Closed-loop RGB-D SLAM Multi-Contact Control for humanoid robots

Arnaud Tanguy, Pierre Gergondet, Andrew I. Comport, Abderrahmane Kheddar

Closed-loop RGB-D *SLAM* Multi-Contact Control for Humanoid Robots

Arnaud Tanguy[2,1], Pierre Gergondet[3], Andrew I. Comport[1] and Abderrahmane Kheddar[2,3]

***Abstract*— This paper discusses the integration of a state-of-the-art dense 6D simultaneous localisation and mapping (*D6DSLAM*) to close the QP control loop on an humanoid robot in multi-contact motions. Our multi-contact planning defines desired contacts based on 3D (CAD) models, and generates a reaching plan. Registration of the 3D model onto an RGB-D key-frame graph representation of the explored environment allows to use of visual odometry to make real-time adjustments to the reaching plan, leading to improved in robustness from a wide range of perturbations. Extensive results are presented on various complex tasks using the HRP-2Kai humanoid robot including valve and car's steering-wheel grasping, multi-contact stair climbing from approximate initial humanoid poses.**

I. Introduction

The DARPA Robotics Challenge (DRC)[1] highlighted the possibility to use humanoid robots in rescue and intervention operations. Recently Airbus Group has shown interest in deploying humanoid technology to automate part of the assembly operations in aircraft manufacturing lines[2]. Humanoid robots can exploit multi-contact technology for locomotion and manipulation in confined spaces. In these two use-cases or other similar large-scale workspaces such as shipyards or buildings, a large part of the tasks, many of the tools and the environment are known *a priori* or partially.

Two fundamental challenges of prime importance will be considered in this paper; (i) the ability of the robot to localize itself in a changing but nearly structured environment (e.g. using SLAM technology), and (ii) the ability of the robot to have very reliable and precise multi-contact operation capabilities, which requires using perception in closed-loop with the control of the robot.

For dense localization and mapping, *D6DSLAM* technology is one of the most prominent solutions. Substantial research and development efforts have led to high robustness, accuracy and real-time efficiency. Such a mature and industrial grade solution will be readily accessible in the near future. Even so, few working solutions have been exploited in the context of humanoid locomotion. Previously real-time walking and navigation coupled to dense real-time vision was first presented with [1]. More recently real-time footstep-planning abilities coupled to real-time vision have been demonstrated using dense stereo *SLAM* in [2] for the case of uneven terrain walking. To our knowledge no attempts have gone further than walking. Multi-contact control and interaction is a key behavior for humanoids because it allows to: (i) create closed kinematics chains to drive higher forces; (ii) plan for stable postures using additional contact supports by means of hands and/or any other limbs; and (iii) perform complex locomotion in confined spaces.

*This work is supported by grant from RoboHow.Cog, the Comanoid H2020 project and PIXMAP

[1] CNRS-University of Nice Sophia Antipolis, I3S, France

[2] CNRS-University of Montpellier, LIRMM, Interactive Digital Humans, France

[3] CNRS-AIST Joint Robotics Laboratory, UMI3218/RL, Japan

[1] `www.theroboticschallenge.org`

[2] `www.comanoid.eu`

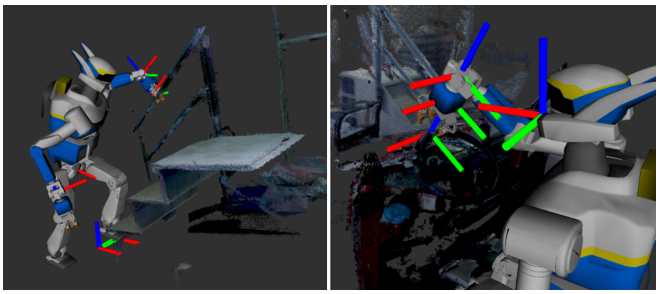

Fig. 1. Reconstructed view of the multi-contact stair climbing and driving wheel grasping trials. The localization of the robot is shown *w.r.t. D6DSLAM*'s map

We propose a full system that integrates the QP-controller detailed in [3][4] with state-of-the-art 6D dense SLAM detailed in [1]. *D6DSLAM* need only rely on cheap, compact, widely available, RGB-D sensors, such as the *Asus Xtion PRO Live* or their outdoor counterparts. The SLAM approach allows to estimate, in real-time, both the environment map and the robot's location within that map. By using such a scheme, it is possible to close the QP-control loop by continuously adjusting the target's pose according to the estimated pose of the robot within its surrounding environment.

We show such possibilities via a set of several representative humanoid walking, climbing and grasping tasks. In summary, the following trials have all been conducted after an extended walking phase starting from arbitrary unknown initial location: (i) Grasping a valve, and (ii) Grasping a car handlebar. The following are performed without walking, but from an approximate initial location: (iii) Grasping a steering wheel, (iv) Executing the first step of a complex multi-contact stair-climbing plan.

