fifth_cervical_↩ vertebra	
Most_posterior_↩ point_of_tip_of_↩ spinous_process↩ _of_C6	LMK_CV6, CV6		Sixth_cervical_↩ vertebra	
Most_posterior_↩ point_of_tip_of_↩ spinous_process↩ _of_C7	LMK_CV7, CV7		Seventh_cervical↩ _vertebra	
Nasion	LMK_SNA	Center of the depression between the orbits ; used_for_piper_↩ landmarks	Skull	
Neck_of_the_left↩ _talus	FNK_L	The neck of the talus is located between the head and the trochlea	Skeleton_of_left_↩ foot	
Neck_of_the_↩ right_talus	FNK_R	The neck of the talus is located between the head and the trochlea	Skeleton_of_↩ right_foot	
PatellaL_anterior↩ _proximal		proximal point on the anterior side of the patella ; used_for_piper_↩ landmarks_left	Left_patella	

PatellaL_↩ posterior_center		the center of the posterior surface of the patella ; used_for_piper_↩ landmarks_left	Left_patella	
PatellaL_↩ posterior_proximal		proximal point on the posterior side of the patella ; used_for_piper_↩ landmarks_left	Left_patella	
PatellaR_anterior↩ _proximal		proximal point on the anterior side of the patella ; used_for_piper_↩ landmarks_right	Right_patella	
PatellaR_↩ posterior_center		the center of the posterior surface of the patella ; used_for_piper_↩ landmarks_right	Right_patella	
PatellaR_↩ posterior_proximal		proximal point on the posterior side of the patella ; used_for_piper_↩ landmarks_right	Right_patella	
Patella_PAX_L	PatellaL_distal	; used_for_piper↩ _landmarks_left	Left_patella	
Patella_PAX_R	PatellaR_distal	; used_for_piper↩ _landmarks_right	Right_patella	
Patella_PCE_L	PatellaL_anterior↩ _center	; used_for_piper↩ _landmarks_left	Left_patella	
Patella_PCE_R	PatellaR_anterior↩ _center	; used_for_piper↩ _landmarks_right	Right_patella	
Patella_PLE_L	PatellaL_lateral	; used_for_piper↩ _landmarks_left	Left_patella	
Patella_PLE_R	PatellaR_lateral	; used_for_piper↩ _landmarks_right	Right_patella	
Patella_PME_L	PatellaL_medial	; used_for_piper↩ _landmarks_left	Left_patella	
Patella_PME_R	PatellaR_medial	; used_for_piper↩ _landmarks_right	Right_patella	
Posterior_Inferior↩ _Iliac_Spine_L	LMK_IPI_L, Posterior_Inferior↩ _Iliac_Spine_Left	Prominence located below the posterior iliac spine ; used_for_piper↩ _landmarks	Left_hip_bone	
Posterior_Inferior↩ _Iliac_Spine_R	LMK_IPI_R, Posterior_Inferior↩ _Iliac_Spine_Right	Prominence located below the posterior iliac spine ; used_for_piper↩ _landmarks	Right_hip_bone	

Posterior_arch_↩ of_C1	LMK_C1PostArch	Most posterior point of the posterior arch ; used_for_piper_↩ landmarks	Atlas	
Posterior_↩ extremity_of_the↩ _left_eighth_rib	Extr_r08L, Extremity_of_left↩ _rib8	point at the posterior extremity of the rib	Left_eighth_rib	
Posterior_↩ extremity_of_the↩ _left_eleventh_rib	Extr_r11L, Extremity_of_left↩ _rib11	point at the posterior extremity of the rib	Left_eleventh_rib	
Posterior_↩ extremity_of_the↩ _left_fifth_rib	Extr_r05L, Extremity_of_left↩ _rib5	point at the posterior extremity of the rib	Left_fifth_rib	
Posterior_↩ extremity_of_the↩ _left_first_rib	Extr_r01L, Extremity_of_left↩ _rib1	point at the posterior extremity of the rib	Left_first_rib	
Posterior_↩ extremity_of_the↩ _left_fourth_rib	Extr_r04L, Extremity_of_left↩ _rib4	point at the posterior extremity of the rib	Left_fourth_rib	
Posterior_↩ extremity_of_the↩ _left_ninth_rib	Extr_r09L, Extremity_of_left↩ _rib9	point at the posterior extremity of the rib	Left_ninth_rib	
Posterior_↩ extremity_of_the↩ _left_second_rib	Extr_r02L, Extremity_of_left↩ _rib2	point at the posterior extremity of the rib	Left_second_rib	
Posterior_↩ extremity_of_the↩ _left_seventh_rib	Extr_r07L, Extremity_of_left↩ _rib7	point at the posterior extremity of the rib	Left_seventh_rib	
Posterior_↩ extremity_of_the↩ _left_sixth_rib	Extr_r06L, Extremity_of_left↩ _rib6	point at the posterior extremity of the rib	Left_sixth_rib	
Posterior_↩ extremity_of_the↩ _left_tenth_rib	Extr_r10L, Extremity_of_left↩ _rib10	point at the posterior extremity of the rib	Left_tenth_rib	
Posterior_↩ extremity_of_the↩ _left_third_rib	Extr_r03L, Extremity_of_left↩ _rib3	point at the posterior extremity of the rib	Left_third_rib	
Posterior_↩ extremity_of_the↩ _left_twelfth_rib	Extr_r12L, Extremity_of_left↩ _rib12	point at the posterior extremity of the rib	Left_twelfth_rib	
Posterior_↩ extremity_of_the↩ _right_eighth_rib	Extr_r08R, Extremity_of_↩ right_rib8	point at the posterior extremity of the rib	Right_eighth_rib	
Posterior_↩ extremity_of_the↩ _right_eleventh_rib	Extr_r11R, Extremity_of_↩ right_rib11	point at the posterior extremity of the rib	Right_eleventh_rib	
Posterior_↩ extremity_of_the↩ _right_fifth_rib	Extr_r05R, Extremity_of_↩ right_rib5	point at the posterior extremity of the rib	Right_fifth_rib	
Posterior_↩ extremity_of_the↩ _right_first_rib	Extr_r01R, Extremity_of_↩ right_rib1	point at the posterior extremity of the rib	Right_first_rib	

Posterior_↩ extremity_of_the↩ _right_fourth_rib	Extr_r04R, Extremity_of_↩ right_rib4	point at the posterior extremity of the rib	Right_fourth_rib	
Posterior_↩ extremity_of_the↩ _right_ninth_rib	Extr_r09R, Extremity_of_↩ right_rib9	point at the posterior extremity of the rib	Right_ninth_rib	
Posterior_↩ extremity_of_the↩ _right_second_rib	Extr_r02R, Extremity_of_↩ right_rib2	point at the posterior extremity of the rib	Right_second_rib	
Posterior_↩ extremity_of_the↩ _right_seventh_rib	Extr_r07R, Extremity_of_↩ right_rib7	point at the posterior extremity of the rib	Right_seventh_rib	
Posterior_↩ extremity_of_the↩ _right_sixth_rib	Extr_r06R, Extremity_of_↩ right_rib6	point at the posterior extremity of the rib	Right_sixth_rib	
Posterior_↩ extremity_of_the↩ _right_tenth_rib	Extr_r10R, Extremity_of_↩ right_rib10	point at the posterior extremity of the rib	Right_tenth_rib	
Posterior_↩ extremity_of_the↩ _right_third_rib	Extr_r03R, Extremity_of_↩ right_rib3	point at the posterior extremity of the rib	Right_third_rib	
Posterior_↩ extremity_of_the↩ _right_twelfth_rib	Extr_r12R, Extremity_of_↩ right_rib12	point at the posterior extremity of the rib	Right_twelfth_rib	
Posterior_↩ intercondylar_↩ area_of_the_left↩ _tibia		insertion of PCL ; used_for_piper_↩ landmarks_left	Left_tibia	
Posterior_↩ intercondylar_↩ area_of_the_↩ right_tibia		insertion of PCL ; used_for_piper_↩ landmarks_right	Right_tibia	
Posterior_↩ midpoint_of_the_↩ body_of_C3		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Third_cervical_↩ vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_C4		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Fourth_cervical_↩ vertebra	

Posterior_↩ midpoint_of_the_↩ body_of_C5		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Fifth_cervical_↩ vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_C6		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Sixth_cervical_↩ vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_C7		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Seventh_cervical↩ _vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_L1		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	First_lumbar_↩ vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_L2		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Second_lumbar_↩ vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_L3		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Third_lumbar_↩ vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_L4		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Fourth_lumbar_↩ vertebra	

Posterior_↩ midpoint_of_the_↩ body_of_L5		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Fifth_lumbar_↩ vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_T1		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	First_thoracic_↩ vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_T10		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Tenth_thoracic_↩ vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_T11		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Eleventh_↩ thoracic_vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_T12		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Twelfth_thoracic_↩ vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_T2		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Second_thoracic↩ _vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_T3		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Third_thoracic_↩ vertebra	

Posterior_↩ midpoint_of_the_↩ body_of_T4		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Fourth_thoracic_↩ vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_T5		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Fifth_thoracic_↩ vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_T6		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Sixth_thoracic_↩ vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_T7		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Seventh_thoracic↩ _vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_T8		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Eighth_thoracic_↩ vertebra	
Posterior_↩ midpoint_of_the_↩ body_of_T9		the posterior point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper↩ _landmarks	Ninth_thoracic_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_C2	LMK_C2Lower↩ RearEnd	Posterior point of the inferior end-plate	Axis	
Posterior_point_↩ of_the_inferior_↩ plate_of_C3	LMK_C3Lower↩ RearEnd	Posterior point of the inferior end-plate	Third_cervical_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_C4	LMK_C4Lower↩ RearEnd	Posterior point of the inferior end-plate	Fourth_cervical_↩ vertebra	

Posterior_point_↩ of_the_inferior_↩ plate_of_C5	LMK_C5Lower↩ RearEnd	Posterior point of the inferior end-plate	Fifth_cervical_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_C6	LMK_C6Lower↩ RearEnd	Posterior point of the inferior end-plate	Sixth_cervical_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_C7	LMK_C7Lower↩ RearEnd	Posterior point of the inferior end-plate	Seventh_cervical↩ _vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_L1	LMK_L1Lower↩ RearEnd	Posterior point of the inferior end-plate	First_lumbar_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_L2	LMK_L2Lower↩ RearEnd	Posterior point of the inferior end-plate	Second_lumbar_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_L3	LMK_L3Lower↩ RearEnd	Posterior point of the inferior end-plate	Third_lumbar_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_L4	LMK_L4Lower↩ RearEnd	Posterior point of the inferior end-plate	Fourth_lumbar_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_L5	LMK_L5Lower↩ RearEnd	Posterior point of the inferior end-plate	Fifth_lumbar_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_T1	LMK_T01Lower↩ RearEnd	Posterior point of the inferior end-plate	First_thoracic_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_T10	LMK_T10Lower↩ RearEnd	Posterior point of the inferior end-plate	Tenth_thoracic_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_T11	LMK_T11Lower↩ RearEnd	Posterior point of the inferior end-plate	Eleventh_↩ thoracic_vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_T12	LMK_T12Lower↩ RearEnd	Posterior point of the inferior end-plate	Twelfth_thoracic_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_T2	LMK_T02Lower↩ RearEnd	Posterior point of the inferior end-plate	Second_thoracic↩ _vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_T3	LMK_T03Lower↩ RearEnd	Posterior point of the inferior end-plate	Third_thoracic_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_T4	LMK_T04Lower↩ RearEnd	Posterior point of the inferior end-plate	Fourth_thoracic_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_T5	LMK_T05Lower↩ RearEnd	Posterior point of the inferior end-plate	Fifth_thoracic_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_T6	LMK_T06Lower↩ RearEnd	Posterior point of the inferior end-plate	Sixth_thoracic_↩ vertebra	

Posterior_point_↩ of_the_inferior_↩ plate_of_T7	LMK_T07Lower↩ RearEnd	Posterior point of the inferior end-plate	Seventh_thoracic↩ _vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_T8	LMK_T08Lower↩ RearEnd	Posterior point of the inferior end-plate	Eighth_thoracic_↩ vertebra	
Posterior_point_↩ of_the_inferior_↩ plate_of_T9	LMK_T09Lower↩ RearEnd	Posterior point of the inferior end-plate	Ninth_thoracic_↩ vertebra	
Posterior_point_↩ of_the_sacral_plate	DS, Most_posterior_↩ point_sacral_plate	; used_for_piper↩ _landmarks	Pelvic_skeleton	
Posterior_point_↩ of_the_superior_↩ plate_of_C3	LMK_C3Upper↩ RearEnd	Posterior point of the superior end-plate	Third_cervical_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_C4	LMK_C4Upper↩ RearEnd	Posterior point of the superior end-plate	Fourth_cervical_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_C5	LMK_C5Upper↩ RearEnd	Posterior point of the superior end-plate	Fifth_cervical_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_C6	LMK_C6Upper↩ RearEnd	Posterior point of the superior end-plate	Sixth_cervical_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_C7	LMK_C7Upper↩ RearEnd	Posterior point of the superior end-plate	Seventh_cervical↩ _vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_L1	LMK_L1Upper↩ RearEnd	Posterior point of the superior end-plate	First_lumbar_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_L2	LMK_L2Upper↩ RearEnd	Posterior point of the superior end-plate	Second_lumbar_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_L3	LMK_L3Upper↩ RearEnd	Posterior point of the superior end-plate	Third_lumbar_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_L4	LMK_L4Upper↩ RearEnd	Posterior point of the superior end-plate	Fourth_lumbar_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_L5	LMK_L5Upper↩ RearEnd	Posterior point of the superior end-plate	Fifth_lumbar_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_T1	LMK_T01Upper↩ RearEnd	Posterior point of the superior end-plate	First_thoracic_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_T10	LMK_T10Upper↩ RearEnd	Posterior point of the superior end-plate	Tenth_thoracic_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_T11	LMK_T11Upper↩ RearEnd	Posterior point of the superior end-plate	Eleventh_↩ thoracic_vertebra	

Posterior_point_↩ of_the_superior_↩ plate_of_T12	LMK_T12Upper↩ RearEnd	Posterior point of the superior end-plate	Twelfth_thoracic_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_T2	LMK_T02Upper↩ RearEnd	Posterior point of the superior end-plate	Second_thoracic↩ _vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_T3	LMK_T03Upper↩ RearEnd	Posterior point of the superior end-plate	Third_thoracic_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_T4	LMK_T04Upper↩ RearEnd	Posterior point of the superior end-plate	Fourth_thoracic_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_T5	LMK_T05Upper↩ RearEnd	Posterior point of the superior end-plate	Fifth_thoracic_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_T6	LMK_T06Upper↩ RearEnd	Posterior point of the superior end-plate	Sixth_thoracic_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_T7	LMK_T07Upper↩ RearEnd	Posterior point of the superior end-plate	Seventh_thoracic↩ _vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_T8	LMK_T08Upper↩ RearEnd	Posterior point of the superior end-plate	Eighth_thoracic_↩ vertebra	
Posterior_point_↩ of_the_superior_↩ plate_of_T9	LMK_T09Upper↩ RearEnd	Posterior point of the superior end-plate	Ninth_thoracic_↩ vertebra	
Pubic_symphysis	LMK_IPJ_M, Ilium_IPJ	Middle point between left and right anterior angles of the pubic joint	Right_hip_bone	
Pubic_symphysis↩ _of_the_left_hip	LMK_IPJ_L	Pubic symphisis	Left_hip_bone	
Pubic_symphysis↩ _of_the_right_hip	LMK_IPJ_R	Pubic symphisis	Pelvic_skeleton	
Radius_RDE_L			Left_radius	
Radius_RDE_R			Right_radius	
Ribs_RL10_L			Left_tenth_rib	
Ribs_RL10_R			Right_tenth_rib	
Ribs_RL8_L			Left_eighth_rib	
Ribs_RL8_R			Right_eighth_rib	
Ribs_RL9_L			Left_ninth_rib	
Ribs_RL9_R			Right_ninth_rib	
Right_Femur_↩ Lesser_Trochanter		in the middle ; used_for_piper_↩ landmarks_right	Right_femur	
Right_acetabular↩ _center	RHJCPEL, Cotyle_R, Ilium_IAC_R	Center of the Right acetabular cup ; used_for_piper_↩ landmarks	Right_hip_bone	Kepple1997

Right_anterior_↩ inferior_iliac_spine	RAIIS, anterior_inferior_↩ iliac_spine_R	Right Anterior Inferior Iliac Spine	Pelvic_skeleton	FMA, Kepple1997
Right_anterior_↩ superior_iliac_↩ spine	RASIS, anterior_superior↩ _iliac_spine_R, Ilium_IAS_R, ASIS_R	Right Anterior Superior Iliac Spine ; used_for_piper↩ _landmarks	Right_hip_bone	FMA, Kepple1997
Right_clavicle_↩ acromioclavicular↩ _joint		center of articular face of acromioclavicular joint on clavicle	Right_clavicle	
Right_clavicle_↩ sternoclavicular_↩ joint		center of articular face of sternoclavicular joint on clavicle	Right_clavicle	
Right_ischial_↩ tuberosity	ischial_tuberosity↩ _R, Ilium_IIT_R	; used_for_piper↩ _landmarks	Right_hip_bone	
Right_jaw_angle	LMK_JAN_R, Jaw_Angle_R	Lower posterior corner of the body and ramus of the jaw ; used_for_↩ piper_landmarks	Mandible	
Right_mastoid_↩ process	LMK_SMP_R, Skull_SMP_R	The mastoid process is found at the posterolateral aspect of the skull. It is a large protuberance for the origin of the sternoc... <Preview truncated at 128 characters> ; used_for_piper_↩ landmarks	Skull	FMA
Right_midpoint_↩ of_the_body_of_C3		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Third_cervical_↩ vertebra	
Right_midpoint_↩ of_the_body_of_C4		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Fourth_cervical_↩ vertebra	

Right_midpoint_↩ of_the_body_of_C5		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Fifth_cervical_↩ vertebra	
Right_midpoint_↩ of_the_body_of_C6		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Sixth_cervical_↩ vertebra	
Right_midpoint_↩ of_the_body_of_C7		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Seventh_cervical↩ _vertebra	
Right_midpoint_↩ of_the_body_of_L1		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	First_lumbar_↩ vertebra	
Right_midpoint_↩ of_the_body_of_L2		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Second_lumbar_↩ vertebra	
Right_midpoint_↩ of_the_body_of_L3		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Third_lumbar_↩ vertebra	
Right_midpoint_↩ of_the_body_of_L4		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Fourth_lumbar_↩ vertebra	

Right_midpoint_↩ of_the_body_of_L5		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Fifth_lumbar_↩ vertebra	
Right_midpoint_↩ of_the_body_of_T1		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	First_thoracic_↩ vertebra	
Right_midpoint_↩ of_the_body_of_↩ T10		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Tenth_thoracic_↩ vertebra	
Right_midpoint_↩ of_the_body_of_↩ T11		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Eleventh_↩ thoracic_vertebra	
Right_midpoint_↩ of_the_body_of_↩ T12		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Twelfth_thoracic_↩ vertebra	
Right_midpoint_↩ of_the_body_of_T2		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Second_thoracic↩ _vertebra	
Right_midpoint_↩ of_the_body_of_T3		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Third_thoracic_↩ vertebra	

Right_midpoint_↩ of_the_body_of_T4		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Fourth_thoracic_↩ vertebra	
Right_midpoint_↩ of_the_body_of_T5		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Fifth_thoracic_↩ vertebra	
Right_midpoint_↩ of_the_body_of_T6		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Sixth_thoracic_↩ vertebra	
Right_midpoint_↩ of_the_body_of_T7		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Seventh_thoracic↩ _vertebra	
Right_midpoint_↩ of_the_body_of_T8		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Eighth_thoracic_↩ vertebra	
Right_midpoint_↩ of_the_body_of_T9		the right point on the midplane of the upper and lower vertebral main body surfaces ; used_for_piper_↩ landmarks	Ninth_thoracic_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_C2	LMK_C2Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Axis	
Right_point_of_↩ the_inferior_plate↩ _of_C3	LMK_C3Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Third_cervical_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_C4	LMK_C4Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Fourth_cervical_↩ vertebra	

Right_point_of_↩ the_inferior_plate↩ _of_C5	LMK_C5Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Fifth_cervical_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_C6	LMK_C6Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Sixth_cervical_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_C7	LMK_C7Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Seventh_cervical↩ _vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_L1	LMK_L1Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	First_lumbar_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_L2	LMK_L2Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Second_lumbar_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_L3	LMK_L3Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Third_lumbar_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_L4	LMK_L4Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Fourth_lumbar_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_L5	LMK_L5Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Fifth_lumbar_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_T1	LMK_T01Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	First_thoracic_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_T10	LMK_T10Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Tenth_thoracic_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_T11	LMK_T11Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Eleventh_↩ thoracic_vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_T12	LMK_T12Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Twelfth_thoracic_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_T2	LMK_T02Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Second_thoracic↩ _vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_T3	LMK_T03Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Third_thoracic_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_T4	LMK_T04Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Fourth_thoracic_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_T5	LMK_T05Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Fifth_thoracic_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_T6	LMK_T06Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Sixth_thoracic_↩ vertebra	

Right_point_of_↩ the_inferior_plate↩ _of_T7	LMK_T07Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Seventh_thoracic↩ _vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_T8	LMK_T08Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Eighth_thoracic_↩ vertebra	
Right_point_of_↩ the_inferior_plate↩ _of_T9	LMK_T09Lower↩ RightEnd	Right point of the inferior end-plate (largest dimension)	Ninth_thoracic_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_C3	LMK_C3Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Third_cervical_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_C4	LMK_C4Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Fourth_cervical_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_C5	LMK_C5Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Fifth_cervical_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_C6	LMK_C6Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Sixth_cervical_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_C7	LMK_C7Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Seventh_cervical↩ _vertebra	
Right_point_of_↩ the_superior_↩ plate_of_L1	LMK_L1Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	First_lumbar_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_L2	LMK_L2Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Second_lumbar_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_L3	LMK_L3Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Third_lumbar_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_L4	LMK_L4Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Fourth_lumbar_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_L5	LMK_L5Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Fifth_lumbar_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_T1	LMK_T01Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	First_thoracic_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_T10	LMK_T10Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Tenth_thoracic_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_T11	LMK_T11Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Eleventh_↩ thoracic_vertebra	
Right_point_of_↩ the_superior_↩ plate_of_T12	LMK_T12Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Twelfth_thoracic_↩ vertebra	

Right_point_of_↩ the_superior_↩ plate_of_T2	LMK_T02Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Second_thoracic↩ _vertebra	
Right_point_of_↩ the_superior_↩ plate_of_T3	LMK_T03Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Third_thoracic_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_T4	LMK_T04Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Fourth_thoracic_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_T5	LMK_T05Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Fifth_thoracic_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_T6	LMK_T06Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Sixth_thoracic_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_T7	LMK_T07Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Seventh_thoracic↩ _vertebra	
Right_point_of_↩ the_superior_↩ plate_of_T8	LMK_T08Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Eighth_thoracic_↩ vertebra	
Right_point_of_↩ the_superior_↩ plate_of_T9	LMK_T09Upper↩ RightEnd	Right point of the superior end-plate (largest dimension)	Ninth_thoracic_↩ vertebra	
Right_point_on_↩ the_middle_↩ section_of_the_↩ sternum		is used to define the width of the sternum ; used_for_piper_↩ landmarks	Sternum	
Right_porion		; used_for_piper↩ _landmarks	Skull	
Right_posterior_↩ superior_iliac_↩ spine	RPSIS, posterior↩ _superior_iliac_↩ spine_R, Ilium_IPS_R, PSIS_R	Right Posterior Superior Iliac Spine ; used_for_piper↩ _landmarks	Right_hip_bone	FMA, Kepple1997
Right_scapula_↩ acromioclavicular↩ _joint		center of articular face of acromioclavicular joint on scapula	Right_scapula	
Right_sternum_↩ sternoclavicular_↩ joint	SC2	center of articular face of sternoclavicular joint on sternum ; used_for_piper_↩ landmarks	Sternum	
Right_tibial_↩ tuberosity	RTIATUB, tibial_tuberosity_R, Tibia_TTC_R, TT_R	most anterior point on tibial tubercle ; used_for_piper_↩ landmarks_right	Right_tibia	FMA, Kepple1997

Right_tip_of_the↩ _spinous_↩ process_of_C1	LMK_C2Spinous↩ _R, CV2_1		Axis	
Right_tip_of_the↩ _spinous_↩ process_of_C3	LMK_C3Spinous↩ _R, CV3_1		Third_cervical_↩ vertebra	
Right_tip_of_the↩ _spinous_↩ process_of_C4	LMK_C4Spinous↩ _R, CV4_1		Fourth_cervical_↩ vertebra	
Right_tip_of_the↩ _spinous_↩ process_of_C5	LMK_C5Spinous↩ _R, CV5_1		Fifth_cervical_↩ vertebra	
Right_tip_of_the↩ _spinous_↩ process_of_C6	LMK_C6Spinous↩ _R, CV6_1		Sixth_cervical_↩ vertebra	
SC3			Sternum	
Sacral_hiatus		Ridge next to the sacral hiatus	Sacrum	
Scapula_SRS_L			Left_scapula	
Scapula_SRS_R			Right_scapula	
Sigmoid_notch_↩ of_left_radius	Sigmoid_notch_↩ of_radius_L, Radius_RUN_L	; used_for_piper↩ _landmarks_left	Left_radius	
Sigmoid_notch_↩ of_right_radius	Sigmoid_notch_↩ of_radius_R, Radius_RUN_R	; used_for_piper↩ _landmarks_right	Right_radius	
Skull_SLE_L	Skull_lower_↩ edge_of_orbit_L	; used_for_piper↩ _landmarks	Skull	
Skull_SLE_R	Skull_lower_↩ edge_of_orbit_R	; used_for_piper↩ _landmarks	Skull	
Skull_back		; used_for_piper↩ _landmarks	Skull	
Skull_lateral_left		; used_for_piper↩ _landmarks	Skull	
Skull_lateral_right		; used_for_piper↩ _landmarks	Skull	
Skull_top		; used_for_piper↩ _landmarks	Skull	
Spinous_process↩ _of_second_↩ sacral_vertebra	LMK_SS2	Second posterior eminence on the posterior aspect of the sacrum	Sacrum	
Styloid_process_↩ of_left_radius	Styloid_process_↩ of_radius_L, Radius_RSP_L, RS_L	Most caudal-lateral point on the radial styloid ; used_for_piper_↩ landmarks_left	Left_radius	FMA
Styloid_process_↩ of_left_ulna	Styloid_process_↩ of_ulnar_L, Ulna_USP_L, US_L	Most caudal-medialoint on the ulnar styloid ; used_for_piper↩ _landmarks_left	Left_ulna	FMA

Styloid_process_↩ of_right_radius	Styloid_process_↩ of_radius_R, Radius_RSP_R, RS_R	Most caudal-lateral point on the radial styloid ; used_for_piper_↩ landmarks_right	Right_radius	FMA
Styloid_process_↩ of_right_ulna	Styloid_process_↩ of_ulnar_R, Ulna_USP_R, US_R	Most caudal-medial point on the ulnar styloid ; used_for_piper_↩ landmarks_right	Right_ulna	FMA
Substernale_↩ (xyphoid_process)	LMK_SXS	Joint that separates the sternal body and xiphoid process (normally at the level of the 7th and last pair of ribs that articulat... <Preview truncated at 128 characters> ; used_for_piper_↩ landmarks	Sternum	
Superior_angle_↩ of_left_scapula	SSA_L	; used_for_piper↩ _landmarks_left	Left_scapula	
Superior_angle_↩ of_right_scapula	SSA_R	; used_for_piper↩ _landmarks_right	Right_scapula	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_C3	DC3	; used_for_piper↩ _landmarks	Third_cervical_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_C4	DC4	; used_for_piper↩ _landmarks	Fourth_cervical_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_C5	DC5	; used_for_piper↩ _landmarks	Fifth_cervical_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_C6	DC6	; used_for_piper↩ _landmarks	Sixth_cervical_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_C7	DC7	; used_for_piper↩ _landmarks	Seventh_cervical↩ _vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_L1	DL1	; used_for_piper↩ _landmarks	First_lumbar_↩ vertebra	

Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_L2	DL2	; used_for_piper↩ _landmarks	Second_lumbar_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_L3	DL3	; used_for_piper↩ _landmarks	Third_lumbar_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_L4	DL4	; used_for_piper↩ _landmarks	Fourth_lumbar_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_L5	DL5	; used_for_piper↩ _landmarks	Fifth_lumbar_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_T1	DT01	; used_for_piper↩ _landmarks	First_thoracic_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_T10	DT10	; used_for_piper↩ _landmarks	Tenth_thoracic_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_T11	DT11	; used_for_piper↩ _landmarks	Eleventh_↩ thoracic_vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_T12	DT12	; used_for_piper↩ _landmarks	Twelfth_thoracic_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_T2	DT02	; used_for_piper↩ _landmarks	Second_thoracic↩ _vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_T3	DT03	; used_for_piper↩ _landmarks	Third_thoracic_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_T4	DT04	; used_for_piper↩ _landmarks	Fourth_thoracic_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_T5	DT05	; used_for_piper↩ _landmarks	Fifth_thoracic_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_T6	DT06	; used_for_piper↩ _landmarks	Sixth_thoracic_↩ vertebra	

Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_T7	DT07	; used_for_piper↩ _landmarks	Seventh_thoracic↩ _vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_T8	DT08	; used_for_piper↩ _landmarks	Eighth_thoracic_↩ vertebra	
Superior_point_↩ of_tip_of_↩ spinous_process↩ _of_T9	DT09	; used_for_piper↩ _landmarks	Ninth_thoracic_↩ vertebra	
Supraorbital_↩ notch_L	LMK_SSN_L	Supraorbital notch ; used_for_piper_↩ landmarks	Skull	
Supraorbital_↩ notch_R	LMK_SSN_R	Supraorbital notch ; used_for_piper_↩ landmarks	Skull	
Suprasternale_↩ (manubrium)/↩ Jugular_notch	LMK_SJN, Jugular_notch	Superior margin of the jugular notch of the manubrium on the midline of the sternum ; used_for_piper_↩ landmarks	Sternum	
Tibia_TGT_L			Left_tibia	
Tibia_TGT_R			Right_tibia	
Tibia_TLR_L			Left_tibia	
Tibia_TLR_R			Right_tibia	
Tibia_TMR_L			Left_tibia	
Tibia_TMR_R			Right_tibia	
Tibia_TTC_L			Left_tibia	
Tibia_TTL_L			Left_tibia	
Tibia_TTL_R			Right_tibia	
Tibia_TTM_L			Left_tibia	
Tibia_TTM_R			Right_tibia	
Tip_greater_↩ trochanter_of_↩ left_femur		use middle from side view ; used_for_piper_↩ landmarks_left	Left_femur	
Tip_greater_↩ trochanter_of_↩ right_femur		use middle from side view ; used_for_piper_↩ landmarks_right	Right_femur	
Tip_lateral_↩ malleolus_of_left↩ _fibula	Tip_lateral_↩ malleolus_of_↩ fibula_L, FAL_L	Tip of the malleolus of the fibula ; used_for_piper_↩ landmarks_left	Left_fibula	

