Interactive Locomotion Animation using Path Planning

David Flavigné, Michel Taïx

Interactive Locomotion Animation using Path Planning

David Flavigné
ISIR
UPMC Univ Paris 06,
UMR 7222, ISIR, F-75005, Paris, France
Email: flavigne@isir.upmc.fr

Michel Taïx
CNRS; LAAS ; 7, av. du Colonel Roche,
31077 Toulouse, France, Université de Toulouse;
UPS, INSA, INP, ISAE ; LAAS ;
F-31077 Toulouse, France
Email: taix@laas.fr

Abstract

This paper presents a method aimed at enhancing interaction between a path planning algorithm and a user to guide a robot motion task in a virtual environment. Existing works use a two-step decomposition which limits the interaction between the user and the ongoing process. We propose a modification of a classic motion planning method, the Rapidly-exploring Random Tree to build a Interactive-RRT. This method is based on exchanging forces between the algorithm and the user, and on data gathering (labels) from the virtual scene. The Interactive-RRT is combined with a locomotion controller and then applied to a virtual character for generating a locomotion movement along a path suggested by the user.

1 Introduction

This work concerns the realization of a new approach to solve motion problems in numeric models (digital mockup) based on interaction between a user and a motion planning algorithm. The problem of moving an object inside the digital mockup can be solved generally by two approaches : interaction device or automatic path planning.

The classical approach uses an interaction device (3D mouse or a haptic arm) in order to interact with CAD software. Haptics is a recent enhancement that provides an additional perceptual modality in virtual environments [5]. A force feedback device allows the user to test collision-free paths in assembly tasks during the process of industrial design or benchmark maintenance. The contribution of haptics can improve the use of Virtual Reality [28, 6]. The use of a 3D mouse for interaction, is less expensive than a haptic device and can give very good results in many design processes [7, 24]. The main applications concern the operations of dis-assembly for maintenance checks as well as during final phase assembly in car and aeronautical industries

Application in animation by controlling a virtual character or humanoid using a joystick is a simple an intuitive approach when the user wishes to control the walking direction [8, 9]. But in complex scenes the user may get lost and need help in finding the solution easily.

In robotics, a lot of work in motion planning has been done to compute free paths in digital models for mechanical systems. With the recent results in random planning algorithm [15] it is possible to solve automatically problems for systems with many degrees of freedom. The algorithm computes a collision-free roadmap in a configuration space (*CS*) where the object is reduced to a point. This point represents the robot's model in the environment and the dimension of *CS* is equal to the number of degrees of freedom of the mechanical system. The algorithm searches a free path for a point in the roadmap. Now, companies uses automatic path planning to solve PLM applications [10]. There are mainly two families of methods for building a roadmap, the Probabilistic Roadmap (PRM) [13] and the Rapidly-exploring Random Tree (RRT) [16]. The roadmap in the PRM's case is a graph and in RRT's case a tree. The RRT approach is more interesting to our study because it is faster in the single query case, when the roadmap computation must be limited in time.

These methods are very effective but does not allow interaction with a user who wants to guide movement. Moreover, it suffers from the difficulty in finding narrow passages (see for example [3, 1, 25, 17]). When the algorithm finds the narrow passage entrance in the *CS*, it will quickly progress to find a solution. Random algorithms progress easily in narrow passage but have no global vision of the solution. On the other hand, a user will have much more difficulties to progress in narrow passages.

For a free or low-constrained environment, a user will easily find a solution (for example, we quickly steer the key towards the keyhole). Thus, the user has a good global vision of the problem, but the algorithm will find it difficult to explore and find narrow passage.

It is noticeable that the algorithm and a user with an interaction device, can both work in complementary fashion. The complementarity between user and algorithm is

the essential idea of this work.

The idea of mixing a planning algorithm with user interaction is also interesting in the character animation context. The automatic part can take care of tedious tasks (creating the walking animation for example), while the user takes care of higher level constraints such as determining the path and the upper body movements.

The main contribution of the method is to allow simultaneously cooperation between the loop of roadmap search and the loop of interaction for the user to guide a virtual character.