Whilst both perception and multi-contact control modules have been previously published, combining them together is a non-trivial task that has lead to the following integration challenges:

- Maintaining real-time alignment of the task goal with the mapped environment
- Robot-centric RGB-D calibration
- Robustness to robot's self-observation (creates outliers in SLAM)
- Closed loop QP-control with dense visual *SLAM* for a set of challenging task

II. System

A. Overview

In [5] a methodology was devised to plan a sequence of contacts that realize complex tasks such as to egress/ingress a

car or climb ladder/stairs. The planning and control are computed in a simulated environment with parametric models allowing for efficient computation. In particular, the planning and control methods require both 3D CAD models of the contact targets as well as a 3D model of the humanoid robot. In [4] such model-based planning and control techniques was tested in a vertical ladder climbing case. One major drawback in such an approach is that the control is performed in open-loop with respect to the real environment. It was necessary to place the robot at a well calibrated position. In order to get rid of this drawback, we propose to close the task-space loop using vision and depth sensors.

Contrary to [6], it is assumed here that 3D parametric models of the target objects are available (car, ladder, stairs, valve...). In this case, contact actions and tasks can be planned using these models (i.e. in simulation) and then used to perform online control by maintaining registration between the 3D CAD model and the 3D map acquired by a real-time SLAM algorithm. By doing so, the planner and controller task goals (desired contacts and transition) are updated and corrected throughout the execution of the task by the SLAM algorithm.

We describe the different components and their integration.

B. Multi-contact planning and control

A detailed explanation of the multi-contact planning and control basics used in this paper can be found in [7][8] and its implementation in a ladder climbing scenario in [4]. In order for the paper to be self-contained, the main components will be described briefly.

Consider the scenario of climbing stairs in Fig. 2. The planner is provided with 3D models of the stairs along with that of the robot. For each 3D model the areas where contact can occur are specified off-line. The planning consists of two modules: a contact explorer and a static posture generation module. The contact explorer suggests potential contacts for addition or removal together with associations between pairs of surfaces (robot, object/environment). These contacts are provided as an input to the posture generator. The latter builds and solves a non-linear optimization problem to determine whether a viable posture is feasible. In this case a statically stable posture (or the robot in contact) is communicated to the planner, which then builds a tree of contact transitions up to the goal.

The contact stances can be smoothened as in [4]. The planned stances of contact with associated robot poses are then given as task objectives using a finite state machine (FSM) that manages task scheduling and uncertainties and builds a multi-contact controller as a quadratic program (QP) that solves the whole body motion locally.

As shown in Fig. 2, contacts are realized respectively in the operational space (by defining a contact pose task $\mathcal{T}_{\text{contact}}$) and a desired pose in the articular space (a least weighted task $\mathcal{T}_{\text{posture}}$ to meet at best). Both can be realized using our task-based quadratic programming (QP) controller [4]. The desired contact and posture pose are formulated as errors that appear among the sum of weighted least-squares terms in the QP cost function. The QP controller includes other tasks $\mathcal{T}_{\text{others}}$, and is solved at each control step. The QP

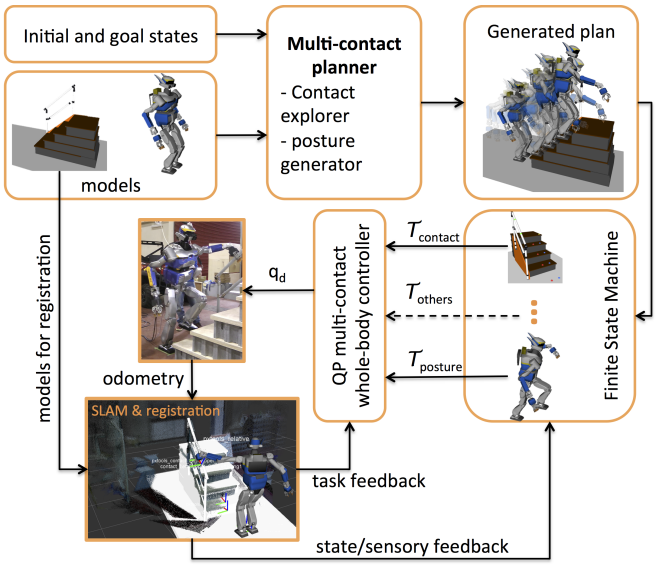

Fig. 2. Overview of the motion generator with the addition of *D6DSLAM* pose estimation closing the loop on the QP-control and the FSM.

variable vector $\mathbf{x} = (\ddot{\mathbf{q}}^T, \lambda^T)^T$, gathers the joint acceleration $\ddot{\mathbf{q}}$, and the linearized friction cones' base weights λ, such that the contact forces $\mathbf{f}$ are equal to $\mathbf{K}_f\lambda$ (with $\mathbf{K}_f$ the discretized friction cone matrix). The desired acceleration $\ddot{\mathbf{q}}$ is integrated twice, $\mathbf{q_d}$ to feed the low level built-in PD control of HRP-2Kai. The driving task with the QP controller writes as follows:

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \sum_{i=1}^{N} w_i \|\mathbf{E}_i(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})\|^2 + w_\lambda \|\boldsymbol{\lambda}\|^2 \\ &\text{subject to} \\ &1)\ \text{dynamic constraints} \\ &2)\ \text{sustained contact positions} \\ &3)\ \text{joint limits} \\ &4)\ \text{non-desired collision avoidance constraints} \\ &5)\ \text{self-collision avoidance constraints,} \end{aligned} \tag{1}$$

where w_i and w_λ are task weights or gains, and $\mathbf{E}_i(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ is the error in the task space. Details on the QP constraints (since they are common to most tasks) can be found in [4]. Contacts are modelized as set-point objective tasks; each contact task (i) is defined by its associated task-error ϵ_i so that $\mathbf{E}_i = K_{p_i}\epsilon_i + K_{v_i}\dot{\epsilon}_i + \ddot{\epsilon}_i$.