Tip_lateral_↩ malleolus_of_left↩ _fibula_on_skin	Tip_lateral_↩ malleolus_of_↩ fibula_L_onSkin, FAL_L_onSkin, LM_L	Tip of the malleolus of the fibula identified with palpation		
Tip_lateral_↩ malleolus_of_↩ right_fibula	Tip_lateral_↩ malleolus_of_↩ fibula_R, FAL_R	Tip of the malleolus of the fibula ; used_for_piper_↩ landmarks_right	Right_fibula	
Tip_lateral_↩ malleolus_of_↩ right_fibula_on_↩ skin	Tip_lateral_↩ malleolus_of_↩ fibula_R_onSkin, FAL_R_onSkin, LM_R	Tip of the malleolus of the fibula identified with palpation		
Tip_medial_↩ malleolus_of_left↩ _tibia	Apex_of_the_↩ medial_malleolus↩ _of_the_left_tibia, Tip_medial_↩ malleolus_of_↩ tibia_L, Tibia_TAM_L, MM_L	; used_for_piper↩ _landmarks_left	Left_tibia	
Tip_medial_↩ malleolus_of_↩ right_tibia	Apex_of_the_↩ medial_malleolus↩ _of_the_right_tibia, Tip_medial_↩ malleolus_of_↩ tibia_R, Tibia_TAM_R, MM_R	; used_for_piper↩ _landmarks_right	Right_tibia	
Tip_of_the_↩ odeontoid/dens_↩ of_C2	LMK_C2Tip	Tip of the odeontoid of C2 ; used_for_piper_↩ landmarks	Axis	
Trigonum_spinae↩ _of_left_scapula	Trigonum_spinae↩ _of_scapula_L, TS_L	midpoint of the triangular surface on the medial border of the scapula n line with the scapula spine ; used_for_piper_↩ landmarks_left	Left_scapula	
Trigonum_spinae↩ _of_right_scapula	Trigonum_spinae↩ _of_scapula_R, TS_R	midpoint of the triangular surface on the medial border of the scapula n line with the scapula spine ; used_for_piper_↩ landmarks_right	Right_scapula	

Tuberosity_of_↩ the_anterior_↩ arch_of_C1	LMK_C1AntArch	Most anterior point of the anterior arch ; used_for_piper↩ _landmarks	Atlas	
Ulna_UCP_L	Coronoid_↩ process_of_left_↩ ulna	; used_for_piper↩ _landmarks_left	Left_ulna	
Ulna_UCP_R	Coronoid_↩ process_of_right↩ _ulna	; used_for_piper↩ _landmarks_right	Right_ulna	
Ulna_UHD_L			Left_ulna	
Ulna_UHD_R			Right_ulna	
Ulna_UHE_L			Left_ulna	
Ulna_UHE_R			Right_ulna	
Ulna_UOA_L	Olecranon_apex↩ _of_the_left_ulna	; used_for_piper↩ _landmarks_left	Left_ulna	
Ulna_UOA_R	Olecranon_apex↩ _of_the_right_ulna	; used_for_piper↩ _landmarks_right	Right_ulna	
Ulna_UOB_L			Left_ulna	
Ulna_UOB_R			Right_ulna	
Ulna_UOM_L			Left_ulna	
Ulna_UOM_R			Right_ulna	
Ulna_URU_L			Left_ulna	
Ulna_URU_R			Right_ulna	
Upper_edge_of_↩ the_concave_of_↩ the_left__↩ clavicular_surface	LMK_SCS_L	Upper edge of the concave clavicular surface ; used_for_piper_↩ landmarks	Sternum	
Upper_edge_of_↩ the_concave_of_↩ the_right__↩ clavicular_surface	LMK_SCS_R	Upper edge of the concave clavicular surface ; used_for_piper_↩ landmarks	Sternum	
Upper_point_of_↩ patellar_surface_↩ groove_of_the_↩ left_femur		upper limit of the groove on the patellar surface of the femur ; used_for_piper_↩ landmarks_left	Left_femur	
Upper_point_of_↩ patellar_surface_↩ groove_of_the_↩ right_femur		upper limit of the groove on the patellar surface of the femur ; used_for_piper_↩ landmarks_right	Right_femur	

VC1		; used_for_piper↩ _landmarks	Atlas	
VC2			Axis	
Ventral_point_of↩ _left_↩ sternoclavicular_↩ joint	LMK_CAS_L, Ventral_point_of↩ _sternoclavicular↩ _joint_L, Clavicle_CAS_L, SC_L	Most ventral point on the sternoclavicular joint ; used_for_piper_↩ landmarks_left	Left_clavicle	
Ventral_point_of↩ _right_↩ sternoclavicular_↩ joint	LMK_CAS_R, Ventral_point_of↩ _sternoclavicular↩ _joint_R, Clavicle_CAS_R, SC_R	Most ventral point on the sternoclavicular joint ; used_for_piper_↩ landmarks_right	Right_clavicle	
Zygomatic_Angle↩ _L	LMK_SZA_L	Lower anterior corner of the zygomatic arch ; used_for_piper_↩ landmarks	Skull	
Zygomatic_Angle↩ _R	LMK_SZA_R	Lower anterior corner of the zygomatic arch ; used_for_piper_↩ landmarks	Skull	
elbow_joint_↩ radius_L			Left_radius	
elbow_joint_↩ radius_R			Left_radius	
knee_joint_tibia_L			Left_tibia	
knee_joint_tibia_R			Right_tibia	
rotule_scapula_L	Center_of_↩ scapular_↩ articulated_↩ surface_to_the_↩ left_humerus	; used_for_piper↩ _landmarks_left	Left_scapula	
rotule_scapula_R	Center_of_↩ scapular_↩ articulated_↩ surface_to_the_↩ right_humerus	; used_for_piper↩ _landmarks_right	Right_scapula	

17 Appendix: List of frames

Frame name	Synonyms	Description	PartOf	Landmarks	Bibliography
C1_C2_arch↩ _frame			Axis	[Center_↩ upper_plate_↩ of_C2, Center_lower↩ _plate_of_C1]	ISB_JCS

C1_Skull_↩ frame			Atlas	[Center_↩ atlanto_↩ occipital_joint, Center_↩ atlanto_↩ occipital_joint]	ISB_JCS
C1_frame	Atlas_frame		Atlas	[Tuberosity_↩ of_the_↩ anterior_arch↩ _of_C1, Posterior_↩ arch_of_C1, Lower_↩ articular_↩ facet_of_C1_R, Lower_↩ articular_↩ facet_of_C1_L]	ISB_BCS
C2_C1_arch↩ _frame			Atlas	[Center_↩ lower_plate_↩ of_C1, Center_↩ upper_plate_↩ of_C2]	ISB_JCS
C2_C3_arch↩ _frame			Third_↩ cervical_↩ vertebra	[Center_↩ upper_plate_↩ of_C3, Center_lower↩ _plate_of_C2]	ISB_JCS
C2_frame	Axis_frame		Axis	[Lower_↩ articular_↩ facet_of_C2_R, Lower_↩ articular_↩ facet_of_C2_L, Center_↩ upper_plate_↩ of_C2, Center_lower↩ _plate_of_C2] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_C2, Left_point_of↩ _the_inferior↩ _plate_of_C2, Center_↩ upper_plate_↩ of_C2, Center_lower↩ _plate_of_C2] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_C2, Left_point_of↩ _the_↩ superior_↩ plate_of_C2,	ISB_BCS
				Center_↩ upper_plate_↩ of_C2, Center_lower↩	

C3_C2_arch↩ _frame			Axis	[Center_↩ lower_plate_↩ of_C2, Center_↩ upper_plate_↩ of_C3]	ISB_JCS
C3_C4_arch↩ _frame			Fourth_↩ cervical_↩ vertebra	[Center_↩ upper_plate_↩ of_C4, Center_lower↩ _plate_of_C3]	ISB_JCS
C3_frame	Third_↩ cervical_↩ vertebra_frame		Third_↩ cervical_↩ vertebra	[Lower_↩ articular_↩ facet_of_C3_R, Lower_↩ articular_↩ facet_of_C3_L, Center_↩ upper_plate_↩ of_C3, Center_lower↩ _plate_of_C3] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_C3, Left_point_of↩ _the_inferior↩ _plate_of_C3, Center_↩ upper_plate_↩ of_C3, Center_lower↩ _plate_of_C3] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_C3, Left_point_of↩ _the_↩ superior_↩ plate_of_C3, Center_↩ upper_plate_↩ of_C3, Center_lower↩ _plate_of_C3]	ISB_BCS

C4_C3_arch↩ _frame			Third_↩ cervical_↩ vertebra	[Center_↩ lower_plate_↩ of_C3, Center_↩ upper_plate_↩ of_C4]	ISB_JCS
C4_C5_arch↩ _frame			Fifth_cervical↩ _vertebra	[Center_↩ upper_plate_↩ of_C5, Center_lower↩ _plate_of_C4]	ISB_JCS
C4_frame	Fourth_↩ cervical_↩ vertebra_frame		Fourth_↩ cervical_↩ vertebra	[Lower_↩ articular_↩ facet_of_C4_R, Lower_↩ articular_↩ facet_of_C4_L, Center_↩ upper_plate_↩ of_C4, Center_lower↩ _plate_of_C4] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_C4, Left_point_of↩ _the_inferior↩ _plate_of_C4, Center_↩ upper_plate_↩ of_C4, Center_lower↩ _plate_of_C4] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_C4, Left_point_of↩ _the_↩ superior_↩ plate_of_C4, Center_↩ upper_plate_↩ of_C4, Center_lower↩ _plate_of_C4]	ISB_BCS

C5_C4_arch↩ _frame			Fourth_↩ cervical_↩ vertebra	[Center_↩ lower_plate_↩ of_C4, Center_↩ upper_plate_↩ of_C5]	ISB_JCS
C5_C6_arch↩ _frame			Sixth_↩ cervical_↩ vertebra	[Center_↩ upper_plate_↩ of_C6, Center_lower↩ _plate_of_C5]	ISB_JCS
C5_frame	Fifth_cervical↩ _vertebra_↩ frame		Fifth_cervical↩ _vertebra	[Lower_↩ articular_↩ facet_of_C5_R, Lower_↩ articular_↩ facet_of_C5_L, Center_↩ upper_plate_↩ of_C5, Center_lower↩ _plate_of_C5] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_C5, Left_point_of↩ _the_inferior↩ _plate_of_C5, Center_↩ upper_plate_↩ of_C5, Center_lower↩ _plate_of_C5] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_C5, Left_point_of↩ _the_↩ superior_↩ plate_of_C5, Center_↩ upper_plate_↩ of_C5, Center_lower↩ _plate_of_C5]	ISB_BCS

C6_C5_arch↩ _frame			Fifth_cervical↩ _vertebra	[Center_↩ lower_plate_↩ of_C5, Center_↩ upper_plate_↩ of_C6]	ISB_JCS
C6_C7_arch↩ _frame			Seventh_↩ cervical_↩ vertebra	[Center_↩ upper_plate_↩ of_C7, Center_lower↩ _plate_of_C6]	ISB_JCS
C6_frame	Sixth_↩ cervical_↩ vertebra_frame		Sixth_↩ cervical_↩ vertebra	[Lower_↩ articular_↩ facet_of_C6_R, Lower_↩ articular_↩ facet_of_C6_L, Center_↩ upper_plate_↩ of_C6, Center_lower↩ _plate_of_C6] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_C6, Left_point_of↩ _the_inferior↩ _plate_of_C6, Center_↩ upper_plate_↩ of_C6, Center_lower↩ _plate_of_C6] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_C6, Left_point_of↩ _the_↩ superior_↩ plate_of_C6, Center_↩ upper_plate_↩ of_C6, Center_lower↩ _plate_of_C6]	ISB_BCS

C7_C6_arch↩ _frame			Sixth_↩ cervical_↩ vertebra	[Center_↩ lower_plate_↩ of_C6, Center_↩ upper_plate_↩ of_C7]	ISB_JCS
C7_T1_arch↩ _frame			First_thoracic↩ _vertebra	[Center_↩ upper_plate_↩ of_T1, Center_lower↩ _plate_of_C7]	ISB_JCS
C7_frame	Seventh_↩ cervical_↩ vertebra_frame		Seventh_↩ cervical_↩ vertebra	[Lower_↩ articular_↩ facet_of_C7_R, Lower_↩ articular_↩ facet_of_C7_L, Center_↩ upper_plate_↩ of_C7, Center_lower↩ _plate_of_C7] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_C7, Left_point_of↩ _the_inferior↩ _plate_of_C7, Center_↩ upper_plate_↩ of_C7, Center_lower↩ _plate_of_C7] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_C7, Left_point_of↩ _the_↩ superior_↩ plate_of_C7, Center_↩ upper_plate_↩ of_C7, Center_lower↩ _plate_of_C7]	ISB_BCS

L1_L2_arch_↩ frame			Second_↩ lumbar_↩ vertebra	[Center_↩ upper_plate_↩ of_L2, Center_lower↩ _plate_of_L1]	ISB_JCS
L1_T12_arch↩ _frame			Twelfth_↩ thoracic_↩ vertebra	[Center_↩ lower_plate_↩ of_T12, Center_↩ upper_plate_↩ of_L1]	ISB_JCS
L1_frame	First_lumbar↩ _vertebra_↩ frame		First_lumbar↩ _vertebra	[Lower_↩ articular_↩ facet_of_L1_R, Lower_↩ articular_↩ facet_of_L1_L, Center_↩ upper_plate_↩ of_L1, Center_lower↩ _plate_of_L1] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_L1, Left_point_of↩ _the_inferior↩ _plate_of_L1, Center_↩ upper_plate_↩ of_L1, Center_lower↩ _plate_of_L1] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_L1, Left_point_of↩ _the_↩ superior_↩ plate_of_L1, Center_↩ upper_plate_↩ of_L1, Center_lower↩ _plate_of_L1]	ISB_BCS

L2_L1_arch_↩ frame			First_lumbar↩ _vertebra	[Center_↩ lower_plate_↩ of_L1, Center_↩ upper_plate_↩ of_L2]	ISB_JCS
L2_L3_arch_↩ frame			Third_lumbar↩ _vertebra	[Center_↩ upper_plate_↩ of_L3, Center_lower↩ _plate_of_L2]	ISB_JCS
L2_frame	Second_↩ lumbar_↩ vertebra_frame		Second_↩ lumbar_↩ vertebra	[Lower_↩ articular_↩ facet_of_L2_R, Lower_↩ articular_↩ facet_of_L2_L, Center_↩ upper_plate_↩ of_L2, Center_lower↩ _plate_of_L2] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_L2, Left_point_of↩ _the_inferior↩ _plate_of_L2, Center_↩ upper_plate_↩ of_L2, Center_lower↩ _plate_of_L2] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_L2, Left_point_of↩ _the_↩ superior_↩ plate_of_L2, Center_↩ upper_plate_↩ of_L2, Center_lower↩ _plate_of_L2]	ISB_BCS

L3_L2_arch_↩ frame			Second_↩ lumbar_↩ vertebra	[Center_↩ lower_plate_↩ of_L2, Center_↩ upper_plate_↩ of_L3]	ISB_JCS
L3_L4_arch_↩ frame			Fourth_↩ lumbar_↩ vertebra	[Center_↩ upper_plate_↩ of_L4, Center_lower↩ _plate_of_L3]	ISB_JCS
L3_frame	Third_lumbar↩ _vertebra_↩ frame		Third_lumbar↩ _vertebra	[Lower_↩ articular_↩ facet_of_L3_R, Lower_↩ articular_↩ facet_of_L3_L, Center_↩ upper_plate_↩ of_L3, Center_lower↩ _plate_of_L3] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_L3, Left_point_of↩ _the_inferior↩ _plate_of_L3, Center_↩ upper_plate_↩ of_L3, Center_lower↩ _plate_of_L3] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_L3, Left_point_of↩ _the_↩ superior_↩ plate_of_L3, Center_↩ upper_plate_↩ of_L3, Center_lower↩ _plate_of_L3]	ISB_BCS

L4_L3_arch_↩ frame			Third_lumbar↩ _vertebra	[Center_↩ lower_plate_↩ of_L3, Center_↩ upper_plate_↩ of_L4]	ISB_JCS
L4_L5_arch_↩ frame			Fifth_lumbar↩ _vertebra	[Center_↩ upper_plate_↩ of_L5, Center_lower↩ _plate_of_L4]	ISB_JCS
L4_frame	Fourth_↩ lumbar_↩ vertebra_frame		Fourth_↩ lumbar_↩ vertebra	[Lower_↩ articular_↩ facet_of_L4_R, Lower_↩ articular_↩ facet_of_L4_L, Center_↩ upper_plate_↩ of_L4, Center_lower↩ _plate_of_L4] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_L4, Left_point_of↩ _the_inferior↩ _plate_of_L4, Center_↩ upper_plate_↩ of_L4, Center_lower↩ _plate_of_L4] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_L4, Left_point_of↩ _the_↩ superior_↩ plate_of_L4, Center_↩ upper_plate_↩ of_L4, Center_lower↩ _plate_of_L4]	ISB_BCS

L5_L4_arch_↩ frame			Fourth_↩ lumbar_↩ vertebra	[Center_↩ lower_plate_↩ of_L4, Center_↩ upper_plate_↩ of_L5]	ISB_JCS
L5_S1_arch_↩ frame			Sacrum	[Center_↩ Sacral_Plate, Center_lower↩ _plate_of_L5]	ISB_JCS
L5_frame	Fifth_lumbar↩ _vertebra_↩ frame		Fifth_lumbar↩ _vertebra	[Lower_↩ articular_↩ facet_of_L5_R, Lower_↩ articular_↩ facet_of_L5_L, Center_↩ upper_plate_↩ of_L5, Center_lower↩ _plate_of_L5] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_L5, Left_point_of↩ _the_inferior↩ _plate_of_L5, Center_↩ upper_plate_↩ of_L5, Center_lower↩ _plate_of_L5] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_L5, Left_point_of↩ _the_↩ superior_↩ plate_of_L5, Center_↩ upper_plate_↩ of_L5, Center_lower↩ _plate_of_L5]	ISB_BCS

Left_↩ calcaneus_↩ ankle_frame			Left_calcaneus	[Tip_medial_↩ malleolus_of↩ _left_tibia, Tip_lateral_↩ malleolus_of↩ _left_fibula, Medial_point↩ _medial_↩ condyle_of_↩ left_tibia, Lateral_point↩ _lateral_↩ condyle_of_↩ left_tibia]	ISB_JCS
Left_carpal_↩ radiocarpal_↩ frame			Skeleton_of_↩ left_hand	[Lateral_↩ epicondyle_↩ of_left_↩ humerus, Styloid_↩ process_of_↩ left_radius, Styloid_↩ process_of_↩ left_ulna]	ISB_JCS
Left_clavicle_↩ acromioclavicular↩ _frame			Left_clavicle	[Left_clavicle↩ _↩ acromioclavicular↩ _joint, Left_scapula↩ _↩ acromioclavicular↩ _joint]	ISB_JCS
Left_clavicle_↩ sternoclavicular↩ _frame			Left_clavicle	[Left_↩ sternum_↩ sternoclavicular↩ _joint, Left_clavicle_↩ sternoclavicular↩ _joint]	ISB_JCS
Left_femur_↩ hip_frame		femur JCS for left hip	Left_femur	[Medial_↩ epicondyle_↩ of_left_femur, Lateral_↩ epicondyle_↩ of_left_femur, Head_center↩ _of_left_femur]	ISB_JCS

Left_femur_↩ knee_frame			Left_femur	[Medial_↩ epicondyle_↩ of_left_femur, Lateral_↩ epicondyle_↩ of_left_femur, Head_center↩ _of_left_femur]	ISB_JCS
Left_↩ humerus_↩ glenohumeral↩ _frame			Left_humerus	[Medial_↩ epicondyle_↩ of_left_↩ humerus, Lateral_↩ epicondyle_↩ of_left_↩ humerus, Head_center↩ _of_left_↩ humerus]	ISB_JCS
Left_↩ humerus_↩ humeroulnar↩ _frame			Left_humerus	[Medial_↩ epicondyle_↩ of_left_↩ humerus, Lateral_↩ epicondyle_↩ of_left_↩ humerus, Head_center↩ _of_left_↩ humerus]	ISB_JCS
Left_radius_↩ radiocarpal_↩ frame			Left_radius	[Lateral_↩ epicondyle_↩ of_left_↩ humerus, Styloid_↩ process_of_↩ left_radius, Styloid_↩ process_of_↩ left_ulna]	ISB_JCS
Left_scapula↩ _↩ acromioclavicular↩ _frame			Left_scapula	[Left_clavicle↩ _↩ acromioclavicular↩ _joint, Left_scapula↩ _↩ acromioclavicular↩ _joint]	ISB_JCS

Left_scapula↩ _↩ glenohumeral↩ _frame			Left_scapula	[Head_center↩ _of_left_↩ humerus]	ISB_JCS
Left_tibia_↩ ankle_frame			Left_tibia	[Tip_medial_↩ malleolus_of↩ _left_tibia, Tip_lateral_↩ malleolus_of↩ _left_fibula, Medial_point↩ _medial_↩ condyle_of_↩ left_tibia, Lateral_point↩ _lateral_↩ condyle_of_↩ left_tibia]	ISB_JCS
Left_tibia_↩ knee_frame			Left_tibia	[Tip_medial_↩ malleolus_of↩ _left_tibia, Tip_lateral_↩ malleolus_of↩ _left_fibula, Medial_point↩ _medial_↩ condyle_of_↩ left_tibia, Lateral_point↩ _lateral_↩ condyle_of_↩ left_tibia, Medial_↩ epicondyle_↩ of_left_femur, Lateral_↩ epicondyle_↩ of_left_femur]	ISB_JCS

Left_ulna_↩ humeroulnar↩ _frame			Left_ulna	[Medial_↩ epicondyle_↩ of_left_↩ humerus, Lateral_↩ epicondyle_↩ of_left_↩ humerus, Styloid_↩ process_of_↩ left_ulna]	ISB_JCS
Pelvic_frame	Pelvic_↩ skeleton_frame		Pelvic_skeleton	[Right_↩ acetabular_↩ center, Left_↩ acetabular_↩ center, LPSIS, RPSIS, LASIS, RASIS]	ISB_BCS
Pelvis_left_↩ hip_frame		pelvis JCS for left hip	Left_hip_bone	[LPSIS, RPSIS, LASIS, RASIS, Left_↩ acetabular_↩ center]	ISB_JCS
Pelvis_right_↩ hip_frame		pelvis JCS for right hip	Right_hip_bone	[LPSIS, RPSIS, LASIS, RASIS, Right_↩ acetabular_↩ center]	ISB_JCS
Right_↩ calcaneus_↩ ankle_frame			Right_↩ calcaneus	[Tip_medial_↩ malleolus_of↩ _right_tibia, Tip_lateral_↩ malleolus_of↩ _right_fibula, Medial_point↩ _medial_↩ condyle_of_↩ right_tibia, Lateral_point↩ _lateral_↩ condyle_of_↩ right_tibia]	ISB_JCS

Right_carpal↩ _radiocarpal↩ _frame			Skeleton_of_↩ right_hand	[Lateral_↩ epicondyle_↩ of_right_↩ humerus, Styloid_↩ process_of_↩ right_radius, Styloid_↩ process_of_↩ right_ulna]	ISB_JCS
Right_↩ clavicle_↩ acromioclavicular↩ _frame			Right_clavicle	[Right_↩ clavicle_↩ acromioclavicular↩ _joint, Right_↩ scapula_↩ acromioclavicular↩ _joint]	ISB_JCS
Right_↩ clavicle_↩ sternoclavicular↩ _frame			Right_clavicle	[Right_↩ sternum_↩ sternoclavicular↩ _joint, Right_↩ clavicle_↩ sternoclavicular↩ _joint]	ISB_JCS
Right_femur_↩ hip_frame		femur JCS for right hip	Right_femur	[Medial_↩ epicondyle_↩ of_right_femur, Lateral_↩ epicondyle_↩ of_right_femur, Head_center↩ _of_right_↩ femur]	ISB_JCS
Right_femur_↩ knee_frame			Right_femur	[Medial_↩ epicondyle_↩ of_right_femur, Lateral_↩ epicondyle_↩ of_right_femur, Head_center↩ _of_right_↩ femur]	ISB_JCS

Right_↩ humerus_↩ glenohumeral↩ _frame			Right_humerus	[Medial_↩ epicondyle_↩ of_right_↩ humerus, Lateral_↩ epicondyle_↩ of_right_↩ humerus, Head_center↩ _of_right_↩ humerus]	ISB_JCS
Right_↩ humerus_↩ humeroulnar↩ _frame			Right_humerus	[Medial_↩ epicondyle_↩ of_right_↩ humerus, Lateral_↩ epicondyle_↩ of_right_↩ humerus, Head_center↩ _of_right_↩ humerus]	ISB_JCS
Right_radius↩ _radiocarpal↩ _frame			Right_radius	[Lateral_↩ epicondyle_↩ of_right_↩ humerus, Styloid_↩ process_of_↩ right_radius, Styloid_↩ process_of_↩ right_ulna]	ISB_JCS
Right_↩ scapula_↩ acromioclavicular↩ _frame			Right_scapula	[Right_↩ clavicle_↩ acromioclavicular↩ _joint, Right_↩ scapula_↩ acromioclavicular↩ _joint]	ISB_JCS
Right_↩ scapula_↩ glenohumeral↩ _frame			Right_scapula	[Head_center↩ _of_right_↩ humerus]	ISB_JCS
Right_tibia_↩ ankle_frame			Right_tibia	[Tip_medial_↩ malleolus_of↩ _right_tibia, Tip_lateral_↩ malleolus_of↩ _right_fibula, Medial_point↩ _medial_↩ condyle_of_↩ right_tibia, Lateral_point↩ _lateral_↩ condyle_of_↩ right_tibia]	ISB_JCS

Right_tibia_↩ knee_frame			Right_tibia	[Tip_medial_↩ malleolus_of↩ _right_tibia, Tip_lateral_↩ malleolus_of↩ _right_fibula, Medial_point↩ _medial_↩ condyle_of_↩ right_tibia, Lateral_point↩ _lateral_↩ condyle_of_↩ right_tibia, Medial_↩ epicondyle_↩ of_right_femur, Lateral_↩ epicondyle_↩ of_right_femur]	ISB_JCS
Right_ulna_↩ humeroulnar↩ _frame			Right_ulna	[Medial_↩ epicondyle_↩ of_right_↩ humerus, Lateral_↩ epicondyle_↩ of_right_↩ humerus, Styloid_↩ process_of_↩ right_ulna]	ISB_JCS
S1_L5_arch_↩ frame			Fifth_lumbar↩ _vertebra	[Center_↩ lower_plate_↩ of_L5, Center_↩ Sacral_Plate]	ISB_JCS
Skull_C1_↩ frame			Skull	[Center_↩ atlanto_↩ occipital_joint, Center_↩ atlanto_↩ occipital_joint]	ISB_JCS
Sternum_left↩ _↩ sternoclavicular↩ _frame			Sternum	[Left_↩ sternum_↩ sternoclavicular↩ _joint, Left_clavicle_↩ sternoclavicular↩ _joint]	ISB_JCS

Sternum_↩ right_↩ sternoclavicular↩ _frame			Sternum	[Right_↩ sternum_↩ sternoclavicular↩ _joint, Right_↩ clavicle_↩ sternoclavicular↩ _joint]	ISB_JCS
T10_T11_↩ arch_frame			Eleventh_↩ thoracic_↩ vertebra	[Center_↩ upper_plate_↩ of_T11, Center_lower↩ _plate_of_T10]	ISB_JCS
T10_T9_arch↩ _frame			Ninth_↩ thoracic_↩ vertebra	[Center_↩ lower_plate_↩ of_T9, Center_↩ upper_plate_↩ of_T10]	ISB_JCS
T10_frame	Tenth_↩ thoracic_↩ vertebra_frame		Tenth_↩ thoracic_↩ vertebra	[Lower_↩ articular_↩ facet_of_T10↩ _R, Lower_↩ articular_↩ facet_of_T10↩ _L, Center_↩ upper_plate_↩ of_T10, Center_lower↩ _plate_of_T10] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_T10, Left_point_of↩ _the_inferior↩ _plate_of_T10, Center_↩ upper_plate_↩ of_T10, Center_lower↩ _plate_of_T10] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_T10, Left_point_of↩ _the_↩ superior_↩ plate_of_T10, Center_↩ upper_plate_↩ of_T10, Center_lower↩ _plate_of_T10]	ISB_BCS

T11_T10_↩ arch_frame			Tenth_↩ thoracic_↩ vertebra	[Center_↩ lower_plate_↩ of_T10, Center_↩ upper_plate_↩ of_T11]	ISB_JCS
T11_T12_↩ arch_frame			Twelfth_↩ thoracic_↩ vertebra	[Center_↩ upper_plate_↩ of_T12, Center_lower↩ _plate_of_T11]	ISB_JCS
T11_frame	Eleventh_↩ thoracic_↩ vertebra_frame		Eleventh_↩ thoracic_↩ vertebra	[Lower_↩ articular_↩ facet_of_T11↩ _R, Lower_↩ articular_↩ facet_of_T11↩ _L, Center_↩ upper_plate_↩ of_T11, Center_lower↩ _plate_of_T11] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_T11, Left_point_of↩ _the_inferior↩ _plate_of_T11, Center_↩ upper_plate_↩ of_T11, Center_lower↩ _plate_of_T11] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_T11, Left_point_of↩ _the_↩ superior_↩ plate_of_T11, Center_↩ upper_plate_↩ of_T11, Center_lower↩ _plate_of_T11]	ISB_BCS

T12_L1_arch↩ _frame			First_lumbar↩ _vertebra	[Center_↩ upper_plate_↩ of_L1, Center_lower↩ _plate_of_T12]	ISB_JCS
T12_T11_↩ arch_frame			Eleventh_↩ thoracic_↩ vertebra	[Center_↩ lower_plate_↩ of_T11, Center_↩ upper_plate_↩ of_T12]	ISB_JCS
T12_frame	Twelfth_↩ thoracic_↩ vertebra_frame		Twelfth_↩ thoracic_↩ vertebra	[Lower_↩ articular_↩ facet_of_T12↩ _R, Lower_↩ articular_↩ facet_of_T12↩ _L, Center_↩ upper_plate_↩ of_T12, Center_lower↩ _plate_of_T12] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_T12, Left_point_of↩ _the_inferior↩ _plate_of_T12, Center_↩ upper_plate_↩ of_T12, Center_lower↩ _plate_of_T12] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_T12, Left_point_of↩ _the_↩ superior_↩ plate_of_T12, Center_↩ upper_plate_↩ of_T12, Center_lower↩ _plate_of_T12]	ISB_BCS

T1_C7_arch↩ _frame			Seventh_↩ cervical_↩ vertebra	[Center_↩ lower_plate_↩ of_C7, Center_↩ upper_plate_↩ of_T1]	ISB_JCS
T1_T2_arch_↩ frame			Second_↩ thoracic_↩ vertebra	[Center_↩ upper_plate_↩ of_T2, Center_lower↩ _plate_of_T1]	ISB_JCS
T1_frame	First_thoracic↩ _vertebra_↩ frame, T01_frame		First_thoracic↩ _vertebra	[Lower_↩ articular_↩ facet_of_T1_R, Lower_↩ articular_↩ facet_of_T1_L, Center_↩ upper_plate_↩ of_T1, Center_lower↩ _plate_of_T1] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_T1, Left_point_of↩ _the_inferior↩ _plate_of_T1, Center_↩ upper_plate_↩ of_T1, Center_lower↩ _plate_of_T1] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_T1, Left_point_of↩ _the_↩ superior_↩ plate_of_T1, Center_↩ upper_plate_↩ of_T1, Center_lower↩ _plate_of_T1]	ISB_BCS