The paper is organized as follows. We begin by describing previous work in interactive motion planning and using motion planning technique for animation. The general interaction loop between user and motion planning algorithm which is at the basis of our work and the problem statement are summarized in section 3. The following section describes the Interactive locomotion planner that is decomposed into two parts : semi-interactive planner and interactive planner Several experiments are illustrated in the various sections.

2 Previous work

The idea of taking into account user input has been studied by various authors specialized in motion planning. In [2] the authors use randomized techniques to transform a colliding user-input path into a collision-free path.

In [21], authors use harmonic functions, computed over a channel solution generated by a cell decomposition, to generate guiding forces that aid the user in the virtual environment. The idea of [18] is to find a free volume between start and goal using octree decomposition and A^*. The path is planned in this volume using an RRT approach.

In all these studies, it is necessary to note that the interaction between the automatic search by motion planning algorithm and the user is simplified by a decomposition in two stages. In our work we do not separate the two stages.

Among other applications of path planning, virtual character animation was also explored. In [4] authors illustrate in a video game the advantages of using path planning for virtual animation. Some works introduced a multi-level roadmap that represent movements on different ground elevations [22] or for different types of locomotion [23]. But path planning based methods are difficult to customize for a flexible use in character animation. On the other hand, motion capture is widely used to generate character animations [14], but can be tedious.

In [20], the author proposed a method to generate automatically an animation of a walking virtual character. The method is divided in a planning step and an animation step. The planning step is done using the visibility-PRM algorithm. The standard linear local method is replaced by a method that uses bezier curves to generate a more natural looking path. The resulting path is resampled to create intermediate configurations that are equally separated from each other in time. This is done to simulate a speed profile along the path.

The animation step is done by interpolating a limited set of motion capture samples. The samples are recorded with constant linear speed v and angular speed ω. Each sample is identified by its (v,ω) values. Then, for each configuration on the planned path, its (v,ω) values are computed and projected into (v,ω) space containing the motion capture samples. The configuration of the virtual character is interpolated from the nearest library samples in the two dimensional (v,ω) space.

In [27], the author stated that the human locomotion has some holonomic parts. To illustrate that fact he proposed a variant of the locomotion controller including a lateral speed v_l corresponding to the holonomic behaviour of human walk. A new motion capture library was recorded to include a lateral speed in the samples.

While these works greatly ease virtual character animation, they are based on probabilistic algorithms that are not controllable by the user. Usually, a user may want to choose the path among obstacles himself according to constraints that can not easily be taken into account by automatic algorithms. The method proposed in section 4 is based on this observation and on the work in [19] and [26].

3 Interactive Motion Planning System

3.1 Approach overview

The proposed approach aims at improving the operator guidance with the help of an automatic path planning algorithm with an interactive device, I-Device. We consider that the user wants to guide the trajectory of the object or virtual character through I-Devices such as space mouse or haptic arm. If I-Device is an haptic device the method uses force through the haptic feedback, in the case of a 3D-mouse, the motion of the mouse is transformed in a *pseudo-force*. We use the same name to represent the real force with haptic device.

The approach has to take into account two constraints:

- The user's direction of movement has to lead the planner development.
- The planner has to return some useful information to the user (haptic and/or visual information) concerning obstacles proximity and if this area has already been explored.

The general plan is constituted by two main loops which will work in parallel (see figure 1). In one loop, the path planner runs to explore the free configuration space, build a roadmap with RRT path planning method and search an automatic solution . The sampling and extension of the initial RRT method is modified to take into account the user interaction. In the other loop, the user moves the object in the virtual environment in six dimensions (position and orientation) with the I-Device as he wishes. The user motion creates a pseudo-force, F_u, and

the algorithm returns the pseudo-force, F_a, that must be seen as disturbance.

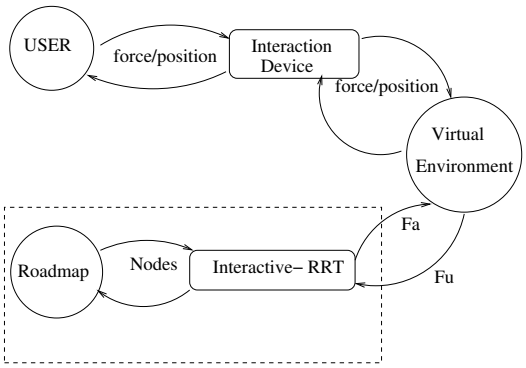


Figure 1. General loop

There are three modes of working:

- The user moves the I-Device more quickly than the development of the planner: the IRRT tracks the direction of the Haptic Device.
- The user does not move the I-Device: the IRRT informs the user of the preferential direction to follow via visual and pseudo-force information.
- Both user and planner are not working at the same time: there is a synchronization between both loops of the main plan, which are working on different frequencies. The difference of frequency between both loop is not a problem because each loop is non-blocking. They're only sharing few data, i.e. pseudo-forces.