We propose to improve contact tasks by altering the FSM actions to take into account a live measurement of the pose of the robot via dense visual odometry. As shown in Figure 2, the *D6DSLAM* method (see Section II-C) is added to the overall architecture where it continuously estimates the robot's pose, and provide clouds for registration with the 3D models all of which is fed back to the QP controller and the finite state machine. The QP controller loop is now effectively closed based on visual odometry estimations. One should note however, that in the case of large discrepancies on-line re-planning might be necessary as the controller might not suffice in recovering such cases.

C. Dense 6D Simultaneous Localization and Mapping

Dense 3D *SLAM* aims at building a map of an environment while simultaneously providing an accurate 6D pose within it. This is achieved by taking advantage of the real-time frame rate of depth-enabled camera sensors, such as the cheap RGB-D sensors or a stereo pairs of cameras, to locally estimate the motion of the sensor for every image frame. For completeness we provide an overview of the multi-keyframe dense SLAM method which we employed – for more details refer to [1]. The basic underlying principle is to estimate a sensor pose $\mathbf{x} \in \mathfrak{se}(3)$ that best explains both the photometric and geometric discrepancies between a pair of RGB-D images: a reference denoted as $(\mathbf{I}^*, \mathbf{P}^*)$ and the current sensor frame $(\mathbf{I}, \mathbf{P})$. Through novel-view synthesis, the current image is iteratively rendered and warps to the reference image location based on the current pose estimate. The pose is estimated by iteratively solving the following non-linear error function for each pixel i, until the discrepancies between the reference and current image are small enough.

$$\mathbf{e}_i(\mathbf{x}) = \begin{bmatrix} \lambda \left(\mathbf{I}_i^*(\mathbf{P}_i^*) - \mathbf{I}_i \left(w(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x}), \overline{\mathbf{P}}_i^*) \right) \right) \\ \mathbf{N}_i^{*\top} \left(\mathbf{P}_i^* - \mathbf{\Pi}\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x})\overline{\mathbf{P}}_i \right) \end{bmatrix} \in \mathbb{R}^2, \tag{2}$$

where the first row of equation (2) is the photometric term relating the reference and current images ($\mathbf{I}^*$ and $\mathbf{I}$) and the second row is a point-to-plane ICP error with projective data association. The surface normals $\mathbf{N}_i^*$ are computed for the reference image from the time integrated set of 3D points $\mathbf{P}^*$. The over-line denotes homogeneous coordinates and $\mathbf{\Pi}$ converts from homogeneous coordinates. The function $w(\cdot)$ is the inverse warping function whose role is to generate a novel-view of the world's geometry from the current image based on the current pose estimate. $\widehat{\mathbf{T}} \in \mathbb{SE}(3)$ represents the last available pose estimate, and $\mathbf{T}(\mathbf{x})$ the current pose increment. λ is the weight which defines the relative uncertainty between depth and image measurements.

This non-linear error is iteratively minimized using a Gauss-Newton approach such that:

$$\mathbf{x} = -(\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T\mathbf{e}, \tag{3}$$

where $\mathbf{J}$ contains the stacked Jacobian matrices of the errors of equation (2), and $\mathbf{e}$ is the stacked error vector.

The pose estimate $\widehat{\mathbf{T}}$ is finally updated using a homogeneous update until convergence as $\widehat{\mathbf{T}} \leftarrow \widehat{\mathbf{T}}\mathbf{T}(\mathbf{x})$.

The basic alignment procedure is extended over large environments using a keyframe-graph to represent extended and more complex environments with occlusions and changing resolutions. In this case pose tracking is performed *w.r.t.* a predicted reference RGD-D frame which is generated by fusing the N closest keyframes in the graph. New frames are added to the graph if the sensor explores new and un-mapped areas. The graph's keyframes are continuously optimized and updated with new information by fusing depth and color data from new images. Robust M-Estimators are used to minimize the impact of noise and outliers (either sensor or environment related) on tracking. This keyframe-graph is a memory-efficient representation suitable for large-scale environment, and it provides efficient ways of doing novel-view synthesis [9], generating alternative representations such as 3D point-cloud and octree maps. It also provides computational efficiency through the use of multi-resolution keyframes. Additionally, the approach is made robust to global illumination changes through high-dynamic range [9] images, accounts for motion blur and rolling-shutter effects [10], and makes it possible to generate super-resolution maps [11].