T2_T1_arch_↩ frame			First_thoracic↩ _vertebra	[Center_↩ lower_plate_↩ of_T1, Center_↩ upper_plate_↩ of_T2]	ISB_JCS
T2_T3_arch_↩ frame			Third_↩ thoracic_↩ vertebra	[Center_↩ upper_plate_↩ of_T3, Center_lower↩ _plate_of_T2]	ISB_JCS
T2_frame	Second_↩ thoracic_↩ vertebra_frame, T02_frame		Second_↩ thoracic_↩ vertebra	[Lower_↩ articular_↩ facet_of_T2_R, Lower_↩ articular_↩ facet_of_T2_L, Center_↩ upper_plate_↩ of_T2, Center_lower↩ _plate_of_T2] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_T2, Left_point_of↩ _the_inferior↩ _plate_of_T2, Center_↩ upper_plate_↩ of_T2, Center_lower↩ _plate_of_T2] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_T2, Left_point_of↩ _the_↩ superior_↩ plate_of_T2, Center_↩ upper_plate_↩ of_T2, Center_lower↩ _plate_of_T2]	ISB_BCS

T3_T2_arch_↩ frame			Second_↩ thoracic_↩ vertebra	[Center_↩ lower_plate_↩ of_T2, Center_↩ upper_plate_↩ of_T3]	ISB_JCS
T3_T4_arch_↩ frame			Fourth_↩ thoracic_↩ vertebra	[Center_↩ upper_plate_↩ of_T4, Center_lower↩ _plate_of_T3]	ISB_JCS
T3_frame	Third_↩ thoracic_↩ vertebra_frame, T03_frame		Third_↩ thoracic_↩ vertebra	[Lower_↩ articular_↩ facet_of_T3_R, Lower_↩ articular_↩ facet_of_T3_L, Center_↩ upper_plate_↩ of_T3, Center_lower↩ _plate_of_T3] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_T3, Left_point_of↩ _the_inferior↩ _plate_of_T3, Center_↩ upper_plate_↩ of_T3, Center_lower↩ _plate_of_T3] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_T3, Left_point_of↩ _the_↩ superior_↩ plate_of_T3, Center_↩ upper_plate_↩ of_T3, Center_lower↩ _plate_of_T3]	ISB_BCS

T4_T3_arch_↩ frame			Third_↩ thoracic_↩ vertebra	[Center_↩ lower_plate_↩ of_T3, Center_↩ upper_plate_↩ of_T4]	ISB_JCS
T4_T5_arch_↩ frame			Fifth_thoracic↩ _vertebra	[Center_↩ upper_plate_↩ of_T5, Center_lower↩ _plate_of_T4]	ISB_JCS
T4_frame	Fourth_↩ thoracic_↩ vertebra_frame, T04_frame		Fourth_↩ thoracic_↩ vertebra	[Lower_↩ articular_↩ facet_of_T4_R, Lower_↩ articular_↩ facet_of_T4_L, Center_↩ upper_plate_↩ of_T4, Center_lower↩ _plate_of_T4] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_T4, Left_point_of↩ _the_inferior↩ _plate_of_T4, Center_↩ upper_plate_↩ of_T4, Center_lower↩ _plate_of_T4] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_T4, Left_point_of↩ _the_↩ superior_↩ plate_of_T4, Center_↩ upper_plate_↩ of_T4, Center_lower↩ _plate_of_T4]	ISB_BCS

T5_T4_arch_↩ frame			Fourth_↩ thoracic_↩ vertebra	[Center_↩ lower_plate_↩ of_T4, Center_↩ upper_plate_↩ of_T5]	ISB_JCS
T5_T6_arch_↩ frame			Sixth_↩ thoracic_↩ vertebra	[Center_↩ upper_plate_↩ of_T6, Center_lower↩ _plate_of_T5]	ISB_JCS
T5_frame	Fifth_thoracic↩ _vertebra_↩ frame, T05_frame		Fifth_thoracic↩ _vertebra	[Lower_↩ articular_↩ facet_of_T5_R, Lower_↩ articular_↩ facet_of_T5_L, Center_↩ upper_plate_↩ of_T5, Center_lower↩ _plate_of_T5] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_T5, Left_point_of↩ _the_inferior↩ _plate_of_T5, Center_↩ upper_plate_↩ of_T5, Center_lower↩ _plate_of_T5] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_T5, Left_point_of↩ _the_↩ superior_↩ plate_of_T5, Center_↩ upper_plate_↩ of_T5, Center_lower↩ _plate_of_T5]	ISB_BCS

T6_T5_arch_↩ frame			Fifth_thoracic↩ _vertebra	[Center_↩ lower_plate_↩ of_T5, Center_↩ upper_plate_↩ of_T6]	ISB_JCS
T6_T7_arch_↩ frame			Seventh_↩ thoracic_↩ vertebra	[Center_↩ upper_plate_↩ of_T7, Center_lower↩ _plate_of_T6]	ISB_JCS
T6_frame	Sixth_↩ thoracic_↩ vertebra_frame, T06_frame		Sixth_↩ thoracic_↩ vertebra	[Lower_↩ articular_↩ facet_of_T6_R, Lower_↩ articular_↩ facet_of_T6_L, Center_↩ upper_plate_↩ of_T6, Center_lower↩ _plate_of_T6] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_T6, Left_point_of↩ _the_inferior↩ _plate_of_T6, Center_↩ upper_plate_↩ of_T6, Center_lower↩ _plate_of_T6] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_T6, Left_point_of↩ _the_↩ superior_↩ plate_of_T6, Center_↩ upper_plate_↩ of_T6, Center_lower↩ _plate_of_T6]	ISB_BCS

T7_T6_arch_↩ frame			Sixth_↩ thoracic_↩ vertebra	[Center_↩ lower_plate_↩ of_T6, Center_↩ upper_plate_↩ of_T7]	ISB_JCS
T7_T8_arch_↩ frame			Eighth_↩ thoracic_↩ vertebra	[Center_↩ upper_plate_↩ of_T8, Center_lower↩ _plate_of_T7]	ISB_JCS
T7_frame	Seventh_↩ thoracic_↩ vertebra_frame, T07_frame		Seventh_↩ thoracic_↩ vertebra	[Lower_↩ articular_↩ facet_of_T7_R, Lower_↩ articular_↩ facet_of_T7_L, Center_↩ upper_plate_↩ of_T7, Center_lower↩ _plate_of_T7] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_T7, Left_point_of↩ _the_inferior↩ _plate_of_T7, Center_↩ upper_plate_↩ of_T7, Center_lower↩ _plate_of_T7] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_T7, Left_point_of↩ _the_↩ superior_↩ plate_of_T7, Center_↩ upper_plate_↩ of_T7, Center_lower↩ _plate_of_T7]	ISB_BCS

T8_T7_arch_↩ frame			Seventh_↩ thoracic_↩ vertebra	[Center_↩ lower_plate_↩ of_T7, Center_↩ upper_plate_↩ of_T8]	ISB_JCS
T8_T9_arch_↩ frame			Ninth_↩ thoracic_↩ vertebra	[Center_↩ upper_plate_↩ of_T9, Center_lower↩ _plate_of_T8]	ISB_JCS
T8_frame	Eighth_↩ thoracic_↩ vertebra_frame, T08_frame		Eighth_↩ thoracic_↩ vertebra	[Lower_↩ articular_↩ facet_of_T8_R, Lower_↩ articular_↩ facet_of_T8_L, Center_↩ upper_plate_↩ of_T8, Center_lower↩ _plate_of_T8] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_T8, Left_point_of↩ _the_inferior↩ _plate_of_T8, Center_↩ upper_plate_↩ of_T8, Center_lower↩ _plate_of_T8] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_T8, Left_point_of↩ _the_↩ superior_↩ plate_of_T8, Center_↩ upper_plate_↩ of_T8, Center_lower↩ _plate_of_T8]	ISB_BCS

T9_T10_arch↩ _frame			Tenth_↩ thoracic_↩ vertebra	[Center_↩ upper_plate_↩ of_T10, Center_lower↩ _plate_of_T9]	ISB_JCS
T9_T8_arch_↩ frame			Eighth_↩ thoracic_↩ vertebra	[Center_↩ lower_plate_↩ of_T8, Center_↩ upper_plate_↩ of_T9]	ISB_JCS
T9_frame	Ninth_↩ thoracic_↩ vertebra_frame, T09_frame		Ninth_↩ thoracic_↩ vertebra	[Lower_↩ articular_↩ facet_of_T9_R, Lower_↩ articular_↩ facet_of_T9_L, Center_↩ upper_plate_↩ of_T9, Center_lower↩ _plate_of_T9] **or** [Right_↩ point_of_the↩ _inferior_↩ plate_of_T9, Left_point_of↩ _the_inferior↩ _plate_of_T9, Center_↩ upper_plate_↩ of_T9, Center_lower↩ _plate_of_T9] **or** [Right_↩ point_of_the↩ _superior_↩ plate_of_T9, Left_point_of↩ _the_↩ superior_↩ plate_of_T9, Center_↩ upper_plate_↩ of_T9, Center_lower↩ _plate_of_T9]	ISB_BCS

18 Appendix: List of joints

Joint name	Description	Frames	Bibliography
Atlanto-axial_joint		C1_C2_arch_frame, C2_C1_arch_frame	FMA

Atlanto-occipital_joint		C1_Skull_frame, Skull_C1_frame	FMA
C2-C3_vertebral_arch↩ _joint		C2_C3_arch_frame, C3_C2_arch_frame	FMA
C3-C4_vertebral_arch↩ _joint		C3_C4_arch_frame, C4_C3_arch_frame	FMA
C4-C5_vertebral_arch↩ _joint		C4_C5_arch_frame, C5_C4_arch_frame	FMA
C5-C6_vertebral_arch↩ _joint		C5_C6_arch_frame, C6_C5_arch_frame	FMA
C6-C7_vertebral_arch↩ _joint		C6_C7_arch_frame, C7_C6_arch_frame	FMA
C7-T1_vertebral_arch↩ _joint		C7_T1_arch_frame, T1_C7_arch_frame	FMA
L1-L2_vertebral_arch_↩ joint		L1_L2_arch_frame, L2_L1_arch_frame	FMA
L2-L3_vertebral_arch_↩ joint		L2_L3_arch_frame, L3_L2_arch_frame	FMA
L3-L4_vertebral_arch_↩ joint		L3_L4_arch_frame, L4_L3_arch_frame	FMA
L4-L5_vertebral_arch_↩ joint		L4_L5_arch_frame, L5_L4_arch_frame	FMA
L5-S1_vertebral_arch_↩ joint		L5_S1_arch_frame, S1_L5_arch_frame	FMA
Left_acromioclavicular↩ _joint		Left_scapula_↩ acromioclavicular_frame, Left_clavicle_↩ acromioclavicular_frame	FMA
Left_ankle_joint		Left_tibia_ankle_frame, Left_calcaneus_ankle_↩ frame	FMA
Left_elbow_joint		Left_humerus_↩ humeroulnar_frame, Left_ulna_↩ humeroulnar_frame	FMA
Left_glenohumeral_joint		Left_scapula_↩ glenohumeral_frame, Left_humerus_↩ glenohumeral_frame	FMA
Left_hip_joint		Pelvis_left_hip_frame, Left_femur_hip_frame	FMA
Left_knee_joint		Left_femur_knee_frame, Left_tibia_knee_frame	FMA
Left_sternoclavicular_↩ joint		Sternum_left_↩ sternoclavicular_frame, Left_clavicle_↩ sternoclavicular_frame	FMA
Left_wrist_joint		Left_radius_↩ radiocarpal_frame, Left_carpal_↩ radiocarpal_frame	FMA

Right_↩ acromioclavicular_joint		Right_scapula_↩ acromioclavicular_frame, Right_clavicle_↩ acromioclavicular_frame	FMA
Right_ankle_joint		Right_tibia_ankle_frame, Right_calcaneus_↩ ankle_frame	FMA
Right_elbow_joint		Right_humerus_↩ humeroulnar_frame, Right_ulna_↩ humeroulnar_frame	FMA
Right_glenohumeral_↩ joint		Right_scapula_↩ glenohumeral_frame, Right_humerus_↩ glenohumeral_frame	FMA
Right_hip_joint		Pelvis_right_hip_frame, Right_femur_hip_frame	FMA
Right_knee_joint		Right_femur_knee_↩ frame, Right_tibia_knee_frame	FMA
Right_sternoclavicular↩ _joint		Sternum_right_↩ sternoclavicular_frame, Right_clavicle_↩ sternoclavicular_frame	FMA
Right_wrist_joint		Right_radius_↩ radiocarpal_frame, Right_carpal_↩ radiocarpal_frame	FMA
T1-T2_vertebral_arch_↩ joint		T1_T2_arch_frame, T2_T1_arch_frame	FMA
T10-T11_vertebral_↩ arch_joint		T10_T11_arch_frame, T11_T10_arch_frame	FMA
T11-T12_vertebral_↩ arch_joint		T11_T12_arch_frame, T12_T11_arch_frame	FMA
T12-L1_vertebral_arch↩ _joint		T12_L1_arch_frame, L1_T12_arch_frame	FMA
T2-T3_vertebral_arch_↩ joint		T2_T3_arch_frame, T3_T2_arch_frame	FMA
T3-T4_vertebral_arch_↩ joint		T3_T4_arch_frame, T4_T3_arch_frame	FMA
T4-T5_vertebral_arch_↩ joint		T4_T5_arch_frame, T5_T4_arch_frame	FMA
T5-T6_vertebral_arch_↩ joint		T5_T6_arch_frame, T6_T5_arch_frame	FMA
T6-T7_vertebral_arch_↩ joint		T6_T7_arch_frame, T7_T6_arch_frame	FMA
T7-T8_vertebral_arch_↩ joint		T7_T8_arch_frame, T8_T7_arch_frame	FMA
T8-T9_vertebral_arch_↩ joint		T8_T9_arch_frame, T9_T8_arch_frame	FMA
T9-T10_vertebral_arch↩ _joint		T9_T10_arch_frame, T10_T9_arch_frame	FMA

19 Appendix: Example of model description files

19.1 Example of file for LS Dyna format (fixed size)

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE model_description SYSTEM "ModelRules.dtd">
<model_description>
    <units length="mm" mass="kg" age="year" />
    <source>model_01.dyn</source>
    <format_rules format="LSDyna_fix">../formatrules/Formatrules_LSDyna_Fixed.pfr</format_rules>
    <anthropometry name="age">
        <units age="year"/>
        <value>28</value>
    </anthropometry>
    <anthropometry name="height">
        <units length="mm"/>
        <value>1749</value>
    </anthropometry>
    <anthropometry name="weight">
        <units mass="kg"/>
        <value>77</value>
    </anthropometry>

    <!-- GENERICMETADATA -->
    <genericmetadata name="Gen_1">
            <keyword kw="*SET_NODE_LIST_TITLE">
                <id>10002</id>
            </keyword>
            <keyword kw="*SET_PART_LIST_TITLE">
                <id>1</id>
            </keyword>
    </genericmetadata>
    <genericmetadata name="auto">
            <keyword kw="*SET_BEAM_TITLE">
                <id>10</id>
            </keyword>
            <keyword kw="*SET_NODE_LIST_TITLE">
                <id>10002</id>
            </keyword>
    </genericmetadata>
    <genericmetadata name="Gen_2_node">
            <keyword kw="*NODE">
                <id>7048431 7048432</id>
            </keyword>
    </genericmetadata>
    <!-- ENTITY -->
    <entity name="Entity_1">
        <keyword kw="*SET_BEAM_TITLE">
            <id>10</id>
        </keyword>
    </entity>
    <entity name="Entity_2">
        <keyword kw="*SET_SHELL_LIST_TITLE">
            <id>11 111</id>
        </keyword>
    </entity>
    <entity name="auto">
        <keyword kw="*SET_PART_LIST_TITLE">
            <id>1</id>
        </keyword>
    </entity>
    <entity name="Auto">
        <keyword kw="*SET_PART_LIST_TITLE">
            <id>1999999</id>
        </keyword>
    </entity>
    <!-- LANDMARK -->
    <landmarks name="pointcoord" type="point">
        <coord>12 13 14</coord>
    </landmarks>
    <landmarks name="barycoord" type="barycenter">
        <coord>10 20 30
               100 200 300</coord>
    </landmarks>
    <landmarks name="Auto" type="point">
        <keyword kw="*SET_NODE_LIST_TITLE">
            <id>1999991</id>
        </keyword>
    </landmarks>
    <landmarks name="auto" type="point">
        <keyword kw="*SET_NODE_LIST_TITLE">
            <id>10001</id>
        </keyword>
    </landmarks>
    <landmarks name="Landmark_2" type="sphere">
        <keyword kw="*SET_NODE_LIST_TITLE">
            <id>10002</id>
        </keyword>
    </landmarks>
```

```
86      <landmarks name="Landmark_Node" type="point">
87          <keyword kw="*NODE">
88              <id>7020597</id>
89          </keyword>
90      </landmarks>
91      <!-- CONTROL POINT -->
92      <controlPoint name="CP_0" role="source">
93          <keyword kw="*NODE">
94              <id>7020597</id>
95          </keyword>
96      </controlPoint>
97      <controlPoint name="CP_1" role="source">
98          <keyword kw="*SET_NODE_LIST_TITLE">
99              <id>1</id>
100          </keyword>
101      </controlPoint>
102      <controlPoint name="CP_2" role="source">
103          <coord>
104        23        -43.3839         20.0958         225.346
105        24         -65.291               0         229.255
106        25        -21.7513               0         222.236
107        26        -43.3839        -20.0958         225.346
108        27        -26.8911               0         193.384
109          </coord>
110          <weight>-20 -20 -20 -45 -45</weight>
111          <as_bones>1 1 1 0 0</as_bones>
112          <as_skin> 0 0 0 1 1</as_skin>
113      </controlPoint>
114      <controlPoint name="CP_3" role="source">
115          <coordfix>
116         -10 20 30
117         -20 30 40
118          </coordfix>
119      </controlPoint>
120      <!-- JOINT -->
121      <joint name="Joint_1">
122          <entity_master name="Entity_1">
123              <setFrame>
124                  <keyword kw="*DEFINE_COORDINATE_SYSTEM_TITLE">
125                      <id>1992102</id>
126                  </keyword>
127              </setFrame>
128          </entity_master>
129          <entity_slave name="Entity_2">
130              <setFrame>
131                  <keyword kw="*DEFINE_COORDINATE_SYSTEM_TITLE">
132                      <id>1992102</id>
133                  </keyword>
134              </setFrame>
135          </entity_slave>
136          <setDof>0 0 0 1 1 1</setDof>
137      </joint>
138      <joint name="Joint_2">
139          <entity_master name="Entity_1">
140              <setFrame>
141                  <keyword kw="*DEFINE_COORDINATE_NODES">
142                      <id>1992101</id>
143                  </keyword>
144              </setFrame>
145          </entity_master>
146          <entity_slave name="Entity_5_Auto">
147              <setFrame>
148                  <keyword kw="*DEFINE_COORDINATE_NODES">
149                      <id>1992101</id>
150                  </keyword>
151              </setFrame>
152          </entity_slave>
153          <setDof>0 0 0 0 1 0</setDof>
154          <setConstrainedDofType value="hard"></setConstrainedDofType>
155      </joint>
156      <joint name="Joint_3_generalized_spring">
157          <entity_master name="Entity_1">
158              <setFrame type="global">
159                  <keyword kw="*NODE">
160                      <id>7020602</id>
161                  </keyword>
162              </setFrame>
163          </entity_master>
164          <entity_slave name="Entity_5_Auto">
165              <setFrame type="global">
166                  <keyword kw="*NODE">
167                      <id>7020602</id>
168                  </keyword>
169              </setFrame>
170          </entity_slave>
171          <setDof>0 0 0 1 0 1</setDof>
172          <setConstrainedDofType value="soft"></setConstrainedDofType>
```

```
173     </joint>
174     <hbm_parameter name="Density_1" source="Density">
175         <keyword kw="*MAT_STRAIN_RATE_DEPENDENT_PLASTICITY_TITLE">
176             <id>7000014</id>
177         </keyword>
178     </hbm_parameter>
179     <hbm_parameter name="Density_2" source="Density">
180         <keyword kw="*MAT_PLASTIC_KINEMATIC_TITLE">
181             <id>7000015 7000011</id>
182         </keyword>
183     </hbm_parameter>
184     <hbm_parameter name="test_MATID" source="materialID">
185          <keyword kw="*PART">
186              <id>7000033</id>
187          </keyword>
188     </hbm_parameter>
189     <hbm_parameter name="YoungModulus_1" source="YoungModulus">
190         <keyword kw="*MAT_STRAIN_RATE_DEPENDENT_PLASTICITY_TITLE">
191             <id>7000014</id>
192         </keyword>
193         <keyword kw="*MAT_PLASTIC_KINEMATIC_TITLE">
194             <id>7000015</id>
195         </keyword>
196     </hbm_parameter>
197     <hbm_parameter name="Thickness_1" source="Thickness">
198         <keyword kw="*ELEMENT_SHELL_THICKNESS" />
199         <keyword kw="*ELEMENT_SHELL_BETA" />
200     </hbm_parameter>
201
202 <!--    <parameter name="Density">
203         <keyword kw="*MAT_STRAIN_RATE_DEPENDENT_PLASTICITY_TITLE">
204             <id>7000014</id>
205         </keyword>
206         <keyword kw="*MAT_PLASTIC_KINEMATIC_TITLE">
207             <id>7000015</id>
208         </keyword>
209     </parameter>
210     <parameter name="YoungModulus">
211         <keyword kw="*MAT_PLASTIC_KINEMATIC_TITLE">
212             <id>7000015</id>
213         </keyword>
214     </parameter>
215     <parameter name="Thickness" /> -->
216
217     <contact name="contact_1" type="sliding">
218         <thickness keep="true" />
219         <entity_contact_1 name="Entity_1">
220             <setGroup>
221                 <keyword kw="*SET_NODE_LIST_TITLE">
222                     <id>10001</id>
223                 </keyword>
224             </setGroup>
225         </entity_contact_1>
226         <entity_contact_2 name="Entity_2">
227             <setGroup>
228                 <keyword kw="*SET_NODE_LIST_TITLE">
229                     <id>1</id>
230                 </keyword>
231             </setGroup>
232         </entity_contact_2>
233     </contact>
234     <contact name="contact_2" type="sliding">
235         <thickness keep="false" >0.5</thickness>
236         <entity_contact_1 name="Entity_1">
237             <setGroup>
238                 <keyword kw="*SET_NODE_LIST_TITLE">
239                     <id>10001</id>
240                 </keyword>
241             </setGroup>
242         </entity_contact_1>
243         <entity_contact_2 name="Entity_4_Auto">
244             <setGroup>
245                 <keyword kw="*SET_NODE_LIST_TITLE">
246                     <id>10002</id>
247                 </keyword>
248             </setGroup>
249         </entity_contact_2>
250     </contact>
251 </model_description>
```

19.2 Example of file for PamCrash format

```
1 <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2 <!DOCTYPE model_description SYSTEM "ModelRules.dtd">
```

```
<model_description>
    <units length="mm" mass="kg" age="year" />
    <source>model_01.pc</source>
    <format_rules format="LSDyna">../formatrules/Formatrules_PamCrash.pfr</format_rules>
    <anthropometry name="age">
        <units age="year"/>
        <value>28</value>
    </anthropometry>
    <anthropometry name="height">
        <units length="mm"/>
        <value>1749</value>
    </anthropometry>
    <anthropometry name="weight">
        <units mass="kg"/>
        <value>77</value>
    </anthropometry>


    <entity name="Auto">
        <keyword kw="GROUP">
            <name>'Entity_4_Auto' 'Entity_5_Auto'</name>
        </keyword>
    </entity>


    <hbm_parameter name="Density_1" source="Density">
        <keyword kw="MATER">
            <id>101</id>
        </keyword>
    </hbm_parameter>
    <hbm_parameter name="ModuleG_1" source="ModuleG">
        <keyword kw="MATER">
            <id>101</id>
        </keyword>
    </hbm_parameter>
    <hbm_parameter name="Thickness_1" source="ModuleG">
        <keyword kw="PART">
            <id>7000000</id>
        </keyword>
    </hbm_parameter>
    <landmarks name="Landmark_Node" type="point">
        <keyword kw="NODE">
            <id>7020597</id>
        </keyword>
    </landmarks>


</model_description>
```

19.3 Example of file for geometric model (obj format)

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE model_description SYSTEM "ModelRules.dtd">
<model_description>
    <units length="dm" mass="kg" age="year" />
    <source>.</source>
    <format_rules format="obj"/>
    <!-- ENTITY -->
    <entity name="Entity_1">
        <keyword kw="obj">
            <name>RefGeom_CCTs635_BP091_Humerus_R</name>
         </keyword>
    </entity>
    <entity name="Entity_2">
        <keyword kw="obj">
            <name>RefGeom_CCTs635_BP095_Radius_R RefGeom_CCTs635_BP093_Ulna_R</name>
        </keyword>
    </entity>
    <!-- LANDMARK -->
    <landmarks name="land_1" type="point">
        <keyword kw="RefGeom_CCTs635_BP093_Ulna_R">
            <id>154</id>
        </keyword>
    </landmarks>
    <landmarks name="land_2" type="barycenter">
        <keyword kw="RefGeom_CCTs635_BP091_Humerus_R">
            <id>201 203 202 204</id>
        </keyword>
    </landmarks>
</model_description>
```

20 Appendix: Example of format rules files

20.1 Example of file for LS Dyna format (fixed size)

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE format_description SYSTEM "FormatRules.dtd">
<format_description>
    <formatInformation format="LSDyna_fix">
        <comment>$</comment>
        <includeFile>*INCLUDE *INCLUDE_TRANSFORM</includeFile>
    </formatInformation>


     <meshComponent>
         <componentNode>*NODE</componentNode>
         <componentElement1D>*ELEMENT_BEAM *ELEMENT_DISCRETE</componentElement1D>
         <componentElement2D>*ELEMENT_SHELL *ELEMENT_SHELL_THICKNESS
      *ELEMENT_SHELL_BETA</componentElement2D>
         <componentElement3D>*ELEMENT_SOLID</componentElement3D>
         <componentGNode>*SET_NODE_LIST_TITLE *SET_NODE_LIST *PART</componentGNode>
         <componentGElement1D>*SET_BEAM_TITLE *SET_BEAM *PART</componentGElement1D>
         <componentGElement2D>*SET_SHELL_LIST_TITLE *PART *SET_SHELL_LIST</componentGElement2D>
         <componentGElement3D>*SET_SOLID_TITLE *SET_SOLID *PART</componentGElement3D>
         <componentGGroup>*SET_PART_LIST_TITLE *SET_PART_LIST *PART</componentGGroup>
         <componentFrame>*DEFINE_COORDINATE_NODES *DEFINE_COORDINATE_SYSTEM_TITLE
      *DEFINE_COORDINATE_NODES_TITLE</componentFrame>
         <componentModelParameter>*PART *MAT_MOONEY-RIVLIN_RUBBER_TITLE *MAT_SIMPLIFIED_RUBBER/FOAM_TITLE
      *MAT_GENERAL_VISCOELASTIC_TITLE *DEFINE_CURVE *DEFINE_CURVE_TITLE *MAT_PIECEWISE_LINEAR_PLASTICITY_TITLE
      *MAT_VISCOELASTIC_TITLE *MAT_FABRIC_TITLE *MAT_ELASTIC_FLUID_TITLE *MAT_RIGID_TITLE *MAT_ELASTIC_TITLE
      *MAT_PLASTIC_KINEMATIC_TITLE *MAT_STRAIN_RATE_DEPENDENT_PLASTICITY_TITLE *ELEMENT_SHELL_THICKNESS *ELEMENT_SHELL_BETA
      *SECTION_SHELL_TITLE</componentModelParameter>
         <elementOrdering>
             <penta>1 2 5 4 3 6</penta>
         </elementOrdering>
     </meshComponent>


     <rule keyword="*NODE">
         <nextLine/>
         <doWhile>
             <condition>
                 <notfind value="*"><curLine/></notfind>
             </condition>
             <body>
                 <parseRule>
                     <parse format="UI" length="8" offset="0" repeat="1" separator="none"><variableAssign
      name="NodeId"/></parse>
                     <parse format="F" length="16" repeat="3" offset="8" separator="none"><variableAssign
      name="coord"/></parse>
                 </parseRule>
                 <nextLine/>
             </body>
         </doWhile>
         <objectNode>
             <setId><variable name="NodeId"/></setId>
             <setCoord><variable name="coord"/></setCoord>
         </objectNode>
     </rule>

     <rule keyword="*PART">
         <nextLine/>
         <parseRule>
             <parse format="C" length="80" offset="0" repeat="1" separator="none"
      default="part"><variableAssign name="partname"/></parse>
         </parseRule>
         <nextLine/>
         <parseRule>
             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
      name="PartId"/></parse>
             <parse format="UI" length="10" offset="20" repeat="1" separator="none"><variableAssign
      name="MatId"/></parse>
         </parseRule>
         <nextLine/>
         <objectGroup type="Id">
             <setId>
                 <sourceId>*ELEMENT_BEAM *ELEMENT_DISCRETE *ELEMENT_SHELL *ELEMENT_SHELL_THICKNESS
      *ELEMENT_SOLID</sourceId>
                 <variable name="PartId"/>
             </setId>
             <setName><variable name="partname"/></setName>
             <setPart>true</setPart>
         </objectGroup>
         <objectParameter>
             <setId>
                 <variable name="PartId"/>
```

```
70              </setId>
71              <setValue>
72                  <type>materialID</type>
73                  <value><variable name="MatId"/></value>
74              </setValue>
75          </objectParameter>
76      </rule>
77
78      <rule keyword="*INCLUDE_TRANSFORM">
79          <nextLine/>
80          <parseRule>
81              <parse format="C" length="80" offset="0" repeat="1" separator="none"><variableAssign
        name="HBMfile"/></parse>
82          </parseRule>
83          <nextLine/>
84          <objectModelFile>
85              <setFile><variable name="HBMfile"/></setFile>
86          </objectModelFile>
87      </rule>
88
89      <rule keyword="*INCLUDE">
90          <nextLine/>
91          <parseRule>
92              <parse format="C" length="80" offset="0" repeat="1" separator="none"><variableAssign
        name="HBMfile"/></parse>
93          </parseRule>
94          <nextLine/>
95          <objectModelFile>
96              <setFile><variable name="HBMfile"/></setFile>
97          </objectModelFile>
98      </rule>
99
100
101      <rule keyword="*ELEMENT_SOLID">
102          <nextLine/>
103          <doWhile>
104              <condition>
105                  <notfind value="*">
106                      <curLine/>
107                  </notfind>
108              </condition>
109              <body>
110                  <parseRule>
111                      <parse format="UI" length="8" offset="0" repeat="1" separator="none"><variableAssign
        name="ElemId"/></parse>
112                      <parse format="UI" length="8" offset="8" repeat="1" separator="none"><variableAssign
        name="GroupId"/></parse>
113                      <parse format="UI" length="8" offset="16" repeat="8" separator="none"><variableAssign
        name="ElemDef"/></parse>
114                  </parseRule>
115                  <nextLine/>
116              </body>
117          </doWhile>
118          <objectElement3D>
119              <setId><variable name="ElemId"/></setId>
120              <setElemDef><variable name="ElemDef"/></setElemDef>
121          </objectElement3D>
122          <objectGroup type="Element3D">
123              <setId><variable name="GroupId"/></setId>
124              <!-- <setPart>true</setPart> -->
125              <addInGroup><variable name="ElemId"/></addInGroup>
126          </objectGroup>
127      </rule>
128
129      <rule keyword="*ELEMENT_SHELL">
130          <nextLine/>
131          <doWhile>
132              <condition>
133                  <notfind value="*">
134                      <curLine/>
135                  </notfind>
136              </condition>
137              <body>
138                  <parseRule>
139                      <parse format="UI" length="8" offset="0" repeat="1" separator="none"><variableAssign
        name="ElemId"/></parse>
140                      <parse format="UI" length="8" offset="8" repeat="1" separator="none"><variableAssign
        name="GroupId"/></parse>
141                      <parse format="UI" length="8" offset="16" repeat="4" separator="none"><variableAssign
        name="ElemDef"/></parse>
142                  </parseRule>
143                  <nextLine/>
144              </body>
145          </doWhile>
146          <objectElement2D>
147              <setId><variable name="ElemId"/></setId>
148              <setElemDef><variable name="ElemDef"/></setElemDef>
```