3.2 Interactive RRT Algorithm

Our algorithm is based on Rapidly-exploring Random Tree (RRT) approach [16]. Starting at a given initial configuration, RRTs incrementally search the configuration space for a path connecting the initial and the goal configuration. At each iteration a new configuration is sampled (*RANDOM_CONFIG*) and the extension from the nearest node in the tree (*NEAREST_NODE*) toward this sample is attempted. If the extension succeeds (*CONNECT*) a new node in the roadmap is created at a distance ε of the nearest node. This extension process will be called classical extension in this article.

In our case it is necessary to take into account the interaction with the user. The main modifications of the basic RRT are to be made on the following stages:

- By definition, the state of the node is binary. If the node is free and the connection path is also collision free, the node/edge is created in the roadmap, otherwise, any information is memorized. It is interesting to memorise information for the user when the *CONNECT* function return false.
- The sampling area in *RANDOM_CONFIG* is a crucial point to extend efficiently the roadmap. User's interaction must be taken into account in the definition of the sampling area.
- The *NEAREST_NODE* function has not only to take into account the Euclidian distance but also the most interesting nodes with regard to the task of the user.
- At each iteration, the new I-Device position must be taken into account to compute these differents steps

To take into account these various points we propose a new algorithm called Interactive Rapidly-exploring Random Tree, IRRT, which is presented in detail in [12].

The algorithm is shown in Algorithm 1 and the principal steps are:

- Compute an attractive pseudo-force F_a from the current tree using the roadmap data (nodes) to influence the user movement via an I-Device and haptic and/or visual hints.
- Compute an interaction pseudo-force F_u from the user's input (position and/or movement) to take into account the user's intention.
- Compute sampling from a pseudo-force F_r: choose an efficient way to combine user's intentions and algorithm automatic search (from F_a and F_u).
- Choose nearest neighbor in the tree.
- Extend the roadmap and label the nodes.

Algorithm 1 Interactive-RRT

$T(q_{init})$
for $i = 0$ to N **do**
  $q_{rand} \leftarrow$ SAMPLED_CONFIG(F_a,F_u)
  $q_{near} \leftarrow$ NEAREST_NODE(q_{rand}, T)
  $L_{near} \leftarrow$ NEAREST_NEIGHBORS(q_{user}, T)
  if CONNECT(T, q_{rand}, q_{near}, q_{new}) **then**
    Add_Vertex(T, q_{new})
    Add_Edge(T, q_{near}, q_{new})
    Update_Labels(T, q_{near}, q_{new})
    $COM \leftarrow$ Compute_CenterOfMass(L_{near})
    $F_a \leftarrow$ Compute_FAlgo(COM, q_{user});
    $F_u \leftarrow$ Compute_FUser(q_{user});
  end if
end for

3.2.1 Attractive pseudo-force computation

We first retrieve the user's position in the virtual scene and then take some roadmap nodes (the p nearest nodes) near this position (*NEAREST_NEIGHBORS*). After getting their respective labels, we compute the center of mass (*COM*). The attractive pseudo-force (F_a) is given by the vector that goes from the user's position (q_{user}) to the center of mass (*Compute_FAlgo*).

3.2.2 Interaction pseudo-force computation

Then we have to compute another pseudo-force from the user input intended to guide the algorithm development via a specific sampling method. We first compute the user's movement (i.e.: difference between the current position and the previous recorded position) and then convert it into a pseudo-force vector (F_u) in the configuration space ($Compute\_FUser$).

3.2.3 Sampling method

Allowing the user to control the sampling is allowing him to lead the search. With the two previous computed pseudo-forces, we can compute a new one by : $F_r = \alpha.F_u + (1-\alpha).F_a$.