III. Integration

A. Registration of CAD and D6DSLAM maps

In order to use the planned multi-contact control on a keyframe environment map it is necessary to estimate the transformation between the 3D parametric CAD models used for planning and the observed environment model. The full transformation is a 6D pose which transforms the 3D CAD model onto *D6DSLAM*'s keyframe-graph. In order to achieve this, first points are uniformly sampled from the CAD model using a Poisson-disk sampling method [12]. *D6DSLAM*'s internal keyframe-based representation is also converted into a pointcloud by backprojecting each RGBD-keyframe into a common world reference frame. The resulting pointcloud is then filtered using a voxel grid filter to obtain the desired pointcloud resolution. Converting both 3D models into a common pointcloud representation makes it possible to use widely studied registration methods such as the Iterative Closest Point algorithm [13] to estimate the transformation between them. ICP is a non-linear iterative registration approach (a similar process as that described in II-C for *D6DSLAM*'s tracking) which relies on finding the pose that best minimizes a mean-square distance between two 3D pointclouds. Despite being one of the most widely used methods for registration, the minimization process is only local and requires a good initialization. Since the CAD model only needs to be registered once offline, a rough initial transformation is provided using a 4 point pose estimation procedure [14] and 4 points in correspondence are manually chosen by a user by clicking with a mouse on the image. It should be noted that automatic registration can be performed if *a priori* knowledge is available such as in the case of the DRC driving and egress task. In this case the robot is manually placed in a roughly known seated position onto the driver's seat which provides enough information to initialize the registration procedure.

B. Extrinsic Calibration between sensor and the robot

We need to design a self-calibration procedure that the robot can perform autonomously without requiring the observation of any external calibration pattern.

The robot assumes a set of N random but feasible static postures for which the arm gripper is visible in the camera sensor's field-of-view (see Fig. 3). For each posture, a simple head-scan motion is performed to generate a 3D map of the gripper and associated pair of poses:

1) $\widehat{\mathbf{T}} \in \mathbb{SE}(3)$ the transformation from the best known estimate of the RGB-D sensor pose to the gripper as obtained from the kinematic tree;

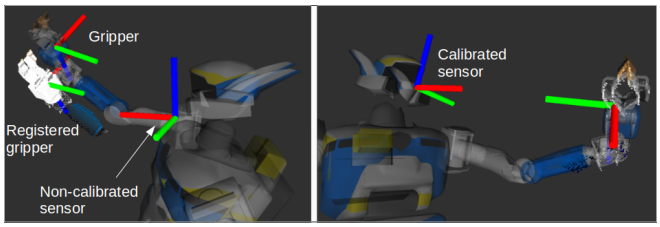

Fig. 3. Calibration performed in simulation, where the RGB-D sensor pose is wrongly defined. The left figure shows the uncalibrated sensor pose, the gripper frame, and a CAD model of the gripper registered onto the pointcloud acquired from *D6DSLAM*. The right part validates the calibration result on a posture that was not part of the calibration set. Notice that the pointcloud is correctly aligned onto the robot's gripper.

Fig. 4. Left: Combined view of *D6DSLAM*'s tracking and HRP-2 showing the successful end of the valve grasping trial. Right: rendering of robot binary mask overlaid on live sensor image.

2) $\mathbf{T}^* \in \mathbb{SE}(3)$ obtained by registering the gripper's CAD model with *D6DSLAM*'s pointcloud, see section III-A.

Using the full set $\mathcal{C} = \left\{(\widehat{\mathbf{T}}_1, \mathbf{T}_1), \cdots, (\widehat{\mathbf{T}}_N, \mathbf{T}_N)\right\}$ of several such poses, the calibration can be formulated as a pose optimization problem.

The pose to be estimated is defined as $\mathbf{T}(\mathbf{x})$, where $x \in \mathfrak{se}(3)$, that minimizes, for each pose pair in $\mathcal{C}$, the following cost function:

$$\mathbf{e}_{\text{cal}} = \mathbf{T}^{*-1}\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x}) - \mathbf{I} \tag{4}$$

This non-linear error is iteratively minimized using a Gauss-Newton approach such that

$$\mathbf{x} = -(\mathbf{J}_{\text{cal}}^{\top}\mathbf{J}_{\text{cal}})^{-1}\mathbf{J}_{\text{cal}}\mathbf{e}_{\text{cal}}, \tag{5}$$

where $\mathbf{J}$ contains the stacked Jacobian matrices of the errors of equation 4. The pose estimate $\widehat{\mathbf{T}}$ is finally updated as in section II-C.

The final pose $\widehat{\mathbf{T}}$ yields the transformation from the previously non-calibrated sensor pose to the calibrated one. Its accuracy depends on several factors: accuracy of the encoders, quality of the registration and variability in the gripper postures. One should note that this isn't a global calibration procedure, and as such, running the calibration process from a clearly wrong posture, as depicted in Fig. 3 may lead to inaccuracies. The process may be repeated several times until the final registration ICP error on validation postures is deemed small enough for the desired accuracy.

C. Self Occlusions

The assumption of a rigid world is made in order to keep geometric consistency throughout tracking. Unfortunately, this assumption is likely to be frequently invalid. Indeed, we do not consider moving objects within the environment itself. This will, to some degree, be handled by *D6DSLAM*'s embedded robustness mechanisms. Moreover, self-observation, *i.e.* the robot's own links being visible within its camera field-of-view, is entirely within our control, and every effort has been made to minimize the effect of self-occlusions. We chose not prevent self-occlusions from happening, but instead prevent them from interfering with *D6DSLAM*'s tracking and mapping. The main idea is to alter the photometric matching algorithm to avoid taking into account areas where the robot is visible. To do so, using the robot's kinematic state and its CAD model, we render a binary mask of the visible parts from the sensor's perspective (see Fig. 4). In order to account for small deviations between the CAD rendering and the actual robot state, we apply a small dilatation to the generated mask.