```
149         </objectElement2D>
150         <objectGroup type="Element2D">
151             <setId><variable name="GroupId"/></setId>
152             <!-- <setPart>true</setPart> -->
153             <addInGroup><variable name="ElemId"/></addInGroup>
154         </objectGroup>
155     </rule>
156 
157     <rule keyword="*ELEMENT_SHELL_THICKNESS">
158         <nextLine/>
159         <doWhile>
160             <condition>
161                 <notfind value="*">
162                     <curLine/>
163                 </notfind>
164             </condition>
165             <body>
166                 <parseRule>
167                     <parse format="UI" length="8" offset="0" repeat="1" separator="none"><variableAssign
      name="ElemId"/></parse>
168                     <parse format="UI" length="8" offset="8" repeat="1" separator="none"><variableAssign
      name="GroupId"/></parse>
169                     <parse format="UI" length="8" offset="16" repeat="4" separator="none"><variableAssign
      name="ElemDef"/></parse>
170                 </parseRule>
171                 <nextLine/>
172                 <parseRule>
173                     <parse format="F" length="16" offset="0" repeat="4" separator="none"><variableAssign
      name="Thick"/></parse>
174                 </parseRule>
175                 <nextLine/>
176             </body>
177         </doWhile>
178         <objectElement2D>
179             <setId><variable name="ElemId"/></setId>
180             <setElemDef><variable name="ElemDef"/></setElemDef>
181         </objectElement2D>
182         <objectGroup type="Element2D">
183             <setId><variable name="GroupId"/></setId>
184             <!-- <setPart>true</setPart> -->
185             <addInGroup><variable name="ElemId"/></addInGroup>
186         </objectGroup>
187         <objectParameter>
188             <setId><variable name="ElemId"/></setId>
189             <setValue>
190                 <type>Thickness</type>
191                 <value><variable name="Thick"/></value>
192             </setValue>
193         </objectParameter>
194     </rule>
195 
196     <rule keyword="*ELEMENT_SHELL_BETA">
197         <nextLine/>
198         <doWhile>
199             <condition>
200                 <notfind value="*">
201                     <curLine/>
202                 </notfind>
203             </condition>
204             <body>
205                 <parseRule>
206                     <parse format="UI" length="8" offset="0" repeat="1" separator="none"><variableAssign
      name="ElemId"/></parse>
207                     <parse format="UI" length="8" offset="8" repeat="1" separator="none"><variableAssign
      name="GroupId"/></parse>
208                     <parse format="UI" length="8" offset="16" repeat="4" separator="none"><variableAssign
      name="ElemDef"/></parse>
209                 </parseRule>
210                 <nextLine/>
211                 <parseRule>
212                     <parse format="F" length="16" offset="0" repeat="4" separator="none"><variableAssign
      name="Thick"/></parse>
213                 </parseRule>
214                 <nextLine/>
215             </body>
216         </doWhile>
217         <objectElement2D>
218             <setId><variable name="ElemId"/></setId>
219             <setElemDef><variable name="ElemDef"/></setElemDef>
220         </objectElement2D>
221         <objectGroup type="Element2D">
222             <setId><variable name="GroupId"/></setId>
223             <!-- <setPart>true</setPart> -->
224             <addInGroup><variable name="ElemId"/></addInGroup>
225         </objectGroup>
226         <objectParameter>
227             <setId><variable name="ElemId"/></setId>
```

```
228             <setValue>
229                 <type>Thickness</type>
230                 <value><variable name="Thick"/></value>
231             </setValue>
232         </objectParameter>
233     </rule>
234 
235     <rule keyword="*ELEMENT_BEAM">
236         <nextLine/>
237         <doWhile>
238             <condition>
239                 <notfind value="*">
240                     <curLine/>
241                 </notfind>
242             </condition>
243             <body>
244                 <parseRule>
245                     <parse format="UI" length="8" offset="0" repeat="1" separator="none"><variableAssign
    name="ElemId"/></parse>
246                     <parse format="UI" length="8" offset="8" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
247                     <parse format="UIpos" length="8" offset="16" repeat="2"
    separator="none"><variableAssign name="ElemDef"/></parse>
248                 </parseRule>
249                 <nextLine/>
250 
251             </body>
252         </doWhile>
253         <objectElement1D>
254             <setId><variable name="ElemId"/></setId>
255             <setElemDef><variable name="ElemDef"/></setElemDef>
256         </objectElement1D>
257         <objectGroup type="Element1D">
258             <setId><variable name="GroupId"/></setId>
259             <!-- <setPart>true</setPart> -->
260             <addInGroup><variable name="ElemId"/></addInGroup>
261         </objectGroup>
262     </rule>
263 
264     <rule keyword="*ELEMENT_DISCRETE">
265         <nextLine/>
266         <doWhile>
267             <condition>
268                 <notfind value="*">
269                     <curLine/>
270                 </notfind>
271             </condition>
272             <body>
273                 <parseRule>
274                     <parse format="UI" length="8" offset="0" repeat="1" separator="none"><variableAssign
    name="ElemId"/></parse>
275                     <parse format="UI" length="8" offset="8" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
276                     <parse format="UIpos" length="8" offset="16" repeat="2"
    separator="none"><variableAssign name="ElemDef"/></parse>
277                 </parseRule>
278                 <nextLine/>
279             </body>
280         </doWhile>
281         <objectElement1D>
282             <setId><variable name="ElemId"/></setId>
283             <setElemDef><variable name="ElemDef"/></setElemDef>
284         </objectElement1D>
285         <objectGroup type="Element1D">
286             <setId><variable name="GroupId"/></setId>
287             <!-- <setPart>true</setPart> -->
288             <addInGroup><variable name="ElemId"/></addInGroup>
289         </objectGroup>
290     </rule>
291 
292     <rule keyword="*SET_PART_LIST_TITLE">
293         <nextLine/>
294         <parseRule>
295             <parse format="C" length="80" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupName"/></parse>
296         </parseRule>
297         <nextLine/>
298         <parseRule>
299             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
300         </parseRule>
301         <objectGroup type="Group">
302             <setId><variable name="GroupId"/></setId>
303             <setName><variable name="GroupName"/></setName>
304         </objectGroup>
305         <nextLine/>
306         <doWhile>
```

```
307             <condition>
308                 <notfind value="*"><curLine/></notfind>
309             </condition>
310             <body>
311                 <parseRule>
312                     <parse format="UI" length="10" offset="0" repeat="8" separator="none"><variableAssign
    name="GGroupId"/></parse>
313                 </parseRule>
314                 <nextLine/>
315             </body>
316         </doWhile>
317         <objectGroup type="Group">
318             <setId><variable name="GroupId"/></setId>
319             <addInGroup>
320                 <sourceId>*ELEMENT_BEAM *ELEMENT_DISCRETE *ELEMENT_SHELL *ELEMENT_SHELL_THICKNESS
    *ELEMENT_SOLID</sourceId>
321                 <variable name="GGroupId"/>
322             </addInGroup>
323         </objectGroup>
324      </rule>
325 
326      <rule keyword="*SET_PART_LIST">
327         <nextLine/>
328         <parseRule>
329             <parse format="UIpos" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
330         </parseRule>
331         <objectGroup type="Group">
332             <setId><variable name="GroupId"/></setId>
333         </objectGroup>
334         <nextLine/>
335         <doWhile>
336             <condition>
337                 <notfind value="*"><curLine/></notfind>
338             </condition>
339             <body>
340                 <parseRule>
341                     <parse format="UI" length="10" offset="0" repeat="8" separator="none"><variableAssign
    name="GGroupId"/></parse>
342                 </parseRule>
343                 <nextLine/>
344             </body>
345         </doWhile>
346         <objectGroup type="Group">
347             <setId><variable name="GroupId"/></setId>
348             <addInGroup>
349                 <sourceId>*ELEMENT_BEAM *ELEMENT_DISCRETE *ELEMENT_SHELL *ELEMENT_SHELL_THICKNESS
    *ELEMENT_SOLID</sourceId>
350                 <variable name="GGroupId"/>
351             </addInGroup>
352         </objectGroup>
353      </rule>
354 
355      <rule keyword="*SET_SOLID_TITLE">
356         <nextLine/>
357         <parseRule>
358             <parse format="C" length="80" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupName"/></parse>
359         </parseRule>
360         <nextLine/>
361         <parseRule>
362             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
363         </parseRule>
364         <objectGroup type="Element3D">
365             <setId><variable name="GroupId"/></setId>
366             <setName><variable name="GroupName"/></setName>
367         </objectGroup>
368         <nextLine/>
369         <doWhile>
370             <condition>
371                 <notfind value="*"><curLine/></notfind>
372             </condition>
373             <body>
374                 <parseRule>
375                     <parse format="UI" length="10" offset="0" repeat="8" separator="none"><variableAssign
    name="Eid"/></parse>
376                 </parseRule>
377                 <nextLine/>
378             </body>
379         </doWhile>
380         <objectGroup type="Element3D">
381             <setId><variable name="GroupId"/></setId>
382             <addInGroup><variable name="Eid"/></addInGroup>
383         </objectGroup>
384      </rule>
385 
```

```
386     <rule keyword="*SET_SOLID">
387         <nextLine/>
388         <parseRule>
389             <parse format="UIpos" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
390         </parseRule>
391         <objectGroup type="Element3D">
392             <setId><variable name="GroupId"/></setId>
393         </objectGroup>
394         <nextLine/>
395         <doWhile>
396             <condition>
397                 <notfind value="*"><curLine/></notfind>
398             </condition>
399             <body>
400                 <parseRule>
401                     <parse format="UI" length="10" offset="0" repeat="8" separator="none"><variableAssign
    name="Eid"/></parse>
402                 </parseRule>
403                 <nextLine/>
404             </body>
405         </doWhile>
406         <objectGroup type="Element3D">
407             <setId><variable name="GroupId"/></setId>
408             <addInGroup><variable name="Eid"/></addInGroup>
409         </objectGroup>
410     </rule>
411
412
413     <rule keyword="*SET_NODE_LIST_TITLE">
414         <nextLine/>
415         <parseRule>
416             <parse format="C" length="80" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupName"/></parse>
417         </parseRule>
418         <nextLine/>
419         <parseRule>
420             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
421         </parseRule>
422         <objectGroup type="Node">
423             <setId><variable name="GroupId"/></setId>
424             <setName><variable name="GroupName"/></setName>
425         </objectGroup>
426         <nextLine/>
427         <doWhile>
428             <condition>
429                 <notfind value="*"><curLine/></notfind>
430             </condition>
431             <body>
432                 <parseRule>
433                     <parse format="UI" length="10" offset="0" repeat="8" separator="none"><variableAssign
    name="Nid"/></parse>
434                 </parseRule>
435                 <nextLine/>
436             </body>
437         </doWhile>
438         <objectGroup type="Node">
439             <setId><variable name="GroupId"/></setId>
440             <addInGroup><variable name="Nid"/></addInGroup>
441         </objectGroup>
442     </rule>
443
444     <rule keyword="*SET_NODE_LIST">
445         <nextLine/>
446         <parseRule>
447             <parse format="UIpos" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
448         </parseRule>
449         <objectGroup type="Node">
450             <setId><variable name="GroupId"/></setId>
451         </objectGroup>
452         <nextLine/>
453         <doWhile>
454             <condition>
455                 <notfind value="*"><curLine/></notfind>
456             </condition>
457             <body>
458                 <parseRule>
459                     <parse format="UI" length="10" offset="0" repeat="8" separator="none"><variableAssign
    name="Nid"/></parse>
460                 </parseRule>
461                 <nextLine/>
462             </body>
463         </doWhile>
464         <objectGroup type="Node">
465             <setId><variable name="GroupId"/></setId>
```

```
466             <addInGroup><variable name="Nid"/></addInGroup>
467         </objectGroup>
468     </rule>
469 
470     <rule keyword="*SET_BEAM_TITLE">
471         <nextLine/>
472         <parseRule>
473             <parse format="C" length="80" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupName"/></parse>
474         </parseRule>
475         <nextLine/>
476         <parseRule>
477             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
478         </parseRule>
479         <objectGroup type="Element1D">
480             <setId><variable name="GroupId"/></setId>
481             <setName><variable name="GroupName"/></setName>
482         </objectGroup>
483         <nextLine/>
484         <doWhile>
485             <condition>
486                 <notfind value="*"><curLine/></notfind>
487             </condition>
488             <body>
489                 <parseRule>
490                     <parse format="UI" length="10" offset="0" repeat="8" separator="none"><variableAssign
    name="EId"/></parse>
491                 </parseRule>
492                 <nextLine/>
493             </body>
494         </doWhile>
495         <objectGroup type="Element1D">
496             <setId><variable name="GroupId"/></setId>
497             <addInGroup><variable name="EId"/></addInGroup>
498         </objectGroup>
499     </rule>
500 
501     <rule keyword="*SET_BEAM">
502         <nextLine/>
503         <parseRule>
504             <parse format="UIpos" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
505         </parseRule>
506         <objectGroup type="Element1D">
507             <setId><variable name="GroupId"/></setId>
508         </objectGroup>
509         <nextLine/>
510         <doWhile>
511             <condition>
512                 <notfind value="*"><curLine/></notfind>
513             </condition>
514             <body>
515                 <parseRule>
516                     <parse format="UI" length="10" offset="0" repeat="8" separator="none"><variableAssign
    name="EId"/></parse>
517                 </parseRule>
518                 <nextLine/>
519             </body>
520         </doWhile>
521         <objectGroup type="Element1D">
522             <setId><variable name="GroupId"/></setId>
523             <addInGroup><variable name="EId"/></addInGroup>
524         </objectGroup>
525     </rule>
526 
527 
528     <rule keyword="*SET_SHELL_LIST_TITLE">
529         <nextLine/>
530         <parseRule>
531             <parse format="C" length="80" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupName"/></parse>
532         </parseRule>
533         <nextLine/>
534         <parseRule>
535             <parse format="UIpos" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
536         </parseRule>
537         <objectGroup type="Element2D">
538             <setId><variable name="GroupId"/></setId>
539             <setName><variable name="GroupName"/></setName>
540         </objectGroup>
541         <nextLine/>
542         <doWhile>
543             <condition>
544                 <notfind value="*"><curLine/></notfind>
545             </condition>
```

```
546             <body>
547                 <parseRule>
548                     <parse format="UI" length="10" offset="0" repeat="8" separator="none"><variableAssign
    name="Eid"/></parse>
549                 </parseRule>
550                 <nextLine/>
551             </body>
552         </doWhile>
553         <objectGroup type="Element2D">
554             <setId><variable name="GroupId"/></setId>
555             <addInGroup><variable name="Eid"/></addInGroup>
556         </objectGroup>
557     </rule>
558 
559     <rule keyword="*SET_SHELL_LIST">
560         <nextLine/>
561         <parseRule>
562             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
563         </parseRule>
564         <objectGroup type="Element2D">
565             <setId><variable name="GroupId"/></setId>
566         </objectGroup>
567         <nextLine/>
568         <doWhile>
569             <condition>
570                 <notfind value="*"><curLine/></notfind>
571             </condition>
572             <body>
573                 <parseRule>
574                     <parse format="UI" length="10" offset="0" repeat="8" separator="none"><variableAssign
    name="Eid"/></parse>
575                 </parseRule>
576                 <nextLine/>
577             </body>
578         </doWhile>
579         <objectGroup type="Element2D">
580             <setId><variable name="GroupId"/></setId>
581             <addInGroup><variable name="Eid"/></addInGroup>
582         </objectGroup>
583     </rule>
584 
585     <rule keyword="*DEFINE_COORDINATE_NODES">
586         <nextLine/>
587         <doWhile>
588             <condition>
589                 <notfind value="*">
590                     <curLine/>
591                 </notfind>
592             </condition>
593             <body>
594                 <parseRule>
595                     <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="FrameId"/></parse>
596                     <parse format="UI" length="10" offset="10" repeat="1" separator="none"><variableAssign
    name="N1"/></parse>
597                     <parse format="UI" length="10" offset="20" repeat="1" separator="none"><variableAssign
    name="N2"/></parse>
598                     <parse format="UI" length="10" offset="30" repeat="1" separator="none"><variableAssign
    name="N3"/></parse>
599                     <parse format="C" length="10" offset="50" repeat="1" separator="none"><variableAssign
    name="Dir"/></parse>
600                 </parseRule>
601                 <nextLine/>
602             </body>
603         </doWhile>
604         <objectFrame>
605             <setId><variable name="FrameId"/></setId>
606             <setOrigin type="nodeid"><variable name="N1"/></setOrigin>
607             <setFirstDirection><variable name="Dir"/></setFirstDirection>
608             <setFirstAxis type="nodeid"><variable name="N2"/></setFirstAxis>
609             <doIf>
610                 <condition>
611                     <equal value="X"><variable name="Dir"/></equal>
612                 </condition>
613                 <body>
614                     <setSecondDirection>Y</setSecondDirection>
615                 </body>
616             </doIf>
617             <doIf>
618                 <condition>
619                     <equal value="Y"><variable name="Dir"/></equal>
620                 </condition>
621                 <body>
622                     <setSecondDirection>Z</setSecondDirection>
623                 </body>
624             </doIf>
```

```
625             <doIf>
626                 <condition>
627                     <equal value="Z"><variable name="Dir"/></equal>
628                 </condition>
629                 <body>
630                     <setSecondDirection>X</setSecondDirection>
631                 </body>
632             </doIf>
633             <setPlane type="nodeid"><variable name="N3"/></setPlane>
634         </objectFrame>
635     </rule>
636 
637 
638     <rule keyword="*DEFINE_COORDINATE_NODES_TITLE">
639         <nextLine/>
640         <doWhile>
641             <condition>
642                 <notfind value="*">
643                     <curLine/>
644                 </notfind>
645             </condition>
646             <body>
647                 <parseRule>
648                     <parse format="C" length="80" offset="0" repeat="1" separator="none"><variableAssign
      name="FrameName"/></parse>
649                 </parseRule>
650                 <nextLine/>
651                 <parseRule>
652                     <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
      name="FrameId"/></parse>
653                     <parse format="UI" length="10" offset="10" repeat="1" separator="none"><variableAssign
      name="N1"/></parse>
654                     <parse format="UI" length="10" offset="20" repeat="1" separator="none"><variableAssign
      name="N2"/></parse>
655                     <parse format="UI" length="10" offset="30" repeat="1" separator="none"><variableAssign
      name="N3"/></parse>
656                     <parse format="C" length="10" offset="50" repeat="1" separator="none"><variableAssign
      name="Dir"/></parse>
657                 </parseRule>
658                 <nextLine/>
659             </body>
660         </doWhile>
661         <objectFrame>
662             <setId><variable name="FrameId"/></setId>
663             <setName><variable name="FrameName"/></setName>
664             <setOrigin type="nodeid"><variable name="N1"/></setOrigin>
665             <setFirstDirection><variable name="Dir"/></setFirstDirection>
666             <setFirstAxis type="nodeid"><variable name="N2"/></setFirstAxis>
667             <doIf>
668                 <condition>
669                     <equal value="X"><variable name="Dir"/></equal>
670                 </condition>
671                 <body>
672                     <setSecondDirection>Y</setSecondDirection>
673                 </body>
674             </doIf>
675             <doIf>
676                 <condition>
677                     <equal value="Y"><variable name="Dir"/></equal>
678                 </condition>
679                 <body>
680                     <setSecondDirection>Z</setSecondDirection>
681                 </body>
682             </doIf>
683             <doIf>
684                 <condition>
685                     <equal value="Z"><variable name="Dir"/></equal>
686                 </condition>
687                 <body>
688                     <setSecondDirection>X</setSecondDirection>
689                 </body>
690             </doIf>
691             <setPlane type="nodeid"><variable name="N3"/></setPlane>
692         </objectFrame>
693     </rule>
694 
695 
696 
697     <rule keyword="*DEFINE_COORDINATE_SYSTEM_TITLE">
698         <nextLine/>
699         <doWhile>
700             <condition>
701                 <notfind value="*">
702                     <curLine/>
703                 </notfind>
704             </condition>
705             <body>
```

```
706                 <parseRule>
707                     <parse format="C" length="80" offset="0" repeat="1" separator="none"><variableAssign
    name="FrameName"/></parse>
708                 </parseRule>
709                 <nextLine/>
710                 <parseRule>
711                     <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="FrameId"/></parse>
712                     <parse format="F" length="10" offset="10" repeat="3" separator="none"><variableAssign
    name="Org"/></parse>
713                     <parse format="F" length="10" offset="40" repeat="3" separator="none"><variableAssign
    name="First"/></parse>
714                 </parseRule>
715                 <nextLine/>
716                 <parseRule>
717                     <parse format="F" length="10" offset="0" repeat="3" separator="none"><variableAssign
    name="Second"/></parse>
718                 </parseRule>
719                 <nextLine/>
720             </body>
721         </doWhile>
722         <objectFrame>
723             <setId><variable name="FrameId"/></setId>
724             <setName><variable name="FrameName"/></setName>
725             <setOrigin type="coord"><variable name="Org"/></setOrigin>
726             <setFirstDirection>X</setFirstDirection>
727             <setFirstAxis type="coord"><variable name="First"/></setFirstAxis>
728             <setSecondDirection>Y</setSecondDirection>
729             <setPlane type="coord"><variable name="Second"/></setPlane>
730         </objectFrame>
731     </rule>
732 
733     <!--    <rule keyword="*CONSTRAINED_JOINT_SPHERICAL_ID">
734         <nextLine/>
735         <parseRule>
736             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="JointId"/></parse>
737         </parseRule>
738         <nextLine/>
739         <parseRule>
740             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="JointCenter"/></parse>
741         </parseRule>
742         <assign>
743             <value>spherical</value>
744             <variableAssign name="JointType"/>
745         </assign>
746         <objectJoint>
747             <setId><variable name="JointId"/></setId>
748             <setType>SPHERICAL</setType>
749             <setCenter><variable name="JointCenter"/></setCenter>
750             <setDof>0 0 0 1 1 1</setDof>
751         </objectJoint>
752     </rule>
753 
754     <rule keyword="*CONSTRAINED_JOINT_REVOLUTE_ID">
755         <nextLine/>
756         <parseRule>
757             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="JointId"/></parse>
758         </parseRule>
759         <nextLine/>
760         <parseRule>
761             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="JointAxis1"/></parse>
762             <parse format="UI" length="10" offset="10" repeat="1" separator="none"><variableAssign
    name="JointAxis2"/></parse>
763         </parseRule>
764         <assign>
765             <value>revolute</value>
766             <variableAssign name="JointType"/>
767         </assign>
768         <objectJoint>
769             <setId><variable name="JointId"/></setId>
770             <setType>REVOLUTE</setType>
771             <setCenter><variable name="JointAxis1"/></setCenter>
772             <setAxis><variable name="JointAxis2"/></setAxis>
773         </objectJoint>
774     </rule>
775 -->
776 
777     <rule keyword="*MAT_STRAIN_RATE_DEPENDENT_PLASTICITY_TITLE">
778         <nextLine/>
779         <nextLine/>
780         <parseRule>
781             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="PropId"/></parse>
```

```
782             <parse format="F" length="10" offset="10" repeat="1" separator="none"><variableAssign
      name="Rho"/></parse>
783             <parse format="F" length="10" offset="20" repeat="1" separator="none"><variableAssign
      name="E"/></parse>
784         </parseRule>
785         <nextLine/>
786         <objectParameter>
787             <setId><variable name="PropId"/></setId>
788             <setValue>
789                 <type>YoungModulus</type>
790                 <value><variable name="E"/></value>
791             </setValue>
792             <setValue>
793                 <type>Density</type>
794                 <value><variable name="Rho"/></value>
795             </setValue>
796         </objectParameter>
797     </rule>
798 
799     <rule keyword="*MAT_VISCOELASTIC_TITLE">
800         <nextLine/>
801         <nextLine/>
802         <parseRule>
803             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
      name="PropId"/></parse>
804             <parse format="F" length="10" offset="10" repeat="1" separator="none"><variableAssign
      name="Rho"/></parse>
805             <parse format="F" length="10" offset="20" repeat="1" separator="none"><variableAssign
      name="K"/></parse>
806             <parse format="F" length="10" offset="30" repeat="1" separator="none"><variableAssign
      name="G0"/></parse>
807             <parse format="F" length="10" offset="40" repeat="1" separator="none"><variableAssign
      name="Gi"/></parse>
808             <parse format="F" length="10" offset="50" repeat="1" separator="none"><variableAssign
      name="Beta"/></parse>
809         </parseRule>
810         <nextLine/>
811         <objectParameter>
812             <setId><variable name="PropId"/></setId>
813             <setValue>
814                 <type>BulkModulus</type>
815                 <value><variable name="K"/></value>
816             </setValue>
817             <setValue>
818                 <type>InstantaneousShearModulus</type>
819                 <value><variable name="G0"/></value>
820             </setValue>
821             <setValue>
822                 <type>InfiniteShearModulus</type>
823                 <value><variable name="Gi"/></value>
824             </setValue>
825             <setValue>
826                 <type>DecayConstant</type>
827                 <value><variable name="Beta"/></value>
828             </setValue>
829             <setValue>
830                 <type>Density</type>
831                 <value><variable name="Rho"/></value>
832             </setValue>
833         </objectParameter>
834     </rule>
835 
836     <rule keyword="*MAT_ELASTIC_FLUID_TITLE">
837         <nextLine/>
838         <nextLine/>
839         <parseRule>
840             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
      name="PropId"/></parse>
841             <parse format="F" length="10" offset="10" repeat="1" separator="none"><variableAssign
      name="Rho"/></parse>
842             <parse format="F" length="10" offset="60" repeat="1" separator="none"><variableAssign
      name="K"/></parse>
843         </parseRule>
844         <nextLine/>
845         <objectParameter>
846             <setId><variable name="PropId"/></setId>
847             <setValue>
848                 <type>BulkModulus</type>
849                 <value><variable name="K"/></value>
850             </setValue>
851             <setValue>
852                 <type>Density</type>
853                 <value><variable name="Rho"/></value>
854             </setValue>
855         </objectParameter>
856     </rule>
857 
```

```
858     <rule keyword="*MAT_GENERAL_VISCOELASTIC_TITLE">
859         <nextLine/>
860         <nextLine/>
861         <parseRule>
862             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="PropId"/></parse>
863             <parse format="F" length="10" offset="10" repeat="1" separator="none"><variableAssign
    name="Rho"/></parse>
864             <parse format="F" length="10" offset="20" repeat="1" separator="none"><variableAssign
    name="K"/></parse>
865         </parseRule>
866         <nextLine/>
867         <nextLine/>
868         <doWhile>
869             <condition>
870                 <notfind value="*">
871                     <curLine/>
872                 </notfind>
873             </condition>
874             <body>
875                 <parseRule>
876                     <parse format="F" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="G"/></parse>
877                     <parse format="F" length="10" offset="20" repeat="1" separator="none"><variableAssign
    name="Ki"/></parse>
878                 </parseRule>
879                 <nextLine/>
880             </body>
881         </doWhile>
882         <objectParameter>
883             <setId><variable name="PropId"/></setId>
884             <setValue>
885                 <type>BulkModulus</type>
886                 <value><variable name="K"/></value>
887             </setValue>
888             <setValue>
889                 <type>Density</type>
890                 <value><variable name="Rho"/></value>
891             </setValue>
892             <setValue>
893                 <type>ShearModulus</type>
894                 <value><variable name="G"/></value>
895             </setValue>
896             <setValue>
897                 <type>BulkRelaxationModulus</type>
898                 <value><variable name="Ki"/></value>
899             </setValue>
900         </objectParameter>
901     </rule>
902
903     <rule keyword="*MAT_FABRIC_TITLE">
904         <nextLine/>
905         <nextLine/>
906         <parseRule>
907             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="PropId"/></parse>
908             <parse format="F" length="10" offset="10" repeat="1" separator="none"><variableAssign
    name="Rho"/></parse>
909             <parse format="F" length="10" offset="20" repeat="1" separator="none"><variableAssign
    name="Ea"/></parse>
910             <parse format="F" length="10" offset="30" repeat="1" separator="none"><variableAssign
    name="Eb"/></parse>
911             <parse format="F" length="10" offset="40" repeat="1" separator="none"><variableAssign
    name="Ec"/></parse>
912         </parseRule>
913         <nextLine/>
914         <parseRule>
915             <parse format="F" length="10" offset="0" repeat="3" separator="none"><variableAssign
    name="G"/></parse>
916         </parseRule>
917         <nextLine/>
918         <objectParameter>
919             <setId><variable name="PropId"/></setId>
920             <setValue>
921                 <type>YoungModulusA</type>
922                 <value><variable name="Ea"/></value>
923             </setValue>
924             <setValue>
925                 <type>YoungModulusB</type>
926                 <value><variable name="Eb"/></value>
927             </setValue>
928             <setValue>
929                 <type>YoungModulusC</type>
930                 <value><variable name="Ec"/></value>
931             </setValue>
932             <setValue>
933                 <type>ShearModulus</type>
```

```
934                 <value><variable name="G"/></value>
935             </setValue>
936             <setValue>
937                 <type>Density</type>
938                 <value><variable name="Rho"/></value>
939             </setValue>
940         </objectParameter>
941     </rule>
942 
943     <rule keyword="*MAT_PLASTIC_KINEMATIC_TITLE">
944         <nextLine/>
945         <nextLine/>
946         <parseRule>
947             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
      name="PropId"/></parse>
948             <parse format="F" length="10" offset="10" repeat="1" separator="none"><variableAssign
      name="Rho"/></parse>
949             <parse format="F" length="10" offset="20" repeat="1" separator="none"><variableAssign
      name="E"/></parse>
950             <parse format="F" length="10" offset="40" repeat="1" separator="none"><variableAssign
      name="SigY"/></parse>
951         </parseRule>
952         <nextLine/>
953         <objectParameter>
954             <setId><variable name="PropId"/></setId>
955             <setValue>
956                 <type>YoungModulus</type>
957                 <value><variable name="E"/></value>
958             </setValue>
959             <setValue>
960                 <type>YeldStress</type>
961                 <value><variable name="SigY"/></value>
962             </setValue>
963             <setValue>
964                 <type>Density</type>
965                 <value><variable name="Rho"/></value>
966             </setValue>
967         </objectParameter>
968     </rule>
969 
970     <rule keyword="*MAT_PIECEWISE_LINEAR_PLASTICITY_TITLE">
971         <nextLine/>
972         <nextLine/>
973         <parseRule>
974             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
      name="PropId"/></parse>
975             <parse format="F" length="10" offset="10" repeat="1" separator="none"><variableAssign
      name="Rho"/></parse>
976             <parse format="F" length="10" offset="20" repeat="1" separator="none"><variableAssign
      name="E"/></parse>
977             <parse format="F" length="10" offset="40" repeat="1" separator="none"><variableAssign
      name="SigY"/></parse>
978         </parseRule>
979         <nextLine/>
980         <objectParameter>
981             <setId><variable name="PropId"/></setId>
982             <setValue>
983                 <type>YoungModulus</type>
984                 <value><variable name="E"/></value>
985             </setValue>
986             <setValue>
987                 <type>YeldStress</type>
988                 <value><variable name="SigY"/></value>
989             </setValue>
990             <setValue>
991                 <type>Density</type>
992                 <value><variable name="Rho"/></value>
993             </setValue>
994         </objectParameter>
995     </rule>
996 
997     <rule keyword="*MAT_ELASTIC_TITLE">
998         <nextLine/>
999         <nextLine/>
1000         <parseRule>
1001             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
      name="PropId"/></parse>
1002             <parse format="F" length="10" offset="10" repeat="1" separator="none"><variableAssign
      name="Rho"/></parse>
1003             <parse format="F" length="10" offset="20" repeat="1" separator="none"><variableAssign
      name="E"/></parse>
1004         </parseRule>
1005         <nextLine/>
1006         <objectParameter>
1007             <setId><variable name="PropId"/></setId>
1008             <setValue>
1009                 <type>YoungModulus</type>
```