The parameter α allows to tune the part of user's intention compared with the algorithm search. The sampling is realized by shooting along the F_r direction. The aim is to have a sampling area deformed along the pseudo-force direction ($SAMPLED\_CONFIG$).

3.2.4 Nearest neighbor

The nearest neighbor search is done by comparing the Euclidean distance between the sampled configuration and each node of the roadmap ($NEAREST\_NEIGHBORS$). Then we can compute COM using the previously computed labels as weights for each node ($Compute\_CenterOfMass$). We take a fixed number p of neighbors (the nearest ones) for this.

3.2.5 Extension

First we compute a classical extension ($CONNECT$). Then, we analyze the result and classify it in one of these three categories : Collision (extension has failed), Reached (extension succeeds and reaches the last shot sample q_{rand}) and Obstacle (extension reaches the node q_{new} but did not reach the shot sample q_{rand}).

3.2.6 Node Labelling

We compute collision labels for the new node and the extended node. The only information we can use comes from the collision tests. In the interactive algorithm, we used them to label roadmap nodes ($Update\_Labels$). There are two types of information we used in the algorithm: the distance from the node to the goal configuration and the number of collisions that occur from trying to extend the node (towards a shot configuration, when this node was chosen as the nearest neighbor).

3.3 Results

We use a haptic device ($haption - virtuose$) to move the object in CAD model. During all the tests the α parameter is equal to 0.5 for balancing user and algorithm part. The number of nearest neighbors p is fixed to ten.

The algorithm was implemented in C++ in the software platform HPP developed at LAAS based on KineoWorks[1] The experiments were performed on a PC Dual Core, 2.1 Ghz and 2 GB ram.

This example is a real use-case from the automotive industry. The problem is to find a collision free path dismounting a part of a car (a silencer) to check the manufacturing process or the maintainability of the assembly.

We show only a part of the real environment for visualization problem and the environment is limited to represent the reality. This example is very constrained and it is very difficult to disassembly the silencer for a user who does not have good practice on haptic arm. To help the user, we did not forbid the penetration of the virtual object moved by the user into obstacles, because without a real 3D display the user can be stuck in a narrow passage in a complex environment without seeing why he cannot move. It is not a problem because the algorithm develops a roadmap in the free-collision space, the soluton is always without collision.

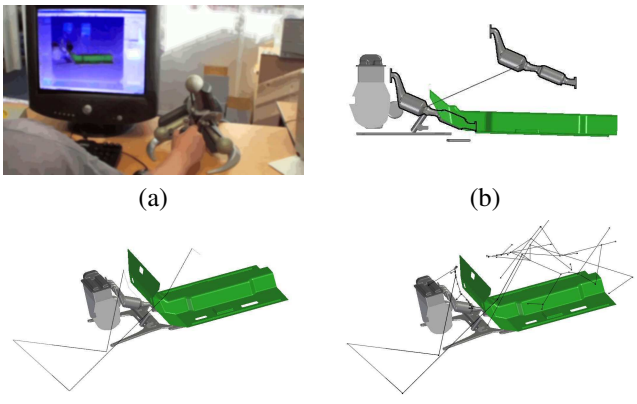


Figure 2. The automotive industrial case.

The input of the algorithm is a CAD model with initial and final configuration for the silencer (figure 2-a). The goal of the user is to take out the silencer by the top.

The example is solved in a few minutes (solution in figure 2-c) and much faster than basic RRT. If the user does not move the haptic device the computation time is the RRT time computation.

The computation time to find the solution is strongly dependent on the user dexterity with the I-Device (analysis is done in [11]).

The roadmap build by the I-RRT with the user's motion is shown in figure 2-d.

This industrial case is a direct application of the motion planning system that allows a user to cooperate with the algorithm to find a feasible path for an object during a disassembly operation. But the system can also be applied to other kind of problem such as character animation, as we shall see in the next section.

[1] KineoWorks is the path planning dedicated Software Development Kit developed by KineoCAM.

4 Interactive Locomotion Planner

Animating a virtual character to imitate human behaviour is a common problem in computer animation. Common solutions are key frames and motion capture. The first one allows users to specify key configurations of the virtual character and then, intermediate positions are interpolated. The drawback is that the user has to specify each key frame manually. The second one allows to copy a human movement to reproduce it on the character, but each movement that we want the character to perform must be recorded.