Let $\mathbf{M} : \Omega \times \mathbb{R}^+ \rightarrow [0,1]; (\mathbf{p}) \mapsto \mathbf{M}(\mathbf{p}, t)$ represent the binary mask corresponding to the measurement acquired at time t. 0-values correspond to areas where part of the robot is visible within the measured frame.

D6DSLAM's tracking equation 2 is modified to incorporate this mask in the following way

$$\mathbf{e}_m(\mathbf{x}) = \mathbf{M}\left(w(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x}), \mathbf{P}^*)\right)\mathbf{e}(\mathbf{x}) \tag{6}$$

In effect, we are considering the error as an outlier at each pixel where the robot is visible.

While this masking scheme effectively improves the robustness, potentially large portions of the input sensor data is effectively ignored, which can lead to tracking inaccuracies, or even in extreme cases loss of tracking.

In addition to dealing with self-occlusion, some steps, not considered here ought to be taken in order to update *D6DSLAM*'s map according to the robots predictable actions within the environment, i.e. if the task is to move an object, then the corresponding object in the map ought to be updated.

D. Control scheme

During the offline planning phase, a set of contact surfaces $\mathcal{C}^* = \{\mathbf{C}_1^*, \cdots, \mathbf{C}_n^*\}$ are defined with respect to a reference CAD model $\mathcal{M}^*$ of the desired task (valve, printer, ladder...). Using CAD model allows to guarantee planned movements within the planning environment. Given an initial reference pose $\mathcal{R}^*$ *w.r.t.* $\mathcal{M}^*$, end-effector trajectories are generated to reach given contact surfaces. At any given moment, the pose $\mathbf{E}_{\mathcal{R}}$ of an end-effector is known *w.r.t.* the base of the kinematic tree. We denote this trajectory by the function $\mathcal{T}^*(\mathbf{E}_{\mathcal{R}}(t), \mathbf{C})$.

Thanks to the calibration parameters of III-B and the estimated 6D-pose $\mathbf{T}_{\mathcal{P}}(\mathbf{t})$ obtained in real-time *w.r.t.* a 3D pointcloud map $\mathcal{P}$, the whole robot kinematic tree is transformed into the world coordinate frame. The pose of any end effector can be expressed *w.r.t.* to $\mathcal{P}$ as

$$\mathbf{E}_{\mathcal{P}}(\mathbf{t}) = \mathbf{T}_{\mathcal{P}}(\mathbf{t})\mathbf{K_s}^{-1}(t)\mathbf{E}_{\mathcal{R}}(t), \tag{7}$$

where $\mathbf{K_s}$ is the pose of the calibrated RGB-D sensor *w.r.t.* the base of the kinematic tree

Registration allows us to express the CAD model $\mathcal{M}^*$ and all its surfaces $\mathcal{C}^*$ *w.r.t.* the environment $\mathcal{P}$

$$\mathbf{M} = \mathbf{R}_{\mathcal{P}}\mathbf{M}^*\mathbf{C}_{\mathcal{P}} = \mathbf{R}_{\mathcal{P}}\mathbf{C}^*, \tag{8}$$

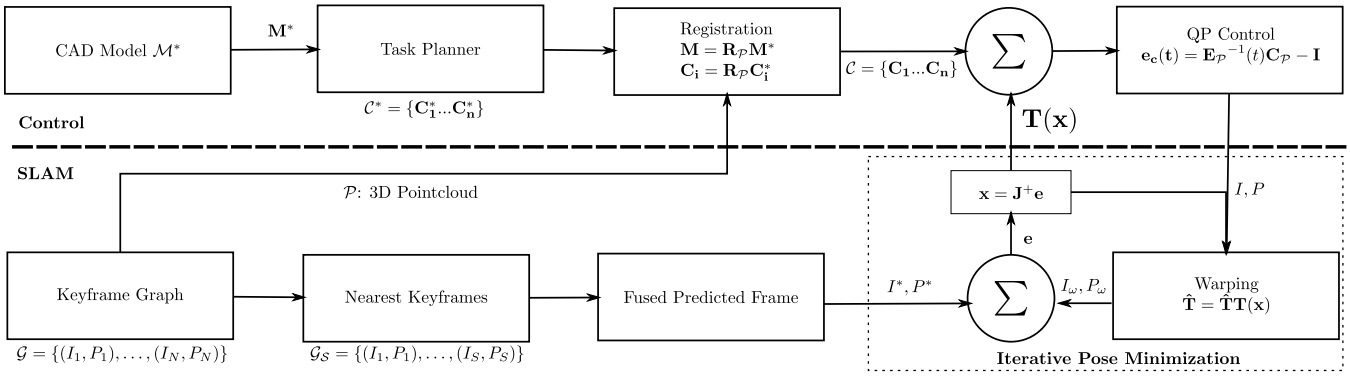

Fig. 5. Closed loop control system. Contact poses $\mathbf{C}^*$ defined during the planning phase *w.r.t.* a CAD model frame $\mathbf{M}^*$ are related to *D6DSLAM*'s map by registration yielding world space contact poses $\mathbf{C}$. HRP's position in world-frame is tracked by *D6DSLAM*'s odometry yielding a pose $\mathbf{T}$. The QP control is tasked with moving an end effector according to the pose error $\mathbf{e_c}$ defined as the error between the contact pose C and the end effector pose obtained as $\mathbf{E}_\mathcal{P}(t)$ by transforming the robot kinematic tree according to the estimated pose $\mathbf{T}$.

where $\mathbf{R}_\mathcal{P}$ is the pose of the registered object in the point-cloud.