```
1010                <value><variable name="E"/></value>
1011            </setValue>
1012            <setValue>
1013                <type>Density</type>
1014                <value><variable name="Rho"/></value>
1015            </setValue>
1016        </objectParameter>
1017    </rule>
1018
1019    <rule keyword="*MAT_MOONEY-RIVLIN_RUBBER_TITLE">
1020        <nextLine/>
1021        <nextLine/>
1022        <parseRule>
1023            <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="PropId"/></parse>
1024            <parse format="F" length="10" offset="10" repeat="1" separator="none"><variableAssign
    name="Rho"/></parse>
1025            <parse format="F" length="10" offset="30" repeat="1" separator="none"><variableAssign
    name="A"/></parse>
1026        </parseRule>
1027        <nextLine/>
1028        <objectParameter>
1029            <setId><variable name="PropId"/></setId>
1030            <setValue>
1031                <type>HalfShearModulus</type>
1032                <value><variable name="A"/></value>
1033            </setValue>
1034            <setValue>
1035                <type>Density</type>
1036                <value><variable name="Rho"/></value>
1037            </setValue>
1038        </objectParameter>
1039    </rule>
1040
1041    <rule keyword="*MAT_SIMPLIFIED_RUBBER/FOAM_TITLE">
1042        <nextLine/>
1043        <nextLine/>
1044        <parseRule>
1045            <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="PropId"/></parse>
1046            <parse format="F" length="10" offset="10" repeat="1" separator="none"><variableAssign
    name="Rho"/></parse>
1047            <parse format="F" length="10" offset="20" repeat="1" separator="none"><variableAssign
    name="K"/></parse>
1048        </parseRule>
1049        <nextLine/>
1050        <objectParameter>
1051            <setId><variable name="PropId"/></setId>
1052            <setValue>
1053                <type>BulkModulus</type>
1054                <value><variable name="K"/></value>
1055            </setValue>
1056            <setValue>
1057                <type>Density</type>
1058                <value><variable name="Rho"/></value>
1059            </setValue>
1060        </objectParameter>
1061    </rule>
1062
1063     <rule keyword="*MAT_RIGID_TITLE">
1064        <nextLine/>
1065        <nextLine/>
1066        <parseRule>
1067            <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
    name="PropId"/></parse>
1068            <parse format="F" length="10" offset="10" repeat="1" separator="none"><variableAssign
    name="Rho"/></parse>
1069            <parse format="F" length="10" offset="20" repeat="1" separator="none"><variableAssign
    name="E"/></parse>
1070        </parseRule>
1071        <nextLine/>
1072        <objectParameter>
1073            <setId><variable name="PropId"/></setId>
1074            <setValue>
1075                <type>YoungModulus</type>
1076                <value><variable name="E"/></value>
1077            </setValue>
1078            <setValue>
1079                <type>Density</type>
1080                <value><variable name="Rho"/></value>
1081            </setValue>
1082        </objectParameter>
1083    </rule>
1084
1085    <rule keyword="*SECTION_SHELL_TITLE">
1086        <nextLine/>
1087        <nextLine/>
```

```
1088         <parseRule>
1089             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
     name="SectId"/></parse>
1090         </parseRule>
1091         <nextLine/>
1092         <parseRule>
1093             <parse format="F" length="10" offset="0" repeat="4" separator="none"><variableAssign
     name="Thick"/></parse>
1094         </parseRule>
1095         <nextLine/>
1096         <objectParameter>
1097             <setId><variable name="SectId"/></setId>
1098             <setValue>
1099                 <type>Thickness</type>
1100                 <value><variable name="Thick"/></value>
1101             </setValue>
1102         </objectParameter>
1103     </rule>
1104 
1105     <rule keyword="*DEFINE_CURVE">
1106         <nextLine/>
1107         <parseRule>
1108             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
     name="CurveId"/></parse>
1109             <parse format="F" length="10" offset="20" repeat="1" separator="none"><variableAssign
     name="ScaleFactorAbs"/></parse>
1110             <parse format="F" length="10" offset="30" repeat="1" separator="none"><variableAssign
     name="ScaleFactorOrd"/></parse>
1111         </parseRule>
1112         <nextLine/>
1113         <objectParameter>
1114             <setId><variable name="CurveId"/></setId>
1115             <setValue>
1116                 <type>ScaleFactorAbscissa</type>
1117                 <value><variable name="ScaleFactorAbs"/></value>
1118             </setValue>
1119             <setValue>
1120                 <type>ScaleFactorOrdinate</type>
1121                 <value><variable name="ScaleFactorOrd"/></value>
1122             </setValue>
1123         </objectParameter>
1124     </rule>
1125 
1126     <rule keyword="*DEFINE_CURVE_TITLE">
1127         <nextLine/>
1128         <nextLine/>
1129         <parseRule>
1130             <parse format="UI" length="10" offset="0" repeat="1" separator="none"><variableAssign
     name="CurveId"/></parse>
1131             <parse format="F" length="10" offset="20" repeat="1" separator="none"><variableAssign
     name="ScaleFactorAbs"/></parse>
1132             <parse format="F" length="10" offset="30" repeat="1" separator="none"><variableAssign
     name="ScaleFactorOrd"/></parse>
1133         </parseRule>
1134         <nextLine/>
1135         <objectParameter>
1136             <setId><variable name="CurveId"/></setId>
1137             <setValue>
1138                 <type>ScaleFactorAbscissa</type>
1139                 <value><variable name="ScaleFactorAbs"/></value>
1140             </setValue>
1141             <setValue>
1142                 <type>ScaleFactorOrdinate</type>
1143                 <value><variable name="ScaleFactorOrd"/></value>
1144             </setValue>
1145         </objectParameter>
1146     </rule>
1147 </format_description>
```

20.2 Example of file for LS Dyna format (comma separated value)

```
1 <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2 <!DOCTYPE format_description SYSTEM "FormatRules.dtd">
3 <format_description>
4     <formatInformation format="LSDyna_separator">
5         <comment>$</comment>
6         <includeFile>*INCLUDE</includeFile>
7         <separator>,</separator>
8     </formatInformation>
9 
10 
11     <meshComponent>
12         <componentNode>*NODE</componentNode>
13         <componentElement1D>*ELEMENT_BEAM *ELEMENT_DISCRETE</componentElement1D>
```

```
14          <componentElement2D>*ELEMENT_SHELL *ELEMENT_SHELL_THICKNESS
        *ELEMENT_SHELL_BETA</componentElement2D>
15          <componentElement3D>*ELEMENT_SOLID</componentElement3D>
16          <componentGNode>*SET_NODE_LIST_TITLE *SET_NODE_LIST *PART</componentGNode>
17          <componentGElement1D>*SET_BEAM_TITLE *SET_BEAM *PART</componentGElement1D>
18          <componentGElement2D>*SET_SHELL_LIST_TITLE *PART *SET_SHELL_LIST</componentGElement2D>
19          <componentGElement3D>*SET_SOLID_TITLE *SET_SOLID *PART</componentGElement3D>
20          <componentGGroup>*SET_PART_LIST_TITLE *SET_PART_LIST *PART</componentGGroup>
21          <componentFrame>*DEFINE_COORDINATE_NODES *DEFINE_COORDINATE_SYSTEM_TITLE
        *DEFINE_COORDINATE_NODES_TITLE</componentFrame>
22          <componentModelParameter>*PART *MAT_GENERAL_VISCOELASTIC_TITLE *DEFINE_CURVE_TITLE *DEFINE_CURVE
        *MAT_PIECEWISE_LINEAR_PLASTICITY_TITLE *MAT_VISCOELASTIC_TITLE *MAT_FABRIC_TITLE *MAT_ELASTIC_FLUID_TITLE
        *MAT_RIGID_TITLE *MAT_ELASTIC_TITLE *MAT_PLASTIC_KINEMATIC_TITLE *MAT_STRAIN_RATE_DEPENDENT_PLASTICITY_TITLE
        *ELEMENT_SHELL_THICKNESS *ELEMENT_SHELL_BETA *SECTION_SHELL_TITLE</componentModelParameter>
23          <elementOrdering>
24              <penta>1 2 5 4 3 6</penta>
25          </elementOrdering>
26      </meshComponent>
27
28
29      <rule keyword="*NODE">
30          <nextLine/>
31          <doWhile>
32              <condition>
33                  <notfind value="*"><curLine/></notfind>
34              </condition>
35              <body>
36                  <parseRule>
37                      <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
        name="NodeId"/></parse>
38                      <parse format="F" length="0" repeat="3" offset="1" separator="none"><variableAssign
        name="coord"/></parse>
39                  </parseRule>
40                  <nextLine/>
41              </body>
42          </doWhile>
43          <objectNode>
44              <setId><variable name="NodeId"/></setId>
45              <setCoord><variable name="coord"/></setCoord>
46          </objectNode>
47      </rule>
48
49      <rule keyword="*PART">
50          <nextLine/>
51          <parseRule>
52              <parse format="C" length="0" offset="0" repeat="1" separator="none"
        default="part"><variableAssign name="partname"/></parse>
53          </parseRule>
54          <nextLine/>
55          <parseRule>
56              <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
        name="PartId"/></parse>
57              <parse format="UI" length="0" offset="2" repeat="1" separator="none"><variableAssign
        name="MatId"/></parse>
58           </parseRule>
59          <nextLine/>
60          <objectGroup type="Id">
61              <setId>
62                  <sourceId>*ELEMENT_BEAM *ELEMENT_DISCRETE *ELEMENT_SHELL *ELEMENT_SHELL_THICKNESS
        *ELEMENT_SOLID</sourceId>
63                  <variable name="PartId"/>
64              </setId>
65              <setName><variable name="partname"/></setName>
66              <setPart>true</setPart>
67          </objectGroup>
68          <objectParameter>
69              <setId>
70                  <variable name="PartId"/>
71              </setId>
72              <setValue>
73                  <type>materialID</type>
74                  <value><variable name="MatId"/></value>
75              </setValue>
76          </objectParameter>
77      </rule>
78
79      <rule keyword="*INCLUDE">
80          <nextLine/>
81          <parseRule>
82              <parse format="C" length="0" offset="0" repeat="1" separator="none"><variableAssign
        name="HBMfile"/></parse>
83          </parseRule>
84          <nextLine/>
85          <objectModelFile>
86              <setFile><variable name="HBMfile"/></setFile>
87          </objectModelFile>
88      </rule>
```

```
89
90
91      <rule keyword="*ELEMENT_SOLID">
92          <nextLine/>
93          <doWhile>
94              <condition>
95                  <notfind value="*">
96                      <curLine/>
97                  </notfind>
98              </condition>
99              <body>
100                  <parseRule>
101                      <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
        name="ElemId"/></parse>
102                      <parse format="UI" length="0" offset="1" repeat="1" separator="none"><variableAssign
        name="GroupId"/></parse>
103                      <parse format="UI" length="0" offset="2" repeat="8" separator="none"
        offsetrepeat="1"><variableAssign name="ElemDef"/></parse>
104                  </parseRule>
105                  <nextLine/>
106              </body>
107          </doWhile>
108          <objectElement3D>
109              <setId><variable name="ElemId"/></setId>
110              <setElemDef><variable name="ElemDef"/></setElemDef>
111          </objectElement3D>
112          <objectGroup type="Element3D">
113              <setId><variable name="GroupId"/></setId>
114              <!-- <setPart>true</setPart> -->
115              <addInGroup><variable name="ElemId"/></addInGroup>
116          </objectGroup>
117      </rule>
118
119      <rule keyword="*ELEMENT_SHELL">
120          <nextLine/>
121          <doWhile>
122              <condition>
123                  <notfind value="*">
124                      <curLine/>
125                  </notfind>
126              </condition>
127              <body>
128                  <parseRule>
129                      <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
        name="ElemId"/></parse>
130                      <parse format="UI" length="0" offset="1" repeat="1" separator="none"><variableAssign
        name="GroupId"/></parse>
131                      <parse format="UI" length="0" offset="2" repeat="4" separator="none"
        offsetrepeat="1"><variableAssign name="ElemDef"/></parse>
132                  </parseRule>
133                  <nextLine/>
134              </body>
135          </doWhile>
136          <objectElement2D>
137              <setId><variable name="ElemId"/></setId>
138              <setElemDef><variable name="ElemDef"/></setElemDef>
139          </objectElement2D>
140          <objectGroup type="Element2D">
141              <setId><variable name="GroupId"/></setId>
142              <!-- <setPart>true</setPart> -->
143              <addInGroup><variable name="ElemId"/></addInGroup>
144          </objectGroup>
145      </rule>
146
147      <rule keyword="*ELEMENT_SHELL_THICKNESS">
148          <nextLine/>
149          <doWhile>
150              <condition>
151                  <notfind value="*">
152                      <curLine/>
153                  </notfind>
154              </condition>
155              <body>
156                  <parseRule>
157                      <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
        name="ElemId"/></parse>
158                      <parse format="UI" length="0" offset="1" repeat="1" separator="none"><variableAssign
        name="GroupId"/></parse>
159                      <parse format="UI" length="0" offset="2" repeat="4" separator="none"
        offsetrepeat="1"><variableAssign name="ElemDef"/></parse>
160                  </parseRule>
161                  <nextLine/>
162                  <parseRule>
163                      <parse format="F" length="0" offset="0" repeat="4" separator="none"
        offsetrepeat="1"><variableAssign name="Thick"/></parse>
164                  </parseRule>
165                  <nextLine/>
```

```
166             </body>
167         </doWhile>
168         <objectElement2D>
169             <setId><variable name="ElemId"/></setId>
170             <setElemDef><variable name="ElemDef"/></setElemDef>
171         </objectElement2D>
172         <objectGroup type="Element2D">
173             <setId><variable name="GroupId"/></setId>
174             <!-- <setPart>true</setPart> -->
175             <addInGroup><variable name="ElemId"/></addInGroup>
176         </objectGroup>
177         <objectParameter>
178             <setId><variable name="ElemId"/></setId>
179             <setValue>
180                 <type>Thickness</type>
181                 <value><variable name="Thick"/></value>
182             </setValue>
183         </objectParameter>
184     </rule>
185 
186     <rule keyword="*ELEMENT_SHELL_BETA">
187         <nextLine/>
188         <doWhile>
189             <condition>
190                 <notfind value="*">
191                     <curLine/>
192                 </notfind>
193             </condition>
194             <body>
195                 <parseRule>
196                     <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="ElemId"/></parse>
197                     <parse format="UI" length="0" offset="1" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
198                     <parse format="UI" length="0" offset="2" repeat="4" separator="none"
    offsetrepeat="1"><variableAssign name="ElemDef"/></parse>
199                 </parseRule>
200                 <nextLine/>
201                 <parseRule>
202                     <parse format="F" length="0" offset="0" repeat="4" separator="none"
    offsetrepeat="1"><variableAssign name="Thick"/></parse>
203                 </parseRule>
204                 <nextLine/>
205             </body>
206         </doWhile>
207         <objectElement2D>
208             <setId><variable name="ElemId"/></setId>
209             <setElemDef><variable name="ElemDef"/></setElemDef>
210         </objectElement2D>
211         <objectGroup type="Element2D">
212             <setId><variable name="GroupId"/></setId>
213             <!-- <setPart>true</setPart> -->
214             <addInGroup><variable name="ElemId"/></addInGroup>
215         </objectGroup>
216         <objectParameter>
217             <setId><variable name="ElemId"/></setId>
218             <setValue>
219                 <type>Thickness</type>
220                 <value><variable name="Thick"/></value>
221             </setValue>
222         </objectParameter>
223     </rule>
224 
225     <rule keyword="*ELEMENT_BEAM">
226         <nextLine/>
227         <doWhile>
228             <condition>
229                 <notfind value="*">
230                     <curLine/>
231                 </notfind>
232             </condition>
233             <body>
234                 <parseRule>
235                     <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="ElemId"/></parse>
236                     <parse format="UI" length="0" offset="1" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
237                     <parse format="UIpos" length="0" offset="2" repeat="2" separator="none"
    offsetrepeat="1"><variableAssign name="ElemDef"/></parse>
238                 </parseRule>
239                 <nextLine/>
240 
241             </body>
242         </doWhile>
243         <objectElement1D>
244             <setId><variable name="ElemId"/></setId>
245             <setElemDef><variable name="ElemDef"/></setElemDef>
```

```
246         </objectElement1D>
247         <objectGroup type="Element1D">
248             <setId><variable name="GroupId"/></setId>
249             <!-- <setPart>true</setPart> -->
250             <addInGroup><variable name="ElemId"/></addInGroup>
251         </objectGroup>
252     </rule>
253 
254     <rule keyword="*ELEMENT_DISCRETE">
255         <nextLine/>
256         <doWhile>
257             <condition>
258                 <notfind value="*">
259                     <curLine/>
260                 </notfind>
261             </condition>
262             <body>
263                 <parseRule>
264                     <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="ElemId"/></parse>
265                     <parse format="UI" length="0" offset="1" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
266                     <parse format="UIpos" length="0" offset="2" repeat="2" separator="none"
    offsetrepeat="1"><variableAssign name="ElemDef"/></parse>
267                 </parseRule>
268                 <nextLine/>
269             </body>
270         </doWhile>
271         <objectElement1D>
272             <setId><variable name="ElemId"/></setId>
273             <setElemDef><variable name="ElemDef"/></setElemDef>
274         </objectElement1D>
275         <objectGroup type="Element1D">
276             <setId><variable name="GroupId"/></setId>
277             <!-- <setPart>true</setPart> -->
278             <addInGroup><variable name="ElemId"/></addInGroup>
279         </objectGroup>
280     </rule>
281 
282     <rule keyword="*SET_PART_LIST_TITLE">
283         <nextLine/>
284         <parseRule>
285             <parse format="C" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupName"/></parse>
286         </parseRule>
287         <nextLine/>
288         <parseRule>
289             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
290         </parseRule>
291         <objectGroup type="Group">
292             <setId><variable name="GroupId"/></setId>
293             <setName><variable name="GroupName"/></setName>
294         </objectGroup>
295         <nextLine/>
296         <doWhile>
297             <condition>
298                 <notfind value="*"><curLine/></notfind>
299             </condition>
300             <body>
301                 <parseRule>
302                     <parse format="UI" length="0" offset="0" repeat="8" separator="none"
    offsetrepeat="1"><variableAssign name="GGroupId"/></parse>
303                 </parseRule>
304                 <nextLine/>
305             </body>
306         </doWhile>
307         <objectGroup type="Group">
308             <setId><variable name="GroupId"/></setId>
309             <addInGroup>
310                 <sourceId>*ELEMENT_BEAM *ELEMENT_DISCRETE *ELEMENT_SHELL *ELEMENT_SHELL_THICKNESS
    *ELEMENT_SOLID</sourceId>
311                 <variable name="GGroupId"/>
312             </addInGroup>
313         </objectGroup>
314     </rule>
315 
316     <rule keyword="*SET_PART_LIST">
317         <nextLine/>
318         <parseRule>
319             <parse format="UIpos" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
320         </parseRule>
321         <objectGroup type="Group">
322             <setId><variable name="GroupId"/></setId>
323         </objectGroup>
324         <nextLine/>
```

```
325         <doWhile>
326             <condition>
327                 <notfind value="*"><curLine/></notfind>
328             </condition>
329             <body>
330                 <parseRule>
331                     <parse format="UI" length="0" offset="0" repeat="8" separator="none"
    offsetrepeat="1"><variableAssign name="GGroupId"/></parse>
332                 </parseRule>
333                 <nextLine/>
334             </body>
335         </doWhile>
336         <objectGroup type="Group">
337             <setId><variable name="GroupId"/></setId>
338             <addInGroup>
339                 <sourceId>*ELEMENT_BEAM *ELEMENT_DISCRETE *ELEMENT_SHELL *ELEMENT_SHELL_THICKNESS
    *ELEMENT_SOLID</sourceId>
340                 <variable name="GGroupId"/>
341             </addInGroup>
342         </objectGroup>
343     </rule>
344 
345     <rule keyword="*SET_SOLID_TITLE">
346         <nextLine/>
347         <parseRule>
348             <parse format="C" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupName"/></parse>
349         </parseRule>
350         <nextLine/>
351         <parseRule>
352             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
353         </parseRule>
354         <objectGroup type="Element3D">
355             <setId><variable name="GroupId"/></setId>
356             <setName><variable name="GroupName"/></setName>
357         </objectGroup>
358         <nextLine/>
359         <doWhile>
360             <condition>
361                 <notfind value="*"><curLine/></notfind>
362             </condition>
363             <body>
364                 <parseRule>
365                     <parse format="UI" length="0" offset="0" repeat="8" separator="none"
    offsetrepeat="1"><variableAssign name="Eid"/></parse>
366                 </parseRule>
367                 <nextLine/>
368             </body>
369         </doWhile>
370         <objectGroup type="Element3D">
371             <setId><variable name="GroupId"/></setId>
372             <addInGroup><variable name="Eid"/></addInGroup>
373         </objectGroup>
374     </rule>
375 
376     <rule keyword="*SET_SOLID">
377         <nextLine/>
378         <parseRule>
379             <parse format="UIpos" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
380         </parseRule>
381         <objectGroup type="Element3D">
382             <setId><variable name="GroupId"/></setId>
383         </objectGroup>
384         <nextLine/>
385         <doWhile>
386             <condition>
387                 <notfind value="*"><curLine/></notfind>
388             </condition>
389             <body>
390                 <parseRule>
391                     <parse format="UI" length="0" offset="0" repeat="8" separator="none"
    offsetrepeat="1"><variableAssign name="Eid"/></parse>
392                 </parseRule>
393                 <nextLine/>
394             </body>
395         </doWhile>
396         <objectGroup type="Element3D">
397             <setId><variable name="GroupId"/></setId>
398             <addInGroup><variable name="Eid"/></addInGroup>
399         </objectGroup>
400     </rule>
401 
402     <rule keyword="*SET_NODE_LIST_TITLE">
403         <nextLine/>
404         <parseRule>
```

```
405             <parse format="C" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupName"/></parse>
406         </parseRule>
407         <nextLine/>
408         <parseRule>
409             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
410         </parseRule>
411         <objectGroup type="Node">
412             <setId><variable name="GroupId"/></setId>
413             <setName><variable name="GroupName"/></setName>
414         </objectGroup>
415         <nextLine/>
416         <doWhile>
417             <condition>
418                 <notfind value="*"><curLine/></notfind>
419             </condition>
420             <body>
421                 <parseRule>
422                     <parse format="UI" length="0" offset="0" repeat="8" separator="none"
    offsetrepeat="1"><variableAssign name="Nid"/></parse>
423                 </parseRule>
424                 <nextLine/>
425             </body>
426         </doWhile>
427         <objectGroup type="Node">
428             <setId><variable name="GroupId"/></setId>
429             <addInGroup><variable name="Nid"/></addInGroup>
430         </objectGroup>
431     </rule>
432 
433     <rule keyword="*SET_NODE_LIST">
434         <nextLine/>
435         <parseRule>
436             <parse format="UIpos" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
437         </parseRule>
438         <objectGroup type="Node">
439             <setId><variable name="GroupId"/></setId>
440         </objectGroup>
441         <nextLine/>
442         <doWhile>
443             <condition>
444                 <notfind value="*"><curLine/></notfind>
445             </condition>
446             <body>
447                 <parseRule>
448                     <parse format="UI" length="0" offset="0" repeat="8" separator="none"
    offsetrepeat="1"><variableAssign name="Nid"/></parse>
449                 </parseRule>
450                 <nextLine/>
451             </body>
452         </doWhile>
453         <objectGroup type="Node">
454             <setId><variable name="GroupId"/></setId>
455             <addInGroup><variable name="Nid"/></addInGroup>
456         </objectGroup>
457     </rule>
458 
459     <rule keyword="*SET_BEAM_TITLE">
460         <nextLine/>
461         <parseRule>
462             <parse format="C" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupName"/></parse>
463         </parseRule>
464         <nextLine/>
465         <parseRule>
466             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
467         </parseRule>
468         <objectGroup type="Element1D">
469             <setId><variable name="GroupId"/></setId>
470             <setName><variable name="GroupName"/></setName>
471         </objectGroup>
472         <nextLine/>
473         <doWhile>
474             <condition>
475                 <notfind value="*"><curLine/></notfind>
476             </condition>
477             <body>
478                 <parseRule>
479                     <parse format="UI" length="0" offset="0" repeat="8" separator="none"
    offsetrepeat="1"><variableAssign name="EId"/></parse>
480                 </parseRule>
481                 <nextLine/>
482             </body>
483         </doWhile>
```

```
484         <objectGroup type="Element1D">
485             <setId><variable name="GroupId"/></setId>
486             <addInGroup><variable name="EId"/></addInGroup>
487         </objectGroup>
488     </rule>
489 
490     <rule keyword="*SET_BEAM">
491         <nextLine/>
492         <parseRule>
493             <parse format="UIpos" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
494         </parseRule>
495         <objectGroup type="Element1D">
496             <setId><variable name="GroupId"/></setId>
497         </objectGroup>
498         <nextLine/>
499         <doWhile>
500             <condition>
501                 <notfind value="*"><curLine/></notfind>
502             </condition>
503             <body>
504                 <parseRule>
505                     <parse format="UI" length="0" offset="0" repeat="8" separator="none"
    offsetrepeat="1"><variableAssign name="EId"/></parse>
506                 </parseRule>
507                 <nextLine/>
508             </body>
509         </doWhile>
510         <objectGroup type="Element1D">
511             <setId><variable name="GroupId"/></setId>
512             <addInGroup><variable name="EId"/></addInGroup>
513         </objectGroup>
514     </rule>
515 
516 
517     <rule keyword="*SET_SHELL_LIST_TITLE">
518         <nextLine/>
519         <parseRule>
520             <parse format="C" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupName"/></parse>
521         </parseRule>
522         <nextLine/>
523         <parseRule>
524             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
525         </parseRule>
526         <objectGroup type="Element2D">
527             <setId><variable name="GroupId"/></setId>
528             <setName><variable name="GroupName"/></setName>
529         </objectGroup>
530         <nextLine/>
531         <doWhile>
532             <condition>
533                 <notfind value="*"><curLine/></notfind>
534             </condition>
535             <body>
536                 <parseRule>
537                     <parse format="UI" length="0" offset="0" repeat="8" separator="none"
    offsetrepeat="1"><variableAssign name="Eid"/></parse>
538                 </parseRule>
539                 <nextLine/>
540             </body>
541         </doWhile>
542         <objectGroup type="Element2D">
543             <setId><variable name="GroupId"/></setId>
544             <addInGroup><variable name="Eid"/></addInGroup>
545         </objectGroup>
546     </rule>
547 
548     <rule keyword="*SET_SHELL_LIST">
549         <nextLine/>
550         <parseRule>
551             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="GroupId"/></parse>
552         </parseRule>
553         <objectGroup type="Element2D">
554             <setId><variable name="GroupId"/></setId>
555             <setName><variable name="GroupName"/></setName>
556         </objectGroup>
557         <nextLine/>
558         <doWhile>
559             <condition>
560                 <notfind value="*"><curLine/></notfind>
561             </condition>
562             <body>
563                 <parseRule>
564                     <parse format="UI" length="0" offset="0" repeat="8" separator="none"
```

```
      offsetrepeat="1"><variableAssign name="Eid"/></parse>
565                 </parseRule>
566                 <nextLine/>
567             </body>
568         </doWhile>
569         <objectGroup type="Element2D">
570             <setId><variable name="GroupId"/></setId>
571             <addInGroup><variable name="Eid"/></addInGroup>
572         </objectGroup>
573      </rule>
574
575
576      <rule keyword="*SET_SHELL_LIST">
577         <nextLine/>
578         <parseRule>
579             <parse format="UIpos" length="0" offset="0" repeat="1" separator="none"><variableAssign
      name="GroupId"/></parse>
580         </parseRule>
581         <objectGroup type="Element2D">
582             <setId><variable name="GroupId"/></setId>
583         </objectGroup>
584         <nextLine/>
585         <doWhile>
586             <condition>
587                 <notfind value="*"><curLine/></notfind>
588             </condition>
589             <body>
590                 <parseRule>
591                     <parse format="UI" length="0" offset="0" repeat="8" separator="none"
      offsetrepeat="1"><variableAssign name="Eid"/></parse>
592                 </parseRule>
593                 <nextLine/>
594             </body>
595         </doWhile>
596         <objectGroup type="Element2D">
597             <setId><variable name="GroupId"/></setId>
598             <addInGroup><variable name="Eid"/></addInGroup>
599         </objectGroup>
600      </rule>
601
602
603      <rule keyword="*DEFINE_COORDINATE_NODES">
604         <nextLine/>
605         <doWhile>
606             <condition>
607                 <notfind value="*">
608                     <curLine/>
609                 </notfind>
610             </condition>
611             <body>
612                 <parseRule>
613                     <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
      name="FrameId"/></parse>
614                     <parse format="UI" length="0" offset="1" repeat="1" separator="none"><variableAssign
      name="N1"/></parse>
615                     <parse format="UI" length="0" offset="2" repeat="1" separator="none"><variableAssign
      name="N2"/></parse>
616                     <parse format="UI" length="0" offset="3" repeat="1" separator="none"><variableAssign
      name="N3"/></parse>
617                     <parse format="C" length="0" offset="5" repeat="1" separator="none"><variableAssign
      name="Dir"/></parse>
618                 </parseRule>
619                 <nextLine/>
620             </body>
621         </doWhile>
622         <objectFrame>
623             <setId><variable name="FrameId"/></setId>
624             <setOrigin type="nodeid"><variable name="N1"/></setOrigin>
625             <setFirstDirection><variable name="Dir"/></setFirstDirection>
626             <setFirstAxis type="nodeid"><variable name="N2"/></setFirstAxis>
627             <doIf>
628                 <condition>
629                     <equal value="X"><variable name="Dir"/></equal>
630                 </condition>
631                 <body>
632                     <setSecondDirection>Y</setSecondDirection>
633                 </body>
634             </doIf>
635             <doIf>
636                 <condition>
637                     <equal value="Y"><variable name="Dir"/></equal>
638                 </condition>
639                 <body>
640                     <setSecondDirection>Z</setSecondDirection>
641                 </body>
642             </doIf>
643             <doIf>
```