The main idea of the interactive locomotion planner is to combine the interactive motion planning system presented in section 3 with the work in [27] on a non-holonomic locomotion controller, described in section 2. This controller use a limited number of motion capture samples to generate the animation of a virtual character along any given path.

We decided to separate the planner into two parts to propose a solution to two different problems of character animation : animation with a predefined trajectory, and real-time animation. In the next sections we will describe these two planners and show some examples of their applications.

4.1 The semi-interactive planner

4.1.1 Principle

The aim of the semi-interactive version of the planner is to perform a specific planning query with a known goal configuration. The main steps of the algorithm are shown by algorithm 2.

Algorithm 2 Semi-Interactive Locomotion Planner

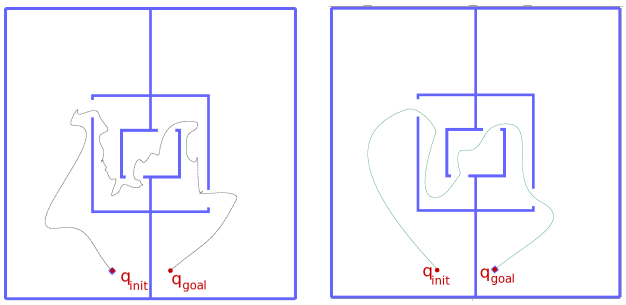
$P_{bezopt} \leftarrow$ IRRT(q_{init},q_{goal})
$T_{root} \leftarrow$ PATH_TO_TRAJ(P_{bezopt})
$T_{anim} \leftarrow$ ANIMATE(T_{root})

It can be divided into three parts:

- A planning step using the interactive motion planning system. The user is guiding a path search between the initial configuration (q_{init}) and the final configuration (q_{goal}). The local method used to link two configurations is modified to make a more human-like path. The path search is done with the bounding box of the virtual character. The bounding box has three degrees-of-freedom (X, Y, θ).

- A transformation step from the path obtained in the previous step (P_{bezopt}) to a trajectory T_{root} containing time data (which was not the case with P_{bezopt}).

- An animation step: for each configuration of T_{root}, the whole-body configuration of the virtual character is computed using the motion capture-based algorithm presented in [27].



(a) A simple local method using Bezier curves (b) The modified local method

Figure 3. A non-interactive planning query

4.1.2 Smoothing

In section 3, we described a system to compute a path interactively for a free-flyer type object. In the present case, the object is not a free-flyer and has non-holonomic constraints. With a simple local method using bezier curves, the path would result as on Fig. 3-a. We implemented a local method similar to the one used in [27], but the method to choose the nearest neighbour was modified to promote human-like walk, unlike the interactive motion planner. This smoothing step can be divided into three parts :

- The weight of rotations in the distance computation was increased.

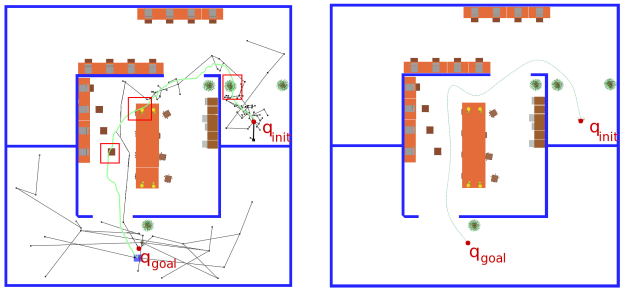
- A coefficient representing the relative positions of the two configurations to link is applied on the distance computation. This coefficient favorizes forward movements more than backward movements.

- The last step uses common path optimization methods (shortcut optimization, etc.) to smooth the path resulting from the two previous steps as they result in a winding path (Fig. 3-b).

4.1.3 Results

In this example (shown on Fig. 4), the user is guiding the trajectory search through a cluttered environment. On Fig. 4-a), we can see the user's path in green. It is approximative (the red frames are showing colliding parts of this path) because the user is not really experienced with the interactive device. The uniform component of the sampling method allows us to avoid the collisions on the user's path. The roadmap build by the interactive algorithm is shown in black. On Fig. 4-b), the computed trajectory is displayed.