This allows us to express the pose error between the current position of an end effector and it's target surface within the environment as:

$$\mathbf{e_c(t)} = \mathbf{E}_\mathcal{P}{}^{-1}(t)\mathbf{C}_\mathcal{P} - \mathbf{I} \tag{9}$$

The effective end-effector trajectory $\mathcal{T}(\mathbf{E}_\mathcal{P}(\mathbf{t}), \mathbf{C}_\mathcal{P})$ is continuously updated according to the relative pose between the end-effector and its target surface, so that at any given moment the controller seeks to minimize the error of Eq. 9. It can be clearly seen in Eq. 7 that the end effector position not only depends on the current absolute encoder readings, but also on the estimated odometry. This means that the QP control will behave as if it was controlled to track a moving target, where in fact, the target is fixed and the end-effector position is recomputed according to odometry.

IV. Experiments and Results

The following experiments aim to demonstrate the ability of the autonomous system to reach the desired contacts, and execute multi-contact plans within an environment where no a-priori map, nor well-calibrated initial pose for the robot is available. All trials are carried out on an HRP-2Kai (used for the DRC), using a low-cost Asus Xtion Pro Live RGB-D sensor and *PIXMAP*'s *D6DSLAM* system [1].

A. Walking phase

For the walking phase two fixed walking targets are defined *w.r.t.* the registered CAD model $\mathcal{M}$. The former is a waypoint used to manually ensure a collision-free path to the later, a final target placed at the expected starting position used during the offline multi-contact planning phase. The posture generator and walking controller from [15] are used to effectively execute the walk. Although we do not adjust the walking plan *w.r.t.* visual odometry, *D6DSLAM*'s tracking and mapping is left active so as to demonstrate its ability to provide sufficiently robust localization to achieve the following tasks.

B. Valve

Inspired by the valve task of the DRC, we aim to, after an extended walking phase from an uncalibrated initial location, establish a gripper-valve contact that would allow us to manipulate it.

1) Setup: HRP-2Kai is placed in an initial configuration ensuring that the valve is visible within the environment to be mapped. That is, facing in the direction of the valve from roughly 4 meters away. The walking targets of Section IV-A, along with a grasp target are attached to the valve CAD model.

2) Walk: As no a-priori map is available, not enough information is yet available for a precise registration. Thus, a rough manual initial registration is performed, which provides walking targets (waypoint and destination). As the robot walks, *D6DSLAM* maps and tracks the robot's motion. Rolling-shutter and motion blur estimation [10] are used to minimize the impact of the high velocity and jerkiness of the walking motion. Still, we observe some inaccuracies in the pointcloud, such as the ones visible in Fig. 6, most notably on the rightmost part, where parts of the pointcloud are mapped in double. This is due to inaccuracies in the pose estimation of some keyframes added during the fast walking motion. This problem could be lessened by first scanning the environment from a stable posture, and walking within the scanned area without mapping, to prevent the creation of inaccurate keyframes. Instead, we opted to perform a head-scan of the target (without resetting the map) from the final walking target, to provide a locally accurate map followed by a second more accurate registration using the method described in III-A.

3) Grasp: A simple control strategy is adopted: without a-priori planning, the QP-controller is tasked with moving the gripper end-effector to the valve target. In order to avoid collisions with the valve, a waypoint in both position and orientation is added directly in front of the desired contact. Throughout the whole motion, the relative pose between the end-effector and its target is updated according to *D6DSLAM*'s pose, as described in III-D. For additional robustness, a guarded approach is used whereby the motion is stopped once a force threshold is attained on the gripper. Our

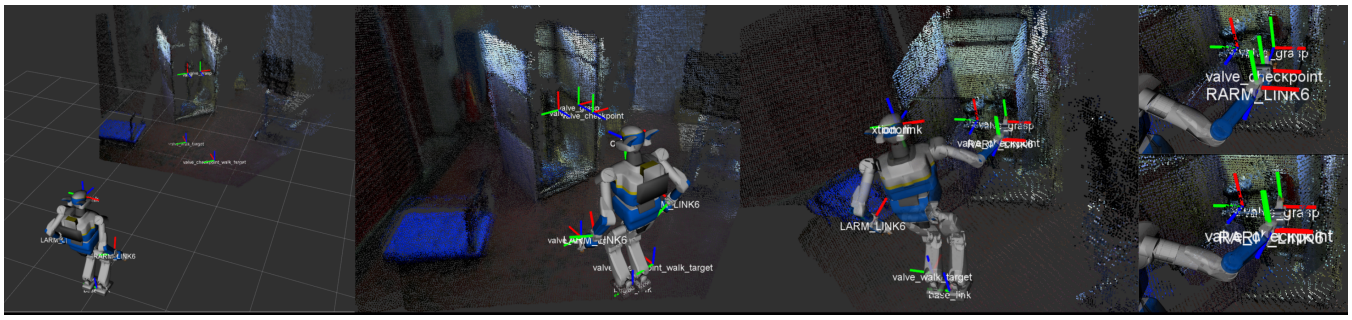

Fig. 6. HRP-2Kai localized within *D6DSLAM*'s 3D reconstructed map at various stages of the valve-grasping trial. From left to right: initial position, position after first walk, position after second walk and beginning of grasping motion. In the last two images, details of the grasping can be seen. The top one shows the end-effector aligned with a checkpoint, the second shows the gripper on the valve, and the checkpoint fully reached.