```
644                 <condition>
645                     <equal value="Z"><variable name="Dir"/></equal>
646                 </condition>
647                 <body>
648                     <setSecondDirection>X</setSecondDirection>
649                 </body>
650             </doIf>
651             <setPlane type="nodeid"><variable name="N3"/></setPlane>
652         </objectFrame>
653     </rule>
654 
655 
656     <rule keyword="*DEFINE_COORDINATE_NODES_TITLE">
657         <nextLine/>
658         <doWhile>
659             <condition>
660                 <notfind value="*">
661                     <curLine/>
662                 </notfind>
663             </condition>
664             <body>
665                 <parseRule>
666                     <parse format="C" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="FrameName"/></parse>
667                 </parseRule>
668                 <nextLine/>
669                 <parseRule>
670                     <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="FrameId"/></parse>
671                     <parse format="UI" length="0" offset="1" repeat="1" separator="none"><variableAssign
    name="N1"/></parse>
672                     <parse format="UI" length="0" offset="2" repeat="1" separator="none"><variableAssign
    name="N2"/></parse>
673                     <parse format="UI" length="0" offset="3" repeat="1" separator="none"><variableAssign
    name="N3"/></parse>
674                     <parse format="C" length="0" offset="5" repeat="1" separator="none"><variableAssign
    name="Dir"/></parse>
675                 </parseRule>
676                 <nextLine/>
677             </body>
678         </doWhile>
679         <objectFrame>
680             <setId><variable name="FrameId"/></setId>
681             <setName><variable name="FrameName"/></setName>
682             <setOrigin type="nodeid"><variable name="N1"/></setOrigin>
683             <setFirstDirection><variable name="Dir"/></setFirstDirection>
684             <setFirstAxis type="nodeid"><variable name="N2"/></setFirstAxis>
685             <doIf>
686                 <condition>
687                     <equal value="X"><variable name="Dir"/></equal>
688                 </condition>
689                 <body>
690                     <setSecondDirection>Y</setSecondDirection>
691                 </body>
692             </doIf>
693             <doIf>
694                 <condition>
695                     <equal value="Y"><variable name="Dir"/></equal>
696                 </condition>
697                 <body>
698                     <setSecondDirection>Z</setSecondDirection>
699                 </body>
700             </doIf>
701             <doIf>
702                 <condition>
703                     <equal value="Z"><variable name="Dir"/></equal>
704                 </condition>
705                 <body>
706                     <setSecondDirection>X</setSecondDirection>
707                 </body>
708             </doIf>
709             <setPlane type="nodeid"><variable name="N3"/></setPlane>
710         </objectFrame>
711     </rule>
712 
713 
714 
715     <rule keyword="*DEFINE_COORDINATE_SYSTEM_TITLE">
716         <nextLine/>
717         <doWhile>
718             <condition>
719                 <notfind value="*">
720                     <curLine/>
721                 </notfind>
722             </condition>
723             <body>
724                 <parseRule>
```

```
725                     <parse format="C" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="FrameName"/></parse>
726                 </parseRule>
727                 <nextLine/>
728                 <parseRule>
729                     <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="FrameId"/></parse>
730                     <parse format="F" length="0" offset="1" repeat="3" separator="none"><variableAssign
    name="Org"/></parse>
731                     <parse format="F" length="0" offset="4" repeat="3" separator="none"
    offsetrepeat="1"><variableAssign name="First"/></parse>
732                 </parseRule>
733                 <nextLine/>
734                 <parseRule>
735                     <parse format="F" length="0" offset="0" repeat="3" separator="none"
    offsetrepeat="1"><variableAssign name="Second"/></parse>
736                 </parseRule>
737                 <nextLine/>
738             </body>
739         </doWhile>
740         <objectFrame>
741             <setId><variable name="FrameId"/></setId>
742             <setName><variable name="FrameName"/></setName>
743             <setOrigin type="coord"><variable name="Org"/></setOrigin>
744             <setFirstDirection>X</setFirstDirection>
745             <setFirstAxis type="coord"><variable name="First"/></setFirstAxis>
746             <setSecondDirection>Y</setSecondDirection>
747             <setPlane type="coord"><variable name="Second"/></setPlane>
748         </objectFrame>
749     </rule>
750 
751     <rule keyword="*MAT_PIECEWISE_LINEAR_PLASTICITY_TITLE">
752         <nextLine/>
753         <nextLine/>
754         <parseRule>
755             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="PropId"/></parse>
756             <parse format="F" length="0" offset="1" repeat="1" separator="none"><variableAssign
    name="Rho"/></parse>
757             <parse format="F" length="0" offset="2" repeat="1" separator="none"><variableAssign
    name="E"/></parse>
758             <parse format="F" length="0" offset="4" repeat="1" separator="none"><variableAssign
    name="SigY"/></parse>
759         </parseRule>
760         <nextLine/>
761         <objectParameter>
762             <setId><variable name="PropId"/></setId>
763             <setValue>
764                 <type>YoungModulus</type>
765                 <value><variable name="E"/></value>
766             </setValue>
767             <setValue>
768                 <type>YeldStress</type>
769                 <value><variable name="SigY"/></value>
770             </setValue>
771             <setValue>
772                 <type>Density</type>
773                 <value><variable name="Rho"/></value>
774             </setValue>
775         </objectParameter>
776     </rule>
777 
778     <rule keyword="*MAT_VISCOELASTIC_TITLE">
779         <nextLine/>
780         <nextLine/>
781         <parseRule>
782             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="PropId"/></parse>
783             <parse format="F" length="0" offset="1" repeat="1" separator="none"><variableAssign
    name="Rho"/></parse>
784             <parse format="F" length="0" offset="2" repeat="1" separator="none"><variableAssign
    name="K"/></parse>
785             <parse format="F" length="0" offset="3" repeat="1" separator="none"><variableAssign
    name="G0"/></parse>
786             <parse format="F" length="0" offset="4" repeat="1" separator="none"><variableAssign
    name="Gi"/></parse>
787             <parse format="F" length="0" offset="5" repeat="1" separator="none"><variableAssign
    name="Beta"/></parse>
788         </parseRule>
789         <nextLine/>
790         <objectParameter>
791             <setId><variable name="PropId"/></setId>
792             <setValue>
793                 <type>BulkModulus</type>
794                 <value><variable name="K"/></value>
795             </setValue>
796             <setValue>
```

```
797                 <type>InstantaneousShearModulus</type>
798                 <value><variable name="G0"/></value>
799             </setValue>
800             <setValue>
801                 <type>InfiniteShearModulus</type>
802                 <value><variable name="Gi"/></value>
803             </setValue>
804             <setValue>
805                 <type>DecayConstant</type>
806                 <value><variable name="Beta"/></value>
807             </setValue>
808             <setValue>
809                 <type>Density</type>
810                 <value><variable name="Rho"/></value>
811             </setValue>
812         </objectParameter>
813     </rule>
814
815     <rule keyword="*MAT_STRAIN_RATE_DEPENDENT_PLASTICITY_TITLE">
816         <nextLine/>
817         <nextLine/>
818         <parseRule>
819             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="PropId"/></parse>
820             <parse format="F" length="0" offset="1" repeat="1" separator="none"><variableAssign
    name="Rho"/></parse>
821             <parse format="F" length="0" offset="2" repeat="1" separator="none"><variableAssign
    name="E"/></parse>
822         </parseRule>
823         <nextLine/>
824         <objectParameter>
825             <setId><variable name="PropId"/></setId>
826             <setValue>
827                 <type>YoungModulus</type>
828                 <value><variable name="E"/></value>
829             </setValue>
830             <setValue>
831                 <type>Density</type>
832                 <value><variable name="Rho"/></value>
833             </setValue>
834         </objectParameter>
835     </rule>
836
837     <rule keyword="*MAT_FABRIC_TITLE">
838         <nextLine/>
839         <nextLine/>
840         <parseRule>
841             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="PropId"/></parse>
842             <parse format="F" length="0" offset="1" repeat="1" separator="none"><variableAssign
    name="Rho"/></parse>
843             <parse format="F" length="0" offset="2" repeat="1" separator="none" default="0"><variableAssign
    name="Ea"/></parse>
844             <parse format="F" length="0" offset="3" repeat="1" separator="none" default="0"><variableAssign
    name="Eb"/></parse>
845             <parse format="F" length="0" offset="4" repeat="1" separator="none" default="0"><variableAssign
    name="Ec"/></parse>
846         </parseRule>
847         <nextLine/>
848         <parseRule>
849             <parse format="F" length="0" offset="0" repeat="3" separator="none" default="0"><variableAssign
    name="G"/></parse>
850         </parseRule>
851         <nextLine/>
852         <objectParameter>
853             <setId><variable name="PropId"/></setId>
854             <setValue>
855                 <type>YoungModulusA</type>
856                 <value><variable name="Ea"/></value>
857             </setValue>
858             <setValue>
859                 <type>YoungModulusB</type>
860                 <value><variable name="Eb"/></value>
861             </setValue>
862             <setValue>
863                 <type>YoungModulusC</type>
864                 <value><variable name="Ec"/></value>
865             </setValue>
866             <setValue>
867                 <type>ShearModulus</type>
868                 <value><variable name="G"/></value>
869             </setValue>
870             <setValue>
871                 <type>Density</type>
872                 <value><variable name="Rho"/></value>
873             </setValue>
874         </objectParameter>
```

```
875     </rule>
876
877     <rule keyword="*MAT_ELASTIC_FLUID_TITLE">
878         <nextLine/>
879         <nextLine/>
880         <parseRule>
881             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
        name="PropId"/></parse>
882             <parse format="F" length="0" offset="1" repeat="1" separator="none"><variableAssign
        name="Rho"/></parse>
883             <parse format="F" length="0" offset="6" repeat="1" separator="none"><variableAssign
        name="K"/></parse>
884         </parseRule>
885         <nextLine/>
886         <objectParameter>
887             <setId><variable name="PropId"/></setId>
888             <setValue>
889                 <type>BulkModulus</type>
890                 <value><variable name="K"/></value>
891             </setValue>
892             <setValue>
893                 <type>Density</type>
894                 <value><variable name="Rho"/></value>
895             </setValue>
896         </objectParameter>
897     </rule>
898
899     <rule keyword="*MAT_PLASTIC_KINEMATIC_TITLE">
900         <nextLine/>
901         <nextLine/>
902         <parseRule>
903             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
        name="PropId"/></parse>
904             <parse format="F" length="0" offset="1" repeat="1" separator="none"><variableAssign
        name="Rho"/></parse>
905             <parse format="F" length="0" offset="2" repeat="1" separator="none"><variableAssign
        name="E"/></parse>
906             <parse format="F" length="0" offset="4" repeat="1" separator="none"><variableAssign
        name="SigY"/></parse>
907         </parseRule>
908         <nextLine/>
909         <objectParameter>
910             <setId><variable name="PropId"/></setId>
911             <setValue>
912                 <type>YoungModulus</type>
913                 <value><variable name="E"/></value>
914             </setValue>
915             <setValue>
916                 <type>YeldStress</type>
917                 <value><variable name="SigY"/></value>
918             </setValue>
919             <setValue>
920                 <type>Density</type>
921                 <value><variable name="Rho"/></value>
922             </setValue>
923         </objectParameter>
924     </rule>
925
926     <rule keyword="*MAT_ELASTIC_TITLE">
927         <nextLine/>
928         <nextLine/>
929         <parseRule>
930             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
        name="PropId"/></parse>
931             <parse format="F" length="0" offset="1" repeat="1" separator="none"><variableAssign
        name="Rho"/></parse>
932             <parse format="F" length="0" offset="2" repeat="1" separator="none"><variableAssign
        name="E"/></parse>
933         </parseRule>
934         <nextLine/>
935         <objectParameter>
936             <setId><variable name="PropId"/></setId>
937             <setValue>
938                 <type>YoungModulus</type>
939                 <value><variable name="E"/></value>
940             </setValue>
941             <setValue>
942                 <type>Density</type>
943                 <value><variable name="Rho"/></value>
944             </setValue>
945         </objectParameter>
946     </rule>
947
948     <rule keyword="*MAT_RIGID_TITLE">
949         <nextLine/>
950         <nextLine/>
951         <parseRule>
```

```
952             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="PropId"/></parse>
953             <parse format="F" length="0" offset="1" repeat="1" separator="none"><variableAssign
    name="Rho"/></parse>
954             <parse format="F" length="0" offset="2" repeat="1" separator="none"><variableAssign
    name="E"/></parse>
955         </parseRule>
956         <nextLine/>
957         <objectParameter>
958             <setId><variable name="PropId"/></setId>
959             <setValue>
960                 <type>YoungModulus</type>
961                 <value><variable name="E"/></value>
962             </setValue>
963             <setValue>
964                 <type>Density</type>
965                 <value><variable name="Rho"/></value>
966             </setValue>
967         </objectParameter>
968      </rule>
969
970      <rule keyword="*SECTION_SHELL_TITLE">
971         <nextLine/>
972         <nextLine/>
973         <parseRule>
974             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="SectId"/></parse>
975         </parseRule>
976         <nextLine/>
977         <parseRule>
978             <parse format="F" length="0" offset="0" repeat="4" separator="none" offsetrepeat="1"
    default="0"><variableAssign name="Thick"/></parse>
979         </parseRule>
980         <nextLine/>
981         <objectParameter>
982             <setId><variable name="SectId"/></setId>
983             <setValue>
984                 <type>Thickness</type>
985                 <value><variable name="Thick"/></value>
986             </setValue>
987         </objectParameter>
988      </rule>
989
990      <rule keyword="*DEFINE_CURVE">
991         <nextLine/>
992         <parseRule>
993             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="CurveId"/></parse>
994             <parse format="F" length="0" offset="2" repeat="1" separator="none"><variableAssign
    name="ScaleFactorAbs"/></parse>
995             <parse format="F" length="0" offset="3" repeat="1" separator="none"><variableAssign
    name="ScaleFactorOrd"/></parse>
996         </parseRule>
997         <nextLine/>
998         <objectParameter>
999             <setId><variable name="CurveId"/></setId>
1000             <setValue>
1001                 <type>ScaleFactorAbscissa</type>
1002                 <value><variable name="ScaleFactorAbs"/></value>
1003             </setValue>
1004             <setValue>
1005                 <type>ScaleFactorOrdinate</type>
1006                 <value><variable name="ScaleFactorOrd"/></value>
1007             </setValue>
1008         </objectParameter>
1009      </rule>
1010
1011      <rule keyword="*DEFINE_CURVE_TITLE">
1012         <nextLine/>
1013         <nextLine/>
1014         <parseRule>
1015             <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
    name="CurveId"/></parse>
1016             <parse format="F" length="0" offset="2" repeat="1" separator="none"><variableAssign
    name="ScaleFactorAbs"/></parse>
1017             <parse format="F" length="0" offset="3" repeat="1" separator="none"><variableAssign
    name="ScaleFactorOrd"/></parse>
1018         </parseRule>
1019         <nextLine/>
1020         <objectParameter>
1021             <setId><variable name="CurveId"/></setId>
1022             <setValue>
1023                 <type>ScaleFactorAbscissa</type>
1024                 <value><variable name="ScaleFactorAbs"/></value>
1025             </setValue>
1026             <setValue>
1027                 <type>ScaleFactorOrdinate</type>
```

```
1028                <value><variable name="ScaleFactorOrd"/></value>
1029            </setValue>
1030        </objectParameter>
1031    </rule>
1032
1033    <rule keyword="*MAT_GENERAL_VISCOELASTIC_TITLE">
1034        <nextLine/>
1035        <nextLine/>
1036        <parseRule>
1037            <parse format="UI" length="0" offset="0" repeat="1" separator="none"><variableAssign
      name="PropId"/></parse>
1038            <parse format="F" length="0" offset="1" repeat="1" separator="none"><variableAssign
      name="Rho"/></parse>
1039            <parse format="F" length="0" offset="2" repeat="1" separator="none"><variableAssign
      name="K"/></parse>
1040        </parseRule>
1041        <nextLine/>
1042        <nextLine/>
1043        <doWhile>
1044            <condition>
1045                <notfind value="*">
1046                    <curLine/>
1047                </notfind>
1048            </condition>
1049            <body>
1050                <parseRule>
1051                    <parse format="F" length="0" offset="0" repeat="1" separator="none"
      default="0"><variableAssign name="G"/></parse>
1052                    <parse format="F" length="0" offset="2" repeat="1" separator="none"
      default="0"><variableAssign name="Ki"/></parse>
1053                </parseRule>
1054                <nextLine/>
1055            </body>
1056        </doWhile>
1057        <objectParameter>
1058            <setId><variable name="PropId"/></setId>
1059            <setValue>
1060                <type>BulkModulus</type>
1061                <value><variable name="K"/></value>
1062            </setValue>
1063            <setValue>
1064                <type>Density</type>
1065                <value><variable name="Rho"/></value>
1066            </setValue>
1067            <setValue>
1068                <type>ShearModulus</type>
1069                <value><variable name="G"/></value>
1070            </setValue>
1071            <setValue>
1072                <type>BulkRelaxationModulus</type>
1073                <value><variable name="Ki"/></value>
1074            </setValue>
1075        </objectParameter>
1076    </rule>
1077
1078
1079 </format_description>
```

20.3 Example of file for PamCrash format

```
1 <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2 <!DOCTYPE format_description SYSTEM "FormatRules.dtd">
3 <format_description>
4     <formatInformation format="PamCrash">
5         <comment>$ #</comment>
6         <includeFile>INCLU</includeFile>
7     </formatInformation>
8
9
10     <meshComponent>
11         <componentNode>NODE</componentNode>
12         <componentElement1D>BEAM SPRING/</componentElement1D>
13         <componentElement2D>SHELL</componentElement2D>
14         <componentElement3D>SOLID</componentElement3D>
15         <componentGNode>GROUP</componentGNode>
16         <componentGElement1D>GROUP</componentGElement1D>
17         <componentGElement2D>GROUP</componentGElement2D>
18         <componentGElement3D>GROUP</componentGElement3D>
19         <componentGGroup>GROUP</componentGGroup>
20         <componentFrame>FRAME</componentFrame>
21         <componentModelParameter>MATER PART</componentModelParameter>
22         <elementOrdering>
23             <penta>1 2 5 4 3 6</penta>
24         </elementOrdering>
```

```
25      </meshComponent>
26
27
28
29      <rule keyword="PART">
30          <parseRule>
31              <parse format="UI" length="8" offset="8" repeat="1" separator="none"><variableAssign
        name="PartId"/></parse>
32              <parse format="C" length="8" offset="16" repeat="1" separator="none"><variableAssign
        name="PartType"/></parse>
33          </parseRule>
34          <objectGroup type="Id">
35              <setId>
36                  <sourceId>SOLID SHELL BEAM SPRING</sourceId>
37                  <variable name="PartId"/>
38              </setId>
39              <setPart>true</setPart>
40          </objectGroup>
41          <doIf>
42              <condition>
43                  <find value="SHELL"><variable name="PartType"/></find>
44              </condition>
45              <body>
46  <!--                <parseRule>
47                      <curLine/>
48                      <parse format="UIpos" length="8" offset="8" repeat="1" separator="none"><variableAssign
        name="PartId"/></parse>
49                  </parseRule> -->
50                  <nextLine/>
51                  <nextLine/>
52                  <nextLine/>
53                  <nextLine/>
54                  <parseRule>
55                      <parse format="F" length="10" offset="0" repeat="1" separator="none"><variableAssign
        name="ParamThick"/></parse>
56                  </parseRule>
57                  <nextLine/>
58                  <objectParameter>
59                      <setId><variable name="PartId"/></setId>
60                      <setValue>
61                          <type>Thickness</type>
62                          <value><variable name="ParamThick"/></value>
63                      </setValue>
64                  </objectParameter>
65              </body>
66          </doIf>
67          <doIf>
68              <condition>
69                  <notfind value="SHELL"><variable name="PartType"/></notfind>
70              </condition>
71              <body>
72                  <nextLine/>
73              </body>
74          </doIf>
75      </rule>
76
77
78      <rule keyword="INCLU">
79          <parseRule>
80              <parse format="C" length="80" offset="8" repeat="1" separator="none"><variableAssign
        name="HBMfile"/></parse>
81          </parseRule>
82          <nextLine/>
83          <objectModelFile>
84              <setFile><variable name="HBMfile"/></setFile>
85          </objectModelFile>
86      </rule>
87
88
89      <rule keyword="NODE">
90          <doWhile>
91              <condition>
92                  <find value="NODE"><curLine/></find>
93              </condition>
94              <body>
95                  <parseRule>
96                      <parse format="UI" length="8" offset="8" repeat="1" separator="none"><variableAssign
        name="NodeId"/></parse>
97                      <parse format="F" length="16" repeat="3" offset="16" separator="none"><variableAssign
        name="coord"/></parse>
98                  </parseRule>
99                  <nextLine/>
100             </body>
101         </doWhile>
102         <objectNode>
103             <setId><variable name="NodeId"/></setId>
104             <setCoord><variable name="coord"/></setCoord>
```

```
105         </objectNode>
106      </rule>
107
108      <rule keyword="SOLID">
109          <doWhile>
110              <condition>
111                  <find value="SOLID"><curLine/></find>
112              </condition>
113              <body>
114                  <parseRule>
115                      <parse format="UI" length="8" offset="8" repeat="1" separator="none"><variableAssign
     name="ElemId"/></parse>
116                      <parse format="UI" length="8" offset="16" repeat="1" separator="none"><variableAssign
     name="GroupId"/></parse>
117                  </parseRule>
118                  <nextLine/>
119                  <parseRule>
120                      <parse format="UIpos" length="8" offset="16" repeat="8"
     separator="none"><variableAssign name="ElemDef"/></parse>
121                  </parseRule>
122                  <nextLine/>
123              </body>
124          </doWhile>
125          <objectElement3D>
126              <setId><variable name="ElemId"/></setId>
127              <setElemDef><variable name="ElemDef"/></setElemDef>
128          </objectElement3D>
129          <objectGroup type="Element3D">
130              <setId><variable name="GroupId"/></setId>
131              <!-- <setPart>true</setPart> -->
132              <addInGroup><variable name="ElemId"/></addInGroup>
133          </objectGroup>
134      </rule>
135
136      <rule keyword="SHELL">
137          <doWhile>
138              <condition>
139                  <find value="SHELL"><curLine/></find>
140              </condition>
141              <body>
142                  <parseRule>
143                      <parse format="UI" length="8" offset="8" repeat="1" separator="none"><variableAssign
     name="ElemId"/></parse>
144                      <parse format="UI" length="8" offset="16" repeat="1" separator="none"><variableAssign
     name="GroupId"/></parse>
145                      <parse format="UIpos" length="8" offset="24" repeat="4"
     separator="none"><variableAssign name="ElemDef"/></parse>
146                  </parseRule>
147                  <nextLine/>
148              </body>
149          </doWhile>
150          <objectElement2D>
151              <setId><variable name="ElemId"/></setId>
152              <setElemDef><variable name="ElemDef"/></setElemDef>
153          </objectElement2D>
154          <objectGroup type="Element2D">
155              <setId><variable name="GroupId"/></setId>
156              <!-- <setPart>true</setPart> -->
157              <addInGroup><variable name="ElemId"/></addInGroup>
158          </objectGroup>
159      </rule>
160
161      <rule keyword="BEAM">
162          <doWhile>
163              <condition>
164                  <find value="BEAM"><curLine/></find>
165              </condition>
166              <body>
167                  <parseRule>
168                      <parse format="UI" length="8" offset="8" repeat="1" separator="none"><variableAssign
     name="ElemId"/></parse>
169                      <parse format="UI" length="8" offset="16" repeat="1" separator="none"><variableAssign
     name="GroupId"/></parse>
170                      <parse format="UI" length="8" offset="24" repeat="2" separator="none"><variableAssign
     name="ElemDef"/></parse>
171                  </parseRule>
172                  <nextLine/>
173              </body>
174          </doWhile>
175          <objectElement1D>
176              <setId><variable name="ElemId"/></setId>
177              <setElemDef><variable name="ElemDef"/></setElemDef>
178          </objectElement1D>
179          <objectGroup type="Element1D">
180              <setId><variable name="GroupId"/></setId>
181              <!-- <setPart>true</setPart> -->
182              <addInGroup><variable name="ElemId"/></addInGroup>
```

```
183         </objectGroup>
184      </rule>
185
186
187      <rule keyword="SPRING/">
188         <doWhile>
189            <condition>
190               <find value="SPRING"><curLine/></find>
191            </condition>
192            <body>
193               <parseRule>
194                  <parse format="UI" length="8" offset="8" repeat="1" separator="none"><variableAssign
      name="ElemId"/></parse>
195                  <parse format="UI" length="8" offset="16" repeat="1" separator="none"><variableAssign
      name="GroupId"/></parse>
196                  <parse format="UI" length="8" offset="24" repeat="2" separator="none"><variableAssign
      name="ElemDef"/></parse>
197               </parseRule>
198               <nextLine/>
199            </body>
200         </doWhile>
201         <objectElement1D>
202            <setId><variable name="ElemId"/></setId>
203            <setElemDef><variable name="ElemDef"/></setElemDef>
204         </objectElement1D>
205         <objectGroup type="Element1D">
206            <setId><variable name="GroupId"/></setId>
207            <!-- <setPart>true</setPart> -->
208            <addInGroup><variable name="ElemId"/></addInGroup>
209         </objectGroup>
210      </rule>
211
212      <rule keyword="GROUP">
213         <parseRule>
214            <parse format="C" length="256" offset="8" repeat="1" separator="none"><variableAssign
      name="GroupName"/></parse>
215         </parseRule>
216         <nextLine/>
217         <objectGroup type="Group">
218            <setId><variable name="GroupName"/></setId>
219            <setName><variable name="GroupName"/></setName>
220         </objectGroup>
221         <doWhile>
222            <condition>
223               <notfind value="END"><curLine/></notfind>
224            </condition>
225            <body>
226               <parseRule>
227                  <parse format="C" length="0" offset="1" repeat="1" separator="ws"><variableAssign
      name="GroupTypeContent"/></parse>
228                  <parse format="C" length="0" offset="2" repeat="8" separator="ws"
      offsetrepeat="1"><variableAssign name="Content"/></parse>
229               </parseRule>
230               <nextLine/>
231               <!-- <forEach>
232                  <variable name="Content"/>
233                  <variableAssign name="CurContent"/>
234                  <body>
235                     <doIf>
236                        <condition>
237                           <find value=":" pos="any"><variable name="CurContent"/></find>
238                        </condition>
239                        <body>
240                           <parseRule>
241                              <variable name="CurContent"/>
242                              <parse format="UIpos" length="0" offset="0" repeat="1"
      separator=":"><variableAssign name="ContentToAddStart"/></parse>
243                              <parse format="UIpos" length="0" offset="1" repeat="1"
      separator=":"><variableAssign name="ContentToAddEnd"/></parse>
244                           </parseRule>
245                           <generateId>
246                              <start><variable name="ContentToAddStart"/></start>
247                              <end><variable name="ContentToAddEnd"/></end>
248                              <variableAssign name="ContentToAdd"/>
249                           </generateId>
250                        </body>
251                     </doIf>
252                     <doIf>
253                        <condition>
254                           <notfind value=":" pos="any"><variable name="CurContent"/></notfind>
255                        </condition>
256                        <body>
257                           <parseRule>
258                              <variable name="CurContent"/>
259                              <parse format="UIpos" length="8" offset="0" repeat="1"
      separator="none"><variableAssign name="ContentToAdd"/></parse>
260                           </parseRule>
```

```
261                         </body>
262                     </doIf>
263                 </body>
264             </forEach> -->
265             <doIf>
266                 <condition>
267                     <equal value="NOD"><variable name="GroupTypeContent"/></equal>
268                 </condition>
269                 <body>
270                     <forEach>
271                         <variable name="Content"/>
272                         <variableAssign name="CurContent"/>
273                         <body>
274                             <doIf>
275                                 <condition>
276                                     <find value=":" pos="any"><variable name="CurContent"/></find>
277                                 </condition>
278                                 <body>
279                                     <parseRule>
280                                         <variable name="CurContent"/>
281                                         <parse format="UIpos" length="0" offset="0" repeat="1"
    separator=":"><variableAssign name="ContentToAddStart"/></parse>
282                                         <parse format="UIpos" length="0" offset="1" repeat="1"
    separator=":"><variableAssign name="ContentToAddEnd"/></parse>
283                                     </parseRule>
284                                     <generateId>
285                                         <start><variable name="ContentToAddStart"/></start>
286                                         <end><variable name="ContentToAddEnd"/></end>
287                                         <variableAssign name="ContentToAdd"/>
288                                     </generateId>
289                                 </body>
290                             </doIf>
291                             <doIf>
292                                 <condition>
293                                     <notfind value=":" pos="any"><variable
    name="CurContent"/></notfind>
294                                 </condition>
295                                 <body>
296                                     <parseRule>
297                                         <variable name="CurContent"/>
298                                         <parse format="UIpos" length="8" offset="0" repeat="1"
    separator="none"><variableAssign name="ContentToAdd"/></parse>
299                                     </parseRule>
300                                 </body>
301                             </doIf>
302                         </body>
303                     </forEach>
304                     <append value="_Node">
305                         <variable name="GroupName"/>
306                         <variableAssign name="GroupName_Node"/>
307                     </append>
308                     <objectGroup type="Node">
309                         <setId><variable name="GroupName_Node"/></setId>
310                         <setName><variable name="GroupName_Node"/></setName>
311                         <addInGroup><variable name="ContentToAdd"/></addInGroup>
312                     </objectGroup>
313                     <objectGroup type="Group">
314                         <setId><variable name="GroupName"/></setId>
315                         <addInGroup><variable name="GroupName_Node"/></addInGroup>
316                     </objectGroup>
317                 </body>
318             </doIf>
319             <doIf>
320                 <condition>
321                     <equal value="ELE"><variable name="GroupTypeContent"/></equal>
322                 </condition>
323                 <body>
324                     <forEach>
325                         <variable name="Content"/>
326                         <variableAssign name="CurContent"/>
327                         <body>
328                             <doIf>
329                                 <condition>
330                                     <find value=":" pos="any"><variable name="CurContent"/></find>
331                                 </condition>
332                                 <body>
333                                     <parseRule>
334                                         <variable name="CurContent"/>
335                                         <parse format="UIpos" length="0" offset="0" repeat="1"
    separator=":"><variableAssign name="ContentToAddStart"/></parse>
336                                         <parse format="UIpos" length="0" offset="1" repeat="1"
    separator=":"><variableAssign name="ContentToAddEnd"/></parse>
337                                     </parseRule>
338                                     <generateId>
339                                         <start><variable name="ContentToAddStart"/></start>
340                                         <end><variable name="ContentToAddEnd"/></end>
341                                         <variableAssign name="ContentToAdd"/>
```

```
342                                     </generateId>
343                                 </body>
344                             </doIf>
345                             <doIf>
346                                 <condition>
347                                     <notfind value=":" pos="any"><variable
    name="CurContent"/></notfind>
348                                 </condition>
349                                 <body>
350                                     <parseRule>
351                                         <variable name="CurContent"/>
352                                         <parse format="UIpos" length="8" offset="0" repeat="1"
    separator="none"><variableAssign name="ContentToAdd"/></parse>
353                                     </parseRule>
354                                 </body>
355                             </doIf>
356                         </body>
357                     </forEach>
358                     <append value="_Element1D">
359                         <variable name="GroupName"/>
360                         <variableAssign name="GroupName_E1D"/>
361                     </append>
362                     <objectGroup type="Element1D">
363                         <setId><variable name="GroupName_E1D"/></setId>
364                         <setName><variable name="GroupName_E1D"/></setName>
365                         <addInGroup><variable name="ContentToAdd"/></addInGroup>
366                     </objectGroup>
367                     <objectGroup type="Group">
368                         <setId><variable name="GroupName"/></setId>
369                         <addInGroup><variable name="GroupName_E1D"/></addInGroup>
370                     </objectGroup>
371                     <append value="_Element2D">
372                         <variable name="GroupName"/>
373                         <variableAssign name="GroupName_E2D"/>
374                     </append>
375                     <objectGroup type="Element2D">
376                         <setId><variable name="GroupName_E2D"/></setId>
377                         <setName><variable name="GroupName_E2D"/></setName>
378                         <addInGroup><variable name="ContentToAdd"/></addInGroup>
379                     </objectGroup>
380                     <objectGroup type="Group">
381                         <setId><variable name="GroupName"/></setId>
382                         <addInGroup><variable name="GroupName_E2D"/></addInGroup>
383                     </objectGroup>
384                     <append value="_Element3D">
385                         <variable name="GroupName"/>
386                         <variableAssign name="GroupName_E3D"/>
387                     </append>
388                     <objectGroup type="Element3D">
389                         <setId><variable name="GroupName_E3D"/></setId>
390                         <setName><variable name="GroupName_E3D"/></setName>
391                         <addInGroup><variable name="ContentToAdd"/></addInGroup>
392                     </objectGroup>
393                     <objectGroup type="Group">
394                         <setId><variable name="GroupName"/></setId>
395                         <addInGroup><variable name="GroupName_E3D"/></addInGroup>
396                     </objectGroup>
397                 </body>
398             </doIf>
399             <doIf>
400                 <condition>
401                     <equal value="PART"><variable name="GroupTypeContent"/></equal>
402                 </condition>
403                 <body>
404                     <forEach>
405                         <variable name="Content"/>
406                         <variableAssign name="CurContent"/>
407                         <body>
408                             <doIf>
409                                 <condition>
410                                     <find value=":" pos="any"><variable name="CurContent"/></find>
411                                 </condition>
412                                 <body>
413                                     <parseRule>
414                                         <variable name="CurContent"/>
415                                         <parse format="UIpos" length="0" offset="0" repeat="1"
    separator=":"><variableAssign name="ContentToAddStart"/></parse>
416                                         <parse format="UIpos" length="0" offset="1" repeat="1"
    separator=":"><variableAssign name="ContentToAddEnd"/></parse>
417                                     </parseRule>
418                                     <generateId>
419                                         <start><variable name="ContentToAddStart"/></start>
420                                         <end><variable name="ContentToAddEnd"/></end>
421                                         <variableAssign name="ContentToAdd"/>
422                                     </generateId>
423                                 </body>
424                             </doIf>
```

```
425                                 <doIf>
426                                     <condition>
427                                         <notfind value=":" pos="any"><variable
    name="CurContent"/></notfind>
428                                     </condition>
429                                     <body>
430                                         <parseRule>
431                                             <variable name="CurContent"/>
432                                             <parse format="UIpos" length="8" offset="0" repeat="1"
    separator="none"><variableAssign name="ContentToAdd"/></parse>
433                                         </parseRule>
434                                     </body>
435                                 </doIf>
436                             </body>
437                         </forEach>
438                         <objectGroup type="Group">
439                             <setId><variable name="GroupName"/></setId>
440                             <sourceId>BEAM SPRING/ SHELL SOLID</sourceId>
441                             <addInGroup><variable name="ContentToAdd"/></addInGroup>
442                         </objectGroup>
443                     </body>
444                 </doIf>
445                 <clearVar>
446                     <variable name="GroupName_Node"/>
447                     <variable name="GroupName_E1D"/>
448                     <variable name="GroupName_E2D"/>
449                     <variable name="GroupName_E3D"/>
450                     <variable name="GroupTypeContent"/>
451                     <variable name="ContentToAdd"/>
452                     <variable name="Content"/>
453                     <variable name="ContentToAddStart"/>
454                     <variable name="ContentToAddEnd"/>
455                 </clearVar>
456             </body>
457         </doWhile>
458     </rule>
459
460     <rule keyword="FRAME">
461         <parseRule>
462             <parse format="UI" length="8" offset="8" repeat="1"><variableAssign name="FrameId"/></parse>
463             <parse format="C" length="8" offset="24" repeat="1"><variableAssign name="FrameType"/></parse>
464         </parseRule>
465         <nextLine/>
466         <parseRule>
467             <parse format="C" length="250" offset="4" repeat="1"><variableAssign name="FrameName"/></parse>
468         </parseRule>
469         <nextLine/>
470         <doIf>
471             <condition>
472                 <equal value="1"><variable name="FrameType"/></equal>
473             </condition>
474             <body>
475                 <parseRule>
476                     <parse format="UI" length="8" offset="8" repeat="1"><variableAssign
    name="FrameN1"/></parse>
477                     <parse format="UI" length="8" offset="16" repeat="1"><variableAssign
    name="FrameN2"/></parse>
478                     <parse format="UI" length="8" offset="24" repeat="1"><variableAssign
    name="FrameN3"/></parse>
479                 </parseRule>
480                 <nextLine/>
481                 <objectFrame>
482                     <setId><variable name="FrameId"/></setId>
483                     <setName><variable name="FrameName"/></setName>
484                     <setOrigin type="nodeid"><variable name="FrameN1"/></setOrigin>
485                     <setFirstDirection>X</setFirstDirection>
486                     <setFirstAxis type="nodeid"><variable name="FrameN2"/></setFirstAxis>
487                     <setSecondDirection>Y</setSecondDirection>
488                     <setPlane type="nodeid"><variable name="FrameN3"/></setPlane>
489                 </objectFrame>
490             </body>
491         </doIf>
492         <doIf>
493             <condition>
494                 <equal value="0"><variable name="FrameType"/></equal>
495             </condition>
496             <body>
497                 <parseRule>
498                     <parse format="F" length="16" offset="8" repeat="3"><variableAssign
    name="FrameV1"/></parse>
499                 </parseRule>
500                 <nextLine/>
501                 <parseRule>
502                     <parse format="F" length="16" offset="8" repeat="3"><variableAssign
    name="FrameV2"/></parse>
503                     <!-- <parse format="UIpos" length="8" offset="56" repeat="1"><variableAssign
    name="FrameN1"/></parse> -->
```

```
504                 </parseRule>
505                 <nextLine/>
506                 <objectFrame>
507                     <setId><variable name="FrameId"/></setId>
508                     <setName><variable name="FrameName"/></setName>
509                     <!-- <setOrigin><variable name="FrameN1"/></setOrigin> -->
510                     <setFirstDirection>X</setFirstDirection>
511                     <setFirstAxis type="vector"><variable name="FrameV1"/></setFirstAxis>
512                     <setSecondDirection>Y</setSecondDirection>
513                     <setPlane type="vector"><variable name="FrameV2"/></setPlane>
514                 </objectFrame>
515             </body>
516         </doIf>
517         <doIf>
518             <condition>
519                 <equal value="3"><variable name="FrameType"/></equal>
520             </condition>
521             <body>
522                 <parseRule>
523                     <parse format="UI" length="8" offset="8" repeat="1"><variableAssign
      name="FrameN1"/></parse>
524                     <parse format="UI" length="8" offset="16" repeat="1"><variableAssign
      name="FrameN2"/></parse>
525                     <parse format="UI" length="8" offset="24" repeat="1"><variableAssign
      name="FrameN3"/></parse>
526                 </parseRule>
527                 <nextLine/>
528                 <objectFrame>
529                     <setId><variable name="FrameId"/></setId>
530                     <setName><variable name="FrameName"/></setName>
531                     <setOrigin type="nodeid"><variable name="FrameN1"/></setOrigin>
532                     <setFirstDirection>Z</setFirstDirection>
533                     <setFirstAxis type="nodeid"><variable name="FrameN2"/></setFirstAxis>
534                     <setSecondDirection>X</setSecondDirection>
535                     <setPlane type="nodeid"><variable name="FrameN3"/></setPlane>
536                 </objectFrame>
537             </body>
538         </doIf>
539      </rule>
540
541
542
543
544      <rule keyword="MATER">
545         <parseRule>
546             <parse format="C" length="8" offset="16" repeat="1" separator="none"><variableAssign
      name="MatType"/></parse>
547         </parseRule>
548         <doIf>
549             <condition>
550                 <equal value="1"><variable name="MatType"/></equal>
551             </condition>
552             <body>
553                 <parseRule>
554                     <parse format="UI" length="8" offset="8" repeat="1" separator="none"><variableAssign
      name="PropId"/></parse>
555                     <parse format="F" length="16" offset="24" repeat="1" separator="none"><variableAssign
      name="Rho"/></parse>
556                 </parseRule>
557                 <nextLine/>
558                 <nextLine/>
559                 <nextLine/>
560                 <parseRule>
561                     <parse format="F" length="10" offset="0" repeat="1" separator="none"><variableAssign
      name="E"/></parse>
562                 </parseRule>
563                 <objectParameter>
564                     <setId><variable name="PropId"/></setId>
565                     <setValue>
566                         <type>ModuleG</type>
567                         <value><variable name="E"/></value>
568                     </setValue>
569                     <setValue>
570                         <type>Density</type>
571                         <value><variable name="Rho"/></value>
572                     </setValue>
573                 </objectParameter>
574             </body>
575         </doIf>
576         <nextLine/>
577      </rule>
578 </format_description>
```

21 Module Index

21.1 Modules

Here is a list of all modules:

22 Namespace Index

22.1 Namespace List

Here is a list of all namespaces with brief descriptions:

23 Hierarchical Index

23.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

24 Class Index

24.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

25 Module Documentation

25.1 Python package piper.anatomyDB

The piper.anatomyDB python package gives access to *anatomy database* queries.

Classes

- class anatomydb::LandmarkCont

  An interface class to store the landmark positions to anatomydb::FrameFactory.
- class anatomydb::Frame

  Store coordinates of a frame.
- class anatomydb::FrameFactory

  This class computes anatomical frames coordinate from landmark positions.

Functions

- void anatomydb::init (std::string const &databaseFile="")

  Initialize the database.

Queries

These functions are the available queries to the anatomical database.

- unsigned int anatomydb::getEntityId (std::string const &name)
- std::string anatomydb::getReferenceName (std::string const &name)
- bool anatomydb::isSynonymOf (std::string const &name1, std::string const &name2)
- std::string anatomydb::getReferenceNameNoThrow (std::string const &name)
- std::string anatomydb::getEntityDescription (std::string const &name)
- bool anatomydb::exists (std::string const &name)
- bool anatomydb::isBone (std::string const &name)
- bool anatomydb::isSkin (std::string const &name)
- bool anatomydb::isLandmark (std::string const &name)
- bool anatomydb::isEntitySubClassOf (std::string const &name, std::string const &className)
- bool anatomydb::isEntityPartOf (std::string const &entity, std::string const &parent, bool recursive=false)
- bool anatomydb::isEntityFromBibliography (std::string const &entity, std::string const &bibliography)
- std::vector< std::string > anatomydb::getSubClassOfList (std::string const &className)
- std::vector< std::string > anatomydb::getParentClassList (std::string const &name)
- std::vector< std::string > anatomydb::getPartOfList (std::string const &name)
- std::vector< std::string > anatomydb::getPartOfList (std::string const &name, std::string const &className)
- std::vector< std::string > anatomydb::getSubPartOfList (std::string const &name, std::string const &class↩ Name, bool recursive=false)
- std::vector< std::string > anatomydb::getSynonymList (std::string const &name, bool omitReference↩ Name=false)
- std::string anatomydb::getSynonymFromBibliography (std::string const &name, std::string const &from↩ Bibliography)
- std::vector< std::string > anatomydb::getEntityBibliographyList (std::string const &name)
- std::vector< std::string > anatomydb::getLandmarkBoneList (std::string const &name)
- std::vector< std::string > anatomydb::getSubClassOfFromBibliographyList (std::string const &className, std::string const &bibliography)
- std::vector< std::string > anatomydb::getAnatomicalEntityList ()
- std::vector< std::string > anatomydb::getLandmarkList ()
- std::vector< std::string > anatomydb::getJointList ()

25.1.1 Detailed Description

The piper.anatomyDB python package gives access to *anatomy database* queries.

25.1.2 Function Documentation

25.1.2.1 bool anatomydb::exists (std::string const & *name*)

25.1.2.2 std::vector<std::string> anatomydb::getAnatomicalEntityList ( )

25.1.2.3 std::vector<std::string> anatomydb::getEntityBibliographyList (std::string const & *name*)

25.1.2.4 std::string anatomydb::getEntityDescription (std::string const & *name*)

25.1.2.5 unsigned int anatomydb::getEntityId (std::string const & *name*)

25.1.2.6 std::vector<std::string> anatomydb::getJointList ( )

25.1.2.7 std::vector<std::string> anatomydb::getLandmarkBoneList (std::string const & *name*)

25.1.2.8 std::vector<std::string> anatomydb::getLandmarkList ( )

25.1.2.9 std::vector<std::string> anatomydb::getParentClassList (std::string const & *name*)

25.1.2.10 std::vector<std::string> anatomydb::getPartOfList (std::string const & *name*)

25.1.2.11 std::vector<std::string> anatomydb::getPartOfList (std::string const & *name,* std::string const & *className*)

25.1.2.12 std::string anatomydb::getReferenceName (std::string const & *name*)

25.1.2.13 std::string anatomydb::getReferenceNameNoThrow (std::string const & *name*)

25.1.2.14 std::vector<std::string> anatomydb::getSubClassOfFromBibliographyList (std::string const & *className,* std::string const & *bibliography*)

25.1.2.15 std::vector<std::string> anatomydb::getSubClassOfList (std::string const & *className*)

25.1.2.16 std::vector<std::string> anatomydb::getSubPartOfList (std::string const & *name,* std::string const & *className,* bool *recursive* = `false`)

25.1.2.17 std::string anatomydb::getSynonymFromBibliography (std::string const & *name,* std::string const & *fromBibliography*)

25.1.2.18 std::vector<std::string> anatomydb::getSynonymList (std::string const & *name,* bool *omitReferenceName* = `false`)

25.1.2.19 void anatomydb::init (std::string const & *databaseFile* = " ")

Initialize the database.

When used in the Piper application, the database is already initialized.

Parameters

databaseFile	optionnal path to the database file.

25.1.2.20 bool anatomydb::isBone (std::string const & *name*)

25.1.2.21 bool anatomydb::isEntityFromBibliography (std::string const & *entity,* std::string const & *bibliography*)

25.1.2.22 bool anatomydb::isEntityPartOf (std::string const & *entity,* std::string const & *parent,* bool *recursive* = `false`)

25.1.2.23 bool anatomydb::isEntitySubClassOf (std::string const & *name,* std::string const & *className*)

25.1.2.24 bool anatomydb::isLandmark (std::string const & *name*)

25.1.2.25 bool anatomydb::isSkin (std::string const & *name*)

25.1.2.26 bool anatomydb::isSynonymOf (std::string const & *name1,* std::string const & *name2*)

25.2 Python package piper.app

The piper.app python package gives access to application related data structures and functions.

Classes

- class piper::Context

  Application context.
- class piper::Project

  Piper project.
- class piper::ModuleParameter

  A piper::Project module parameters.

Functions

- std::string piper::tempDirectoryPath (std::string const &module="")

  Get the application specific temporary path.

Logging

These functions send a message to the application log, using the corresponding level.

- void piper::logDebug (std::string const &message)
- void piper::logInfo (std::string const &message)
- void piper::logWarning (std::string const &message)
- void piper::logError (std::string const &message)
- void piper::logSuccess (std::string const &message)
- void piper::logStart (std::string const &message)
- void piper::logDone (std::string const &message="")

Application signals

These functions send a signal to the application to indicate some thing has changed.

You should call the appropriate function if you modify the model or targets, they will make the application (display, internal data,...) update accordingly.

Warning

In batch mode these functions are not available

- void piper::modelUpdated ()

  Call it after you change any nodes coordinates.
- void piper::metadataChanged ()

  Call it after you change some metadata.
- void piper::targetChanged ()

  Call it after you change some target.
- void piper::historyListChanged ()

  Call it after you change project history.

Modules specific functions

- PIPERCOMMON_EXPORT void piper::physPosiDeform (piper::Project &model, piper::hbm::TargetList const &target)

  This function applies the physics based positioning deformation to model using target.
- PIPERCOMMON_EXPORT void piper::applyTargetScalingParameter (piper::Project &model, piper::hbm::↩ TargetList const &target)

  This function defines scaling parameter value defined in target to model.
- PIPERCOMMON_EXPORT void piper::readContourCL (piper::hbm::FEModel &c_fem, piper::hbm::Metadata &c_meta)

  This function defines reads the contourCL.xml file and populates the contourCL datastructs.
- PIPERCOMMON_EXPORT void piper::importSimplifiedScalableModel (piper::Project &project, std::string const &filepath)

  This function import Simplified Scalable Model from filepath in the project.
- PIPERCOMMON_EXPORT void piper::applyScalingTarget (piper::Project &project, piper::hbm::TargetList const &target, bool const &useKrigingWIntermediates=true)

  This function applies anthropometric dimension target and landmark targets in target to model project useKriging↩ WIntermediates: set true if intermediates target skin and bones are used for transformation.

25.2.1 Detailed Description

The piper.app python package gives access to application related data structures and functions.

25.2.2 Function Documentation

25.2.2.1 PIPERCOMMON_EXPORT void piper::applyScalingTarget (piper::Project & *project,* piper::hbm::TargetList const & *target,* bool const & *useKrigingWIntermediates* = `true`)

This function applies anthropometric dimension target and landmark targets in *target* to model *project useKriging↩ WIntermediates:* set true if intermediates target skin and bones are used for transformation.

25.2.2.2 PIPERCOMMON_EXPORT void piper::applyTargetScalingParameter (piper::Project & *model,* piper::hbm::TargetList const & *target*)

This function defines scaling parameter value defined in *target* to *model*.

25.2.2.3 void piper::historyListChanged ( )

Call it after you change project history.

25.2.2.4 PIPERCOMMON_EXPORT void piper::importSimplifiedScalableModel (piper::Project & *project,* std::string const & *filepath*)

This function import Simplified Scalable Model from *filepath* in the *project*.

25.2.2.5 void piper::logDebug (std::string const & *message*)

25.2.2.6 void piper::logDone (std::string const & *message* = " ")

25.2.2.7 void piper::logError (std::string const & *message*)

25.2.2.8 void piper::logInfo (std::string const & *message*)

25.2.2.9 void piper::logStart (std::string const & *message*)

25.2.2.10 void piper::logSuccess (std::string const & *message*)

25.2.2.11 void piper::logWarning (std::string const & *message*)

25.2.2.12 void piper::metadataChanged ( )

Call it after you change some metadata.

25.2.2.13 void piper::modelUpdated ( )

Call it after you change any nodes coordinates.

25.2.2.14 PIPERCOMMON_EXPORT void piper::physPosiDeform (piper::Project & *model,* piper::hbm::TargetList const & *target*)

This function applies the physics based positioning deformation to *model* using *target*.

25.2.2.15 PIPERCOMMON_EXPORT void piper::readContourCL (piper::hbm::FEModel & *c_fem,* piper::hbm::Metadata & *c_meta*)

This function defines reads the contourCL.xml file and populates the contourCL datastructs.

25.2.2.16 void piper::targetChanged ( )

Call it after you change some target.

25.2.2.17 std::string piper::tempDirectoryPath (std::string const & *module* = " ")

Get the application specific temporary path.

The *module* parameter can create a specific sub directory in that path.

Warning

This temporary folder is erased at application exit.

25.3 Python package piper.hbm

The piper.hbm python package gives access to *Human Body Model* related data structures and functions.

Classes

- class piper::hbm::HumanBodyModel

  This class stores a Human Body Model.
- class piper::hbm::AnthropoMetadata< UnitType >

  Base class for Anthropometry metadata.
- class piper::hbm::AnthropoMetadataHeight

  Anthropometry height metadata.
- class piper::hbm::Node

  This class stores a FE node.
- class piper::hbm::IdKey

  This class stores original FE information.
- class piper::hbm::HistoryManager

  This class gives access to the history of a hbm::HumanBodyModel.
- class piper::hbm::FEModel

  This class stores a Finite Element Model.
- class piper::hbm::Metadata

  This class stores the metadata It is composed by different types of metadata.
- class piper::hbm::TargetList

  This class stores a list of targets It is composed by different types of targets.

Enumerations

- enum piper::units::Length { piper::units::Length::m, piper::units::Length::dm, piper::units::Length::cm, piper::units::Length::mm }

  Length units definition.
- enum piper::hbm::ElementType {
  piper::hbm::ELT_HEXA, piper::hbm::ELT_TETRA, piper::hbm::ELT_PENTA, piper::hbm::ELT_PYRA,
  piper::hbm::ELT_QUAD, piper::hbm::ELT_QUAD_THICK, piper::hbm::ELT_TRI, piper::hbm::ELT_TRI_THICK,
  piper::hbm::ELT_BAR, piper::hbm::ELT_UNDEFINED }

  1D, 2D and 3D elements type definition

Functions

- piper::hbm::AnthropoMetadataHeight AnthropoMetadata piper::hbm::getOriginId () const

  Anthropometry weight metadata.

Variables

- const piper::hbm::Id piper::hbm::ID_UNDEFINED

  Special value for undefined ID.
- class piper::hbm::Node & piper::hbm::getOriginId
- class piper::hbm::TargetList piper::hbm::weight

25.3.1 Detailed Description

The piper.hbm python package gives access to *Human Body Model* related data structures and functions.

25.3.2 Enumeration Type Documentation

25.3.2.1 enum piper::hbm::ElementType

1D, 2D and 3D elements type definition

Enumerator

ELT_HEXA

ELT_TETRA

ELT_PENTA

ELT_PYRA

ELT_QUAD

ELT_QUAD_THICK

ELT_TRI

ELT_TRI_THICK

ELT_BAR

ELT_UNDEFINED

25.3.2.2 enum piper::units::Length [strong]

Length units definition.

Enumerator

m

dm

cm

mm

25.3.3 Function Documentation

25.3.3.1 piper::hbm::AnthropoMetadataHeight AnthropoMetadata piper::hbm::getOriginId ( ) const

Anthropometry weight metadata.

Anthropometry age metadata This class stores a FE frame

25.3.4 Variable Documentation

25.3.4.1 class piper::hbm::Node& piper::hbm::getOriginId

25.3.4.2 const piper::hbm::Id piper::hbm::ID_UNDEFINED

Special value for undefined ID.

25.3.4.3 class piper::hbm::TargetList piper::hbm::weight

25.4 Anatomy Database

The anatomyDB library contains knowledge on the human body anatomy.

The anatomyDB library contains knowledge on the human body anatomy.

Author

Thomas Lemaire, Christophe Lecomte

Date

2015

25.4.1 How the data is managed

The following picture presents how the data is organized in the library :

1. The MyCF database is query to get the basis of the database (anatomical entities such as bones, group of bones, joints,...). This process is done by the `make_entity_mycf.py` script. The obtain data is stored in the `entity_mycf_doNotEdit.csv` file.
2. A set of editable .csv files contains information that cannot be retrieved from MyCF. This includes synonyms for entity names, anatomical landmarks, anatomical frames,...
3. The `create.sql` script is used at compile time to produce the sqlite database file.

25.4.2 How to query the database

The core of the anatomyDB library is a set of functions which wrap SQL queries. In addition to the standard C++ library, several wrappers for other target languages are provided.

25.4.2.1 In Python

25.4.2.2 In Octave

Octave (http://www.octave.org) is an open source alternative to Matlab, and is compatible with it up to a certain level. AnatomyDB functions can be used in octave as shown below :

```
% Copyright (C) 2017 INRIA
% This file is part of the PIPER Framework.
% Version: 1.0.0
%
% The PIPER Framework is free software: you can redistribute it and/or
% modify it under the terms of the GNU Lesser General Public License as
% published by
% the Free Software Foundation, either version 2.1 of the License, or (at
% your option) any later version.
%
% The PIPER Framework is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU Lesser
% General Public License for more details. You should have received a copy
% of the GNU Lesser General Public License along with the PIPER Framework.
% If not, see <http://www.gnu.org/licenses/>.
%
% Contributors include Thomas Lemaire (INRIA)
%
% This work has received funding from the European Union Seventh Framework
% Program ([FP7/2007-2013]) under grant agreement 605544 [PIPER project]).
%
# copy bin/anatomyDB.sqlite and lib/anatomyDB.oct from the build tree
# start with octave octave_demo.m

anatomyDB
anatomyDB.init(".") # anatomyDB install directory
printf "C1 synonyms:\n"
anatomyDB.getSynonymList("C1")
```

```
printf "Bony parts of pelvis:\n"
anatomyDB.getSubPartOfList("Pelvic_skeleton","Bone")
printf "Right_anterior_superior_iliac_spine in Kepple1997:\n"
anatomyDB.getSynonymFromBibliography("Right_anterior_superior_iliac_spine", "Kepple1997")
printf "Landmarks defined in Kepple1997:\n"
anatomyDB.getSubClassOfFromBibliographyList("Landmark","Kepple1997")

printf "Bone frames defined in ISB:\n"
anatomyDB.getSubClassOfFromBibliographyList("Frame","ISB_BCS")

printf "Fill landmarks coordinates:\n"
ff = anatomyDB.FrameFactory_instance()
printf "Available frames in factory:\n"
ff.registeredFrameList()
ff.landmark().add("Head_center_of_left_humerus", [1,2,3])
printf "Left_scapula_glenohumeral frame:\n"
ff.computeFrame("Left_scapula_glenohumeral_frame")
printf "Left_femur_hip frame cannot be computed (missing landmarks):\n"
ff.computeFrame("Left_femur_hip_frame")
```

output :

```
C1 synonyms:
ans =
{
  [1,1] = Atlas
  [2,1] = C1
}
Bony parts of pelvis:
ans =
{
  [1,1] = Coccyx
  [2,1] = Left_hip_bone
  [3,1] = Right_hip_bone
  [4,1] = Sacrum
}
Right_anterior_superior_iliac_spine in Kepple1997:
ans = RASIS
Landmarks defined in Kepple1997:
ans =
{
  [1,1] = RASIS
  [2,1] = LASIS
  [3,1] = RPSIS
  [4,1] = LPSIS
  [5,1] = RHJCPEL
  [6,1] = LHJCPEL
  [7,1] = RAIIS
  [8,1] = LAIIS
  ...
}
Fill landmarks coordinates:
ff =

{
  FrameFactory, ptr = 0x7f3d62ffbf40
  computeFrame (method)
  instance (static method)
  isFrameRegistered (method)
  landmark (method)
  registeredFrameList (method)
}

Available frames in factory:
ans =
{
  [1,1] = C1_C2_arch
  [2,1] = C1_Skull
  ...
  [25,1] = Left_calcaneus_ankle
  [26,1] = Left_carpal_radiocarpal
  [27,1] = Left_clavicle_acromioclavicular
  [28,1] = Left_clavicle_sternoclavicular
  ...
  [38,1] = Left_ulna_humeroulnar
```

```
  [39,1] = Pelvis_left_hip
  [40,1] = Pelvis_right_hip
  [41,1] = Right_calcaneus_ankle
  [42,1] = Right_carpal_radiocarpal
  [43,1] = Right_clavicle_acromioclavicular
  [44,1] = Right_clavicle_sternoclavicular
  [45,1] = Right_femur_hip
  [46,1] = Right_femur_knee
  [47,1] = Right_humerus_glenohumeral
  [48,1] = Right_humerus_humeroulnar
  [49,1] = Right_radius_radiocarpal
  ...
}
Left_scapula_glenohumeral frame:
ans =