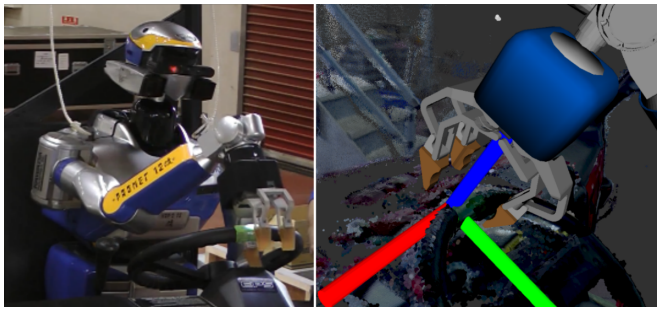

Fig. 7. Combined view of the wheel grasping experiment.

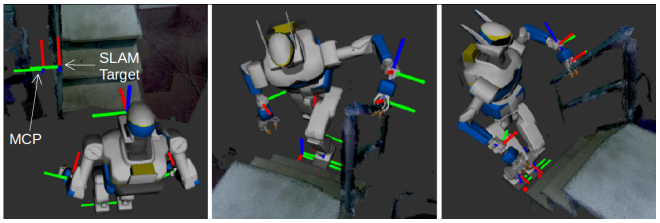

Fig. 8. Combined view of the stairs climbing experiment. The left figure shows the gripper's grasping pose (leftmost frame) that would have been tasked by the MCP without registration, and the corrected registered grasping frame onto the handrail. The others show HRP-2Kai climbing the first step.

experiment shows that, even after walking with live-mapping and no a-priori map, successful grasp may be achieved.

C. Steering Wheel

HRP-2Kai is manually placed into the driver's seat before being actuated. As such the initial pose of the robot cannot be fully controlled, and trying to blindly grasp the wheel would be impossible.

First, a head-scan is performed to acquire a map of the car's dashboard, including the target steering wheel. Even though the initial posture of the robot cannot be fully determined, good convergence of the ICP registration may be achieved without manual initialization, by assuming a plausible initial position of the steering wheel with respect to the robot. We apply the same control strategy as for the valve: a waypoint is defined to avoid collision with the dashboard and the steering wheel, and the target is continuously updated. One should note that due to the narrow space between the robot and the wheel, self-observation is unavoidable. The masking scheme presented in III-C prevents this from affecting the tracking and mapping process.

D. Stairs climbing

While the previous experiments are simple enough not to use the multi-contact planning approach described in II-B, stairs climbing uses it fully. We demonstrate our ability to robustely reach planned contacts from an unknown initial configuration, and under heavy perturbations.

1) Setup: HRP-2Kai is placed in an initial configuration q_i close to the one q_s given as input to the MCP. It is placed so that the QP controller is in a configuration where all its tasks are feasible. The stairs are mapped using a simple head-scan, and their planning model is registered, along with all planned contacts.

2) Closed-loop: The QP is tasked with reaching the first handrail contact of a predefined stair-climbing sequence. Figure 9 starts with control *w.r.t.* to the MCP only and transitions to a closed-loop control strategy. It can clearly be seen that the contact position is modified when transitionning to closed-loop control. This is the result of registration, that accounts for the difference of initial configuration between q_i and q_s. This can be visually observed in Figure 8 where both the planned contact and registered contact are shown. We introduce considerable perturbations by pulling the robot backwards on its ankle flexibilities. The contact trajectory is continuously updated based on *D6DSLAM* odometry as described in III-D, and consequently, the desired contact is reached, as shown in the video-attachement provided with the paper.

V. Conclusions and Future Works

We presented a complete system to close the loop on the QP-control in multi-contact scenarios, whereby the pose of the robot within the world is estimated using visual odometry, and its planned motion is adjusted to guarantee accurate positioning of its end-effectors. In addition, we demonstrate the recent robustness advances of *SLAM* approaches, and propose a strategy to handle the problems caused by having the robot limbs in motion in front of the RGB-D sensor.

Future work will focus on applying the closed-loop method to full multi-contact plans, and providing a detailed analysis of its influence on contact-reaching robustness.