   1
   2
   3
   0
   0
   0
   1

Left_femur_hip frame cannot be computed (missing landmarks):
error: Error while computating frame Left_femur_hip
Unknown landmark: Medial_epicondyle_of_left_femur (Medial_epicondyle_of_left_femur) (SWIG_RuntimeError)
```

26 Namespace Documentation

26.1 anatomydb Namespace Reference

Classes

- class Frame

  Store coordinates of a frame.
- class FrameFactory

  This class computes anatomical frames coordinate from landmark positions.
- class LandmarkCont

  An interface class to store the landmark positions to anatomydb::FrameFactory.

Functions

- void init (std::string const &databaseFile="")

  Initialize the database.

Queries

These functions are the available queries to the anatomical database.

- unsigned int getEntityId (std::string const &name)
- std::string getReferenceName (std::string const &name)
- bool isSynonymOf (std::string const &name1, std::string const &name2)
- std::string getReferenceNameNoThrow (std::string const &name)
- std::string getEntityDescription (std::string const &name)
- bool exists (std::string const &name)
- bool isBone (std::string const &name)
- bool isSkin (std::string const &name)
- bool isLandmark (std::string const &name)
- bool isEntitySubClassOf (std::string const &name, std::string const &className)
- bool isEntityPartOf (std::string const &entity, std::string const &parent, bool recursive=false)
- bool isEntityFromBibliography (std::string const &entity, std::string const &bibliography)
- std::vector< std::string > getSubClassOfList (std::string const &className)
- std::vector< std::string > getParentClassList (std::string const &name)
- std::vector< std::string > getPartOfList (std::string const &name)
- std::vector< std::string > getPartOfList (std::string const &name, std::string const &className)
- std::vector< std::string > getSubPartOfList (std::string const &name, std::string const &className, bool recursive=false)
- std::vector< std::string > getSynonymList (std::string const &name, bool omitReferenceName=false)
- std::string getSynonymFromBibliography (std::string const &name, std::string const &fromBibliography)
- std::vector< std::string > getEntityBibliographyList (std::string const &name)
- std::vector< std::string > getLandmarkBoneList (std::string const &name)
- std::vector< std::string > getSubClassOfFromBibliographyList (std::string const &className, std::string const &bibliography)
- std::vector< std::string > getAnatomicalEntityList ()
- std::vector< std::string > getLandmarkList ()
- std::vector< std::string > getJointList ()

26.2 piper Namespace Reference

Namespaces

- hbm
- units

Classes

- class Context
  Application context.
- class ModuleParameter
  A piper::Project module parameters.
- class Project
  Piper project.

Functions

- std::string tempDirectoryPath (std::string const &module="")
  Get the application specific temporary path.

Logging

These functions send a message to the application log, using the corresponding level.

- void logDebug (std::string const &message)
- void logInfo (std::string const &message)
- void logWarning (std::string const &message)
- void logError (std::string const &message)
- void logSuccess (std::string const &message)
- void logStart (std::string const &message)
- void logDone (std::string const &message="")

Application signals

These functions send a signal to the application to indicate some thing has changed.

You should call the appropriate function if you modify the model or targets, they will make the application (display, internal data,...) update accordingly.

Warning

In batch mode these functions are not available

- void modelUpdated ()
  Call it after you change any nodes coordinates.
- void metadataChanged ()
  Call it after you change some metadata.
- void targetChanged ()
  Call it after you change some target.
- void historyListChanged ()
  Call it after you change project history.

Modules specific functions

- PIPERCOMMON_EXPORT void physPosiDeform (piper::Project &model, piper::hbm::TargetList const &target)
  This function applies the physics based positioning deformation to model using target.
- PIPERCOMMON_EXPORT void applyTargetScalingParameter (piper::Project &model, piper::hbm::↩ TargetList const &target)
  This function defines scaling parameter value defined in target to model.
- PIPERCOMMON_EXPORT void readContourCL (piper::hbm::FEModel &c_fem, piper::hbm::Metadata &c_meta)
  This function defines reads the contourCL.xml file and populates the contourCL datastructs.
- PIPERCOMMON_EXPORT void importSimplifiedScalableModel (piper::Project &project, std::string const &filepath)
  This function import Simplified Scalable Model from filepath in the project.
- PIPERCOMMON_EXPORT void applyScalingTarget (piper::Project &project, piper::hbm::TargetList const &target, bool const &useKrigingWIntermediates=true)
  This function applies anthropometric dimension target and landmark targets in target to model project useKriging↩ WIntermediates: set true if intermediates target skin and bones are used for transformation.

Variables

- std::list< HeightTarget > height

26.2.1 Variable Documentation

26.2.1.1 std::list<HeightTarget> piper::height

26.3 piper::hbm Namespace Reference

Classes

- class AbstractRigidTransformationTarget
- class AbstractTarget
- class AgeTarget
- class AnthropoMetadata

  Base class for Anthropometry metadata.
- class AnthropoMetadataHeight

  Anthropometry height metadata.
- class BaseMetadata
- class BaseMetadataGroup
- class Element
- class Element2D
- class Entity
- class EntityContact
- class EntityJoint
- class EnvironmentModels
- class FEModel

  This class stores a Finite Element Model.
- class FEModelVTK
- class FixedBoneTarget
- class FrameToFrameTarget
- class HeightTarget
- class HistoryManager

  This class gives access to the history of a hbm::HumanBodyModel.
- class HumanBodyModel

  This class stores a Human Body Model.
- class IdKey

  This class stores original FE information.
- class InteractionControlPoint
- class JointTarget
- class Landmark
- class LandmarkTarget
- class Metadata

  This class stores the metadata It is composed by different types of metadata.
- class Node

  This class stores a FE node.
- class TargetList

  This class stores a list of targets It is composed by different types of targets.
- class TargetUnit
- class WeightTarget

Enumerations

- enum ElementType {
  ELT_HEXA, ELT_TETRA, ELT_PENTA, ELT_PYRA,
  ELT_QUAD, ELT_QUAD_THICK, ELT_TRI, ELT_TRI_THICK,
  ELT_BAR, ELT_UNDEFINED }

  1D, 2D and 3D elements type definition

Functions

- piper::hbm::AnthropoMetadataHeight AnthropoMetadata getOriginId () const

  Anthropometry weight metadata.

Variables

- const piper::hbm::Id ID_UNDEFINED

  Special value for undefined ID.
- class piper::hbm::Node & getOriginId
- class piper::hbm::TargetList weight

26.4 piper::units Namespace Reference

Enumerations

- enum Length { Length::m, Length::dm, Length::cm, Length::mm }

  Length units definition.

27 Class Documentation

27.1 piper::hbm::AbstractRigidTransformationTarget Class Reference

Inherits piper::hbm::AbstractTarget, and piper::hbm::TargetUnit< piper::units::Length >.

Inherited by piper::hbm::FrameToFrameTarget, and piper::hbm::JointTarget.

Public Types

- typedef std::map< unsigned int, double > ValueType

Public Member Functions

- ValueType const & value () const
- void setValue (ValueType const &value)

27.1.1 Member Typedef Documentation

27.1.1.1 typedef std::map<unsigned int, double> piper::hbm::AbstractRigidTransformationTarget::ValueType

27.1.2 Member Function Documentation

27.1.2.1 void piper::hbm::AbstractRigidTransformationTarget::setValue (ValueType const & *value*)

27.1.2.2 ValueType const& piper::hbm::AbstractRigidTransformationTarget::value ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.2 piper::hbm::AbstractTarget Class Reference

Inherited by piper::hbm::AbstractRigidTransformationTarget, piper::hbm::AgeTarget, piper::hbm::FixedBoneTarget, piper::hbm::HeightTarget, piper::hbm::LandmarkTarget, and piper::hbm::WeightTarget.

Public Member Functions

- void setName (std::string const &name)
- std::string const & name () const

27.2.1 Member Function Documentation

27.2.1.1 std::string const& piper::hbm::AbstractTarget::name ( ) const

27.2.1.2 void piper::hbm::AbstractTarget::setName (std::string const & *name*)

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.3 piper::hbm::AgeTarget Class Reference

Inherits piper::hbm::AbstractTarget, and piper::hbm::TargetUnit< piper::units::Age >.

Public Member Functions

- AgeTarget ()
- double value () const
- void setValue (double value)

27.3.1 Constructor & Destructor Documentation

27.3.1.1 piper::hbm::AgeTarget::AgeTarget ( )

27.3.2 Member Function Documentation

27.3.2.1 void piper::hbm::AgeTarget::setValue (double *value*)

27.3.2.2 double piper::hbm::AgeTarget::value ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.4 piper::hbm::AnthropoMetadata< UnitType > Class Template Reference

Base class for Anthropometry metadata.

Inherits piper::hbm::BaseMetadata.

Public Member Functions

- AnthropoMetadata ()
- bool isDefined () const
- void setValue (double const &value)
- double value () const

27.4.1 Detailed Description

template<typename UnitType>class piper::hbm::AnthropoMetadata< UnitType >

Base class for Anthropometry metadata.

27.4.2 Constructor & Destructor Documentation

27.4.2.1 template<typename UnitType> piper::hbm::AnthropoMetadata< UnitType >::AnthropoMetadata ( )

27.4.3 Member Function Documentation

27.4.3.1 template<typename UnitType> bool piper::hbm::AnthropoMetadata< UnitType >::isDefined ( ) const

27.4.3.2 template<typename UnitType> void piper::hbm::AnthropoMetadata< UnitType >::setValue (double const & *value*)

27.4.3.3 template<typename UnitType> double piper::hbm::AnthropoMetadata< UnitType >::value ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.5 piper::hbm::AnthropoMetadataHeight Class Reference

Anthropometry height metadata.

Inherits piper::hbm::AnthropoMetadata< piper::units::Length >.

Additional Inherited Members

27.5.1 Detailed Description

Anthropometry height metadata.

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.6 piper::hbm::BaseMetadata Class Reference

Inherited by piper::hbm::AnthropoMetadata< piper::units::Length >, piper::hbm::AnthropoMetadata< UnitType >, piper::hbm::BaseMetadataGroup, and piper::hbm::EntityJoint.

Public Member Functions

- std::string const & name () const

27.6.1 Member Function Documentation

27.6.1.1 std::string const& piper::hbm::BaseMetadata::name ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.7 piper::hbm::BaseMetadataGroup Class Reference

Inherits piper::hbm::BaseMetadata.

Inherited by piper::hbm::Entity, and piper::hbm::Landmark.

Additional Inherited Members

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.8 piper::Context Class Reference

Application context.

Public Member Functions

- piper::Project & project ()

Static Public Member Functions

- static piper::Context & instance ()

27.8.1 Detailed Description

Application context.

In particular it stores the current application project.

Warning

In batch mode, the context cannot be accessed, the piper::Project class should be used directly.

27.8.2 Member Function Documentation

27.8.2.1 static piper::Context& piper::Context::instance ( ) [static]

27.8.2.2 piper::Project& piper::Context::project ( )

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_app.dox

27.9 piper::hbm::Element< T > Class Template Reference

Public Member Functions

- const piper::hbm::ElementType & getType () const
- const piper::hbm::VId & get () const

27.9.1 Member Function Documentation

27.9.1.1 template<typename T> const piper::hbm::VId& piper::hbm::Element< T >::get ( ) const

27.9.1.2 template<typename T> const piper::hbm::ElementType& piper::hbm::Element< T >::getType ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.10 piper::hbm::Element2D Class Reference

Inherits piper::hbm::Element< piper::hbm::Element2D >.

Additional Inherited Members

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.11 piper::hbm::Entity Class Reference

Inherits piper::hbm::BaseMetadataGroup.

Public Member Functions

- std::vector< double > getOneVertex (piper::hbm::FEModel const *fem) const
- piper::hbm::VId const & getEnvelopNodes (piper::hbm::FEModel const &femodel) const
- piper::hbm::VId getEntityNodesIds (const piper::hbm::FEModel &femodel, bool skip1D=false, bool skip2↩D=false, bool skip3D=false) const
- piper::hbm::VId get_groupElement1D ()
- piper::hbm::VId get_groupElement2D ()
- piper::hbm::VId get_groupElement3D ()
- piper::hbm::VId get_groupGroup ()

27.11.1 Member Function Documentation

27.11.1.1 piper::hbm::VId piper::hbm::Entity::get_groupElement1D ( )

27.11.1.2 piper::hbm::VId piper::hbm::Entity::get_groupElement2D ( )

27.11.1.3 piper::hbm::VId piper::hbm::Entity::get_groupElement3D ( )

27.11.1.4 piper::hbm::VId piper::hbm::Entity::get_groupGroup ( )

27.11.1.5 piper::hbm::VId piper::hbm::Entity::getEntityNodesIds (const piper::hbm::FEModel & *femodel*, bool *skip1D* = false, bool *skip2D* = false, bool *skip3D* = false) const

27.11.1.6 piper::hbm::VId const& piper::hbm::Entity::getEnvelopNodes (piper::hbm::FEModel const & *femodel*) const

27.11.1.7 std::vector<double> piper::hbm::Entity::getOneVertex (piper::hbm::FEModel const * *fem*) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.12 piper::hbm::EntityContact Class Reference

Public Member Functions

- std::string const & name () const
- std::string const & typeStr () const
- std::string const & entity1 () const
- std::string const & entity2 () const
- piper::hbm::Id group1 () const
- piper::hbm::Id group2 () const
- bool keepThickness () const
- double thickness () const

27.12.1 Member Function Documentation

27.12.1.1 std::string const& piper::hbm::EntityContact::entity1 ( ) const

27.12.1.2 std::string const& piper::hbm::EntityContact::entity2 ( ) const

27.12.1.3 piper::hbm::Id piper::hbm::EntityContact::group1 ( ) const

27.12.1.4 piper::hbm::Id piper::hbm::EntityContact::group2 ( ) const

27.12.1.5 bool piper::hbm::EntityContact::keepThickness ( ) const

27.12.1.6 std::string const& piper::hbm::EntityContact::name ( ) const

27.12.1.7 double piper::hbm::EntityContact::thickness ( ) const

27.12.1.8 std::string const& piper::hbm::EntityContact::typeStr ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.13 piper::hbm::EntityJoint Class Reference

Inherits piper::hbm::BaseMetadata.

Public Types

- enum ConstrainedDofType { ConstrainedDofType::HARD, ConstrainedDofType::SOFT }

Public Member Functions

- std::vector< bool > getDofVec () const
- std::string const & getEntity1 () const
- std::string const & getEntity2 () const
- piper::hbm::Id const & getEntity1FrameId () const
- piper::hbm::Id const & getEntity2FrameId () const
- ConstrainedDofType getConstrainedDofType () const

27.13.1 Member Enumeration Documentation

27.13.1.1 enum piper::hbm::EntityJoint::ConstrainedDofType [strong]

Enumerator

HARD

SOFT

27.13.2 Member Function Documentation

27.13.2.1 ConstrainedDofType piper::hbm::EntityJoint::getConstrainedDofType ( ) const

27.13.2.2 std::vector<bool> piper::hbm::EntityJoint::getDofVec ( ) const

27.13.2.3 std::string const& piper::hbm::EntityJoint::getEntity1 ( ) const

27.13.2.4 piper::hbm::Id const& piper::hbm::EntityJoint::getEntity1FrameId ( ) const

27.13.2.5 std::string const& piper::hbm::EntityJoint::getEntity2 ( ) const

27.13.2.6 piper::hbm::Id const& piper::hbm::EntityJoint::getEntity2FrameId ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.14 piper::hbm::EnvironmentModels Class Reference

Public Member Functions

- std::vector< std::string > getListNames () const

27.14.1 Member Function Documentation

27.14.1.1 std::vector<std::string> piper::hbm::EnvironmentModels::getListNames ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.15 piper::hbm::FEModel Class Reference

This class stores a Finite Element Model.

Public Member Functions

- std::size_t getNbNode () const
- std::size_t getNbElement1D () const
- std::size_t getNbElement2D () const
- std::size_t getNbElement3D () const
- piper::hbm::Node const & getNode (const Id &id) const
- piper::hbm::Element2D const & getElement2D (const Id &id) const
- piper::hbm::Element3D const & getElement3D (const Id &id) const
- piper::hbm::FEModelVTK * getFEModelVTK ()
- std::vector< double > getFrameOrientationVec (const Id &id) const
- std::vector< double > getFrameOriginVec (const Id &id) const
- std::map< piper::hbm::Id, piper::hbm::IdKey > computePiperIdToIdKeyMapNode () const

27.15.1 Detailed Description

This class stores a Finite Element Model.

It is composed by a set of piper::hbm::Node and Elements.

27.15.2 Member Function Documentation

27.15.2.1 std::map<piper::hbm::Id, piper::hbm::IdKey> piper::hbm::FEModel::computePiperIdToIdKeyMapNode ( ) const

Returns

map to get nodes piper::hbm::IdKey from a node id

27.15.2.2 piper::hbm::Element2D const& piper::hbm::FEModel::getElement2D (const Id & *id*) const

27.15.2.3 piper::hbm::Element3D const& piper::hbm::FEModel::getElement3D (const Id & *id*) const

27.15.2.4 piper::hbm::FEModelVTK* piper::hbm::FEModel::getFEModelVTK ( )

27.15.2.5 std::vector<double> piper::hbm::FEModel::getFrameOrientationVec (const Id & *id*) const

27.15.2.6 std::vector<double> piper::hbm::FEModel::getFrameOriginVec (const Id & *id*) const

27.15.2.7 std::size_t piper::hbm::FEModel::getNbElement1D ( ) const

27.15.2.8 std::size_t piper::hbm::FEModel::getNbElement2D ( ) const

27.15.2.9 std::size_t piper::hbm::FEModel::getNbElement3D ( ) const

27.15.2.10 std::size_t piper::hbm::FEModel::getNbNode ( ) const

27.15.2.11 piper::hbm::Node const& piper::hbm::FEModel::getNode (const Id & *id*) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.16 piper::hbm::FEModelVTK Class Reference

Public Member Functions

- piper::hbm::VId getSelectedElements (int chosenDimension) const
- piper::hbm::VId getSelectedNodes () const

27.16.1 Member Function Documentation

27.16.1.1 piper::hbm::VId piper::hbm::FEModelVTK::getSelectedElements (int *chosenDimension*) const

27.16.1.2 piper::hbm::VId piper::hbm::FEModelVTK::getSelectedNodes ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.17 piper::hbm::FixedBoneTarget Class Reference

Inherits piper::hbm::AbstractTarget.

Public Member Functions

- FixedBoneTarget ()
- FixedBoneTarget (std::string bone)

Public Attributes

- std::string bone

27.17.1 Constructor & Destructor Documentation

27.17.1.1 piper::hbm::FixedBoneTarget::FixedBoneTarget ( )

27.17.1.2 piper::hbm::FixedBoneTarget::FixedBoneTarget (std::string *bone*)

27.17.2 Member Data Documentation

27.17.2.1 std::string piper::hbm::FixedBoneTarget::bone

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.18 anatomydb::Frame Class Reference

Store coordinates of a frame.

27.18.1 Detailed Description

Store coordinates of a frame.

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_anatomydb.dox

27.19 anatomydb::FrameFactory Class Reference

This class computes anatomical frames coordinate from landmark positions.

Public Member Functions

- LandmarkCont & landmark ()
- bool isFrameRegistered (std::string const &name)
- std::vector< std::string > registeredFrameList () const
- anatomydb::Frame computeFrame (std::string const &name)

Static Public Member Functions

- static FrameFactory & instance ()

27.19.1 Detailed Description

This class computes anatomical frames coordinate from landmark positions.

It uses the singleton design pattern, use FrameFactory::instance() to retrieve its unique instance. To use this class you first have to fill the landmarks position.

27.19.2 Member Function Documentation

27.19.2.1 anatomydb::Frame anatomydb::FrameFactory::computeFrame (std::string const & *name*)

27.19.2.2 static FrameFactory& anatomydb::FrameFactory::instance ( ) [static]

27.19.2.3 bool anatomydb::FrameFactory::isFrameRegistered (std::string const & *name*)

27.19.2.4 LandmarkCont& anatomydb::FrameFactory::landmark ( )

Returns

the landmarks container

27.19.2.5 std::vector<std::string> anatomydb::FrameFactory::registeredFrameList ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_anatomydb.dox

27.20 piper::hbm::FrameToFrameTarget Class Reference

Inherits piper::hbm::AbstractRigidTransformationTarget.

Public Attributes

- std::string frameSource
- std::string frameTarget

Additional Inherited Members

27.20.1 Member Data Documentation

27.20.1.1 std::string piper::hbm::FrameToFrameTarget::frameSource

27.20.1.2 std::string piper::hbm::FrameToFrameTarget::frameTarget

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.21 piper::hbm::HeightTarget Class Reference

Inherits piper::hbm::AbstractTarget, and piper::hbm::TargetUnit< piper::units::Length >.

Public Member Functions

- HeightTarget ()
- double value () const
- void setValue (double value)

27.21.1 Constructor & Destructor Documentation

27.21.1.1 piper::hbm::HeightTarget::HeightTarget ( )

27.21.2 Member Function Documentation

27.21.2.1 void piper::hbm::HeightTarget::setValue (double *value*)

27.21.2.2 double piper::hbm::HeightTarget::value ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.22 piper::hbm::HistoryManager Class Reference

This class gives access to the history of a hbm::HumanBodyModel.

Public Member Functions

- std::vector< std::string > const getHistoryList () const
- bool isHistory (std::string const &stringID) const
- std::string const & getCurrent () const
- void setCurrentModel (std::string const &stringID)

27.22.1 Detailed Description

This class gives access to the history of a hbm::HumanBodyModel.

Warning

when changing the currentModel of the application project, do not forget to restore the original model before the script ends.

27.22.2 Member Function Documentation

27.22.2.1 std::string const& piper::hbm::HistoryManager::getCurrent ( ) const

27.22.2.2 std::vector<std::string> const piper::hbm::HistoryManager::getHistoryList ( ) const

27.22.2.3 bool piper::hbm::HistoryManager::isHistory (std::string const & *stringID*) const

27.22.2.4 void piper::hbm::HistoryManager::setCurrentModel (std::string const & *stringID*)

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.23 piper::hbm::HumanBodyModel Class Reference

This class stores a Human Body Model.

Public Member Functions

- piper::hbm::Metadata & metadata ()
- piper::hbm::FEModel & fem ()
- piper::hbm::HistoryManager & history ()
- bool empty () const

27.23.1 Detailed Description

This class stores a Human Body Model.

It is composed by a piper::hbm::FEModel and its piper::hbm::Metadata.

27.23.2 Member Function Documentation

27.23.2.1 bool piper::hbm::HumanBodyModel::empty ( ) const

27.23.2.2 piper::hbm::FEModel& piper::hbm::HumanBodyModel::fem ( )

27.23.2.3 piper::hbm::HistoryManager& piper::hbm::HumanBodyModel::history ( )

27.23.2.4 piper::hbm::Metadata& piper::hbm::HumanBodyModel::metadata ( )

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.24 piper::hbm::IdKey Class Reference

This class stores original FE information.

Public Attributes

- std::string idstr

  original FE code key
- int id

  original FE code id

27.24.1 Detailed Description

This class stores original FE information.

27.24.2 Member Data Documentation

27.24.2.1 int piper::hbm::IdKey::id

original FE code id

27.24.2.2 std::string piper::hbm::IdKey::idstr

original FE code key

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.25 piper::hbm::InteractionControlPoint Class Reference

Public Member Functions

- int getControlPointRole ()

27.25.1 Member Function Documentation

27.25.1.1 int piper::hbm::InteractionControlPoint::getControlPointRole ( )

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.26 piper::hbm::JointTarget Class Reference

Inherits piper::hbm::AbstractRigidTransformationTarget.

Public Member Functions

- JointTarget ()
- JointTarget (std::string joint, ValueType const &value)

Public Attributes

- std::string joint

Additional Inherited Members

27.26.1 Constructor & Destructor Documentation

27.26.1.1 piper::hbm::JointTarget::JointTarget ( )

27.26.1.2 piper::hbm::JointTarget::JointTarget (std::string *joint,* ValueType const & *value*)

27.26.2 Member Data Documentation

27.26.2.1 std::string piper::hbm::JointTarget::joint

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.27 piper::hbm::Landmark Class Reference

Inherits piper::hbm::BaseMetadataGroup.

Public Member Functions

- piper::hbm::Coord position (FEModel const &fem) const
- piper::hbm::VId get_groupNode ()

27.27.1 Member Function Documentation

27.27.1.1 piper::hbm::VId piper::hbm::Landmark::get_groupNode ( )

27.27.1.2 piper::hbm::Coord piper::hbm::Landmark::position (FEModel const & *fem*) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.28 anatomydb::LandmarkCont Class Reference

An interface class to store the landmark positions to anatomydb::FrameFactory.

Public Member Functions

- void add (std::string const &name, anatomydb::Vector3 const &coord)
- anatomydb::Vector3 const & get (std::string const &name) const
- bool contains (std::string const &name) const
- void clear ()
- std::vector< std::string > keys () const

27.28.1 Detailed Description

An interface class to store the landmark positions to anatomydb::FrameFactory.

27.28.2 Member Function Documentation

27.28.2.1 void anatomydb::LandmarkCont::add (std::string const & *name,* anatomydb::Vector3 const & *coord*)

27.28.2.2 void anatomydb::LandmarkCont::clear ( )

27.28.2.3 bool anatomydb::LandmarkCont::contains (std::string const & *name*) const

27.28.2.4 anatomydb::Vector3 const& anatomydb::LandmarkCont::get (std::string const & *name*) const

27.28.2.5 std::vector<std::string> anatomydb::LandmarkCont::keys ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_anatomydb.dox

27.29 piper::hbm::LandmarkTarget Class Reference

Inherits piper::hbm::AbstractTarget, and piper::hbm::TargetUnit< piper::units::Length >.

Public Types

- typedef std::map< unsigned int, double > ValueType

Public Member Functions

- ValueType const & value () const
- void setValue (ValueType const &value)

Public Attributes

- std::string landmark

27.29.1 Member Typedef Documentation

27.29.1.1 typedef std::map<unsigned int, double> piper::hbm::LandmarkTarget::ValueType

27.29.2 Member Function Documentation

27.29.2.1 void piper::hbm::LandmarkTarget::setValue (ValueType const & *value*)

27.29.2.2 ValueType const& piper::hbm::LandmarkTarget::value ( ) const

27.29.3 Member Data Documentation

27.29.3.1 std::string piper::hbm::LandmarkTarget::landmark

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.30 piper::hbm::Metadata Class Reference

This class stores the metadata It is composed by different types of metadata.

Public Types

- typedef std::map< std::string, piper::hbm::Entity > EntityCont
- typedef std::map< std::string, piper::hbm::EntityJoint > JointCont
- typedef std::map< std::string, piper::hbm::EntityContact > ContactCont
- typedef std::map< std::string, piper::hbm::Landmark > LandmarkCont
- typedef std::map< std::string, piper::hbm::InteractionControlPoint > InteractionControlPointCont

Public Member Functions

- piper::units::Length lengthUnit () const
- bool isGravityDefined () const
- piper::hbm::Coord const & gravity () const
- EntityCont const & entities () const
- piper::hbm::Entity const & entity (const std::string &entityname) const
- JointCont const & joints () const
- ContactCont const & contacts () const
- InteractionControlPointCont & interactionControlPoints ()
- LandmarkCont const & landmarks () const

27.30.1 Detailed Description

This class stores the metadata It is composed by different types of metadata.

27.30.2 Member Typedef Documentation

27.30.2.1 typedef std::map<std::string, piper::hbm::EntityContact> piper::hbm::Metadata::ContactCont

27.30.2.2 typedef std::map<std::string, piper::hbm::Entity> piper::hbm::Metadata::EntityCont

27.30.2.3 typedef std::map<std::string, piper::hbm::InteractionControlPoint> piper::hbm::Metadata::↩ InteractionControlPointCont

27.30.2.4 typedef std::map<std::string, piper::hbm::EntityJoint> piper::hbm::Metadata::JointCont

27.30.2.5 typedef std::map<std::string, piper::hbm::Landmark> piper::hbm::Metadata::LandmarkCont

27.30.3 Member Function Documentation

27.30.3.1 ContactCont const& piper::hbm::Metadata::contacts ( ) const

27.30.3.2 EntityCont const& piper::hbm::Metadata::entities ( ) const

27.30.3.3 piper::hbm::Entity const& piper::hbm::Metadata::entity (const std::string & *entityname*) const

27.30.3.4 piper::hbm::Coord const& piper::hbm::Metadata::gravity ( ) const

27.30.3.5 InteractionControlPointCont& piper::hbm::Metadata::interactionControlPoints ( )

27.30.3.6 bool piper::hbm::Metadata::isGravityDefined ( ) const

27.30.3.7 JointCont const& piper::hbm::Metadata::joints ( ) const

27.30.3.8 LandmarkCont const& piper::hbm::Metadata::landmarks ( ) const

27.30.3.9 piper::units::Length piper::hbm::Metadata::lengthUnit ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.31 piper::ModuleParameter Class Reference

A piper::Project module parameters.

Public Member Functions

- bool getBool (std::string const &moduleName, std::string const &parameterName)
- unsigned int getUInt (std::string const &moduleName, std::string const &settingName)
- double getDouble (std::string const &moduleName, std::string const &settingName)
- template<typename T >
  void set (std::string const &moduleName, std::string const &parameterName, T value)

27.31.1 Detailed Description

A piper::Project module parameters.

27.31.2 Member Function Documentation

27.31.2.1 bool piper::ModuleParameter::getBool (std::string const & *moduleName,* std::string const & *parameterName*)

27.31.2.2 double piper::ModuleParameter::getDouble (std::string const & *moduleName,* std::string const & *settingName*)

27.31.2.3 unsigned int piper::ModuleParameter::getUInt (std::string const & *moduleName,* std::string const & *settingName*)

27.31.2.4 template<typename T > void piper::ModuleParameter::set (std::string const & *moduleName,* std::string const & *parameterName,* T *value*)

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_app.dox

27.32 piper::hbm::Node Class Reference

This class stores a FE node.

Public Member Functions

- piper::hbm::Coord const & get () const
- void set (const Coord &def)
- void setCoord (double x, double y, double z)

27.32.1 Detailed Description

This class stores a FE node.

27.32.2 Member Function Documentation

27.32.2.1 piper::hbm::Coord const& piper::hbm::Node::get ( ) const

27.32.2.2 void piper::hbm::Node::set (const Coord & *def*)

27.32.2.3 void piper::hbm::Node::setCoord (double *x,* double *y,* double *z*)

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.33 piper::Project Class Reference

Piper project.

Public Member Functions

- Project ()
- Project (std::string const &projectFile)
- void read (std::string const &projectFile)
- void write (std::string const &projectFile)
- void importModel (std::string const &formatrulesFile, std::string const &modelrulesFile, std::string const &sourceModelFile, piper::units::Length lengthUnit)
- void exportModel (std::string const &directory)

- piper::hbm::HumanBodyModel & model ()
- piper::hbm::EnvironmentModels & environment ()
- piper::hbm::TargetList & target ()
- piper::ModuleParameter & moduleParameter ()

27.33.1 Detailed Description

Piper project.

27.33.2 Constructor & Destructor Documentation

27.33.2.1 piper::Project::Project ( )

27.33.2.2 piper::Project::Project (std::string const & *projectFile*)

27.33.3 Member Function Documentation

27.33.3.1 piper::hbm::EnvironmentModels& piper::Project::environment ( )

27.33.3.2 void piper::Project::exportModel (std::string const & *directory*)

27.33.3.3 void piper::Project::importModel (std::string const & *formatrulesFile,* std::string const & *modelrulesFile,* std::string const & *sourceModelFile,* piper::units::Length *lengthUnit*)

27.33.3.4 piper::hbm::HumanBodyModel& piper::Project::model ( )

27.33.3.5 piper::ModuleParameter& piper::Project::moduleParameter ( )

27.33.3.6 void piper::Project::read (std::string const & *projectFile*)

27.33.3.7 piper::hbm::TargetList& piper::Project::target ( )

27.33.3.8 void piper::Project::write (std::string const & *projectFile*)

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_app.dox

27.34 piper::hbm::TargetList Class Reference

This class stores a list of targets It is composed by different types of targets.

Public Member Functions

- TargetList ()
- void clear ()
- std::size_t size ()
- void add (FixedBoneTarget const &t)
- void add (LandmarkTarget const &t)
- void add (JointTarget const &t)
- void add (FrameToFrameTarget const &t)
- void add (AgeTarget const &t)
- void add (WeightTarget const &t)
- void add (HeightTarget const &t)
- void read (std::string const &filename)
- void write (std::string const &filename) const

Public Attributes

- std::list< FixedBoneTarget > fixedBone
- std::list< LandmarkTarget > landmark
- std::list< JointTarget > joint
- std::list< FrameToFrameTarget > frameToFrame
- std::list< AgeTarget > age

27.34.1 Detailed Description

This class stores a list of targets It is composed by different types of targets.

27.34.2 Constructor & Destructor Documentation

27.34.2.1 piper::hbm::TargetList::TargetList ( )

27.34.3 Member Function Documentation

27.34.3.1 void piper::hbm::TargetList::add (FixedBoneTarget const & *t*)

27.34.3.2 void piper::hbm::TargetList::add (LandmarkTarget const & *t*)

27.34.3.3 void piper::hbm::TargetList::add (JointTarget const & *t*)

27.34.3.4 void piper::hbm::TargetList::add (FrameToFrameTarget const & *t*)

27.34.3.5 void piper::hbm::TargetList::add (AgeTarget const & *t*)

27.34.3.6 void piper::hbm::TargetList::add (WeightTarget const & *t*)

27.34.3.7 void piper::hbm::TargetList::add (HeightTarget const & *t*)

27.34.3.8 void piper::hbm::TargetList::clear ( )

27.34.3.9 void piper::hbm::TargetList::read (std::string const & *filename*)

27.34.3.10 std::size_t piper::hbm::TargetList::size ( )

27.34.3.11 void piper::hbm::TargetList::write (std::string const & *filename*) const

27.34.4 Member Data Documentation

27.34.4.1 std::list<AgeTarget> piper::hbm::TargetList::age

27.34.4.2 std::list<FixedBoneTarget> piper::hbm::TargetList::fixedBone

27.34.4.3 std::list<FrameToFrameTarget> piper::hbm::TargetList::frameToFrame

27.34.4.4 std::list<JointTarget> piper::hbm::TargetList::joint

27.34.4.5 std::list<LandmarkTarget> piper::hbm::TargetList::landmark

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.35 piper::hbm::TargetUnit< T > Class Template Reference

Public Member Functions

- T const & unit () const
- void setUnit (T const &unit)

27.35.1 Member Function Documentation

27.35.1.1 template<typename T> void piper::hbm::TargetUnit< T >::setUnit (T const & *unit*)

27.35.1.2 template<typename T> T const& piper::hbm::TargetUnit< T >::unit ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

27.36 piper::hbm::WeightTarget Class Reference

Inherits piper::hbm::AbstractTarget, and piper::hbm::TargetUnit< piper::units::Mass >.

Public Member Functions

- WeightTarget ()
- double value () const
- void setValue (double value)

27.36.1 Constructor & Destructor Documentation

27.36.1.1 piper::hbm::WeightTarget::WeightTarget ( )

27.36.2 Member Function Documentation

27.36.2.1 void piper::hbm::WeightTarget::setValue (double *value*)

27.36.2.2 double piper::hbm::WeightTarget::value ( ) const

The documentation for this class was generated from the following file:

- D:/PIPER/piper/share/doc/userManual_scripting_piper_hbm.dox

Index