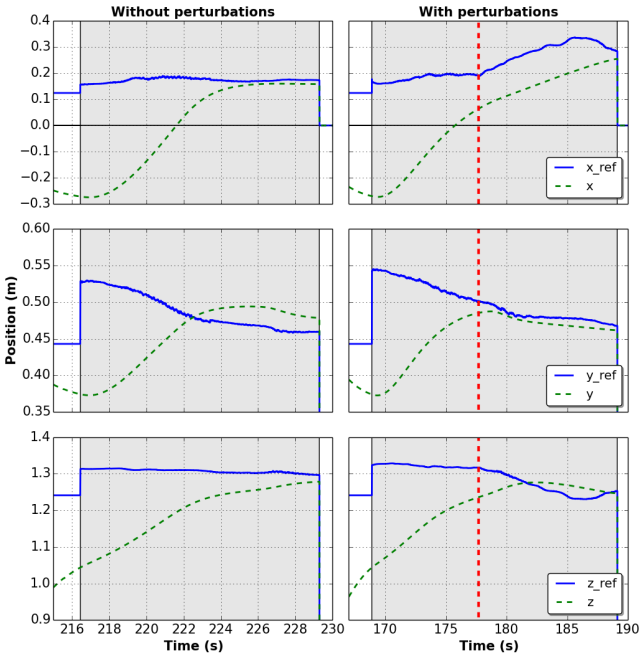

Fig. 9. Closed-loop control of the gripper end-effector tasked with grasping the first handrail contact of a multi-contact stair climbing plan. The white area represents control *w.r.t.* the planning reference, while the gray area corresponds to closed-loop SLAM control. Two independent experiments are considered, on the left contact is reached without additional perturbations, while on the right the robot is manually pulled back at the vertical line. The expected contact position $(x_{ref}, y_{ref}, z_{ref})$ and controlled gripper position (x, y, z) are reported.

Further improvements in both calibration and robustness of *SLAM* approaches in challenging environments need also be considered.

Acknowledgments

The authors would like to thank PIXMAP (www.pixmap3d.com). This work was partially supported by the EU H2020 COMANOID www.comanoid.eu

References

[1] M. Meilland and A. I. Comport, "On unifying key-frame and voxel-based dense visual SLAM at large scales," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Tokyo, Japan, 3-7 November 2013, pp. 3677–3683.

[2] M. F. Fallon, P. Marion, R. Deits, T. Whelan, M. Antone, J. McDonald, and R. Tedrake, "Continuous humanoid locomotion over uneven terrain using stereo fusion," in *IEEE-RAS International Conference on Humanoid Robots*, Seoul, Korea, 3-5 November 2015, pp. 881–888.

[3] K. Bouyarmane and A. Kheddar, "Using a multi-objective controller to synthesize simulated humanoid robot motion with changing contact configurations," in *IEEE International Conference on Intelligent Robots and Systems*, 2011, pp. 4414–4419.

[4] J. Vaillant, A. Kheddar, H. Audren, F. Keith, S. Brossette, A. Escande, K. Bouyarmane, K. Kaneko, M. Morisawa, P. Gergondet, E. Yoshida, S. Kajita, and F. Kanehiro, "Multi-contact vertical ladder climbing with an HRP-2 humanoid," *Autonomous Robots*, vol. 40, no. 3, pp. 561–580, 2016.

[5] K. Bouyarmane, J. Vaillant, F. Keith, and A. Kheddar, "Exploring humanoid robots locomotion capabilities in virtual disaster response scenarios," in *IEEE-RAS International Conference on Humanoid Robots*, Osaka, Japan, 29 November - 1 December 2012, pp. 337–342.

[6] S. Brossette, J. Vaillant, F. Keith, A. Escande, and A. Kheddar, "Point-cloud multi-contact planning for humanoids: Preliminary results," in *IEEE Conference on Robotics Automation and Mechatronics*, Manila, Philippines, 12-15 November 2013, pp. 19–24.

[7] K. Bouyarmane and A. Kheddar, "Humanoid robot locomotion and manipulation step planning," *Advanced Robotics*, no. 26, pp. 1099–1126, July/September 2012.

[8] A. Escande, A. Kheddar, and S. Miossec, "Planning contact points for humanoid robots," *Robotics and Autonomous Systems*, vol. 61, no. 5, pp. 428–442, 2013.

[9] M. Meilland, C. Barat, and A. Comport, "3d high dynamic range dense visual slam and its application to real-time object re-lighting," in *IEEE International Symposium on Mixed and Augmented Reality*, 2013, pp. 143–152.

[10] M. Meilland, T. Drummond, and A. I. Comport, "A unified rolling shutter and motion blur model for 3D visual registration," *IEEE International Conference on Computer Vision*, pp. 2016–2023, 2013.

[11] M. Meilland and A. I. Comport, "Super-resolution 3d tracking and mapping," in *IEEE International Conference on Robotics and Automation*, 6-10 May 2013, pp. 5717–5723.

[12] M. Corsini, P. Cignoni, and R. Scopigno, "Efficient and flexible sampling with blue noise properties of triangular meshes," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 6, pp. 914–924, 2012.

[13] P. Besl and N. McKay, "A method for registration of 3-d shapes," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, 1992, pp. 239–256.

[14] D. F. Dementhon and L. S. Davis, "Model-based object pose in 25 lines of code," *International journal of computer vision*, vol. 15, no. 1-2, pp. 123–141, 1995.

[15] S. Kajita, M. Morisawa, K. Harada, K. Kaneko, F. Kanehiro, K. Fujiwara, and H. Hirukawa, "Biped walking pattern generator allowing auxiliary zmp control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 2993–2999.