Another run (which implies another user path in this case) of the semi-interactive planner in the same environment is shown on Fig. 5-a). In this case, the algorithm chooses another path for avoiding the table in the central room. The Fig. 5-b) shows the animation of the virtual character along the trajectory computed on Fig. 5-a).



(a) In green, the path imposed for the virtual character by the user using the interactive device. In black, the roadmap build during the planning.

(b) The path resulting from transformation using bezier curves.

Figure 4. Guiding character movement

4.2 The interactive planner

4.2.1 Principle

The main goal of the interactive version of the locomotion planner is to animate a walking virtual character while the user is guiding the path. In this version, the goal configuration can be omitted as the position of the user is replacing it. The idea is to animate the character along the path traveled by the user during one iteration of the algorithm main loop. This idea is described in the algorithm 3:

Algorithm 3 Interactive Locomotion Planner

loop
  $P_{linear} \leftarrow$ IRRT(q_{pv})
  $P_{opt} \leftarrow$ BEZIER_OPTIMIZER(P_{linear})
  $T_{root} \leftarrow$ PATH_TO_TRAJ(P_{opt})
  $T_{anim} \leftarrow$ ANIMATE(T_{root})
  $q_{pv} \leftarrow$ PLAY(T_{anim})
end loop

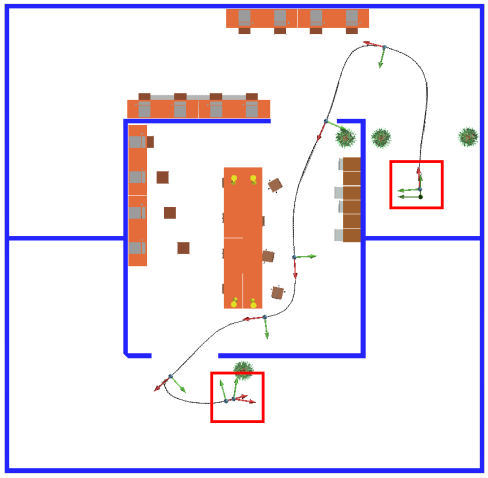
The algorithm can be divided into five steps:

- A planning step
- A transformation step
- A trajectory step
- An animation step
- A playback step

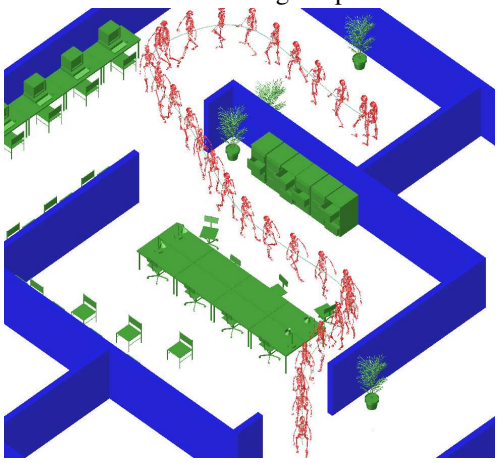
These steps will be detailed in the next section.

4.2.2 Steps

These five steps are computed in parallel with another part of the algorithm which does not appear in the implementation : the user side. During one iteration of the algorithm, the user moves the interactive device to specify the path that the virtual character should follow.



(a) In black, the path obtained with the semi-interactive planner. Axes are indicating the character orientation along the path.

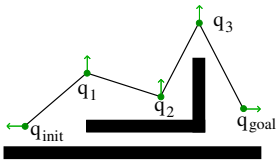


(b) Character animation along the path.

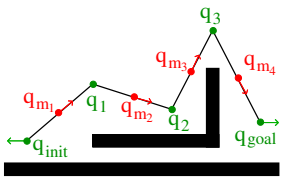
Figure 5. Semi-interactive Locomotion Planner case

1. The first step is the planning step (IRRT(q_{pv})). This step plans the path of the character bounding box between the character start position q_{pv} and the current user's position (specified with the interactive device). For this step, the interactive motion planner described in section 3 was used with some modifications aimed at reduce the computation time. First, the local method was simplified to a simple linear interpolation method. Secondly, the degrees-of-freedom are reduced to 2 : the orientation of the bounding box is not taken into account anymore. The resulting path P_{linear} is collision free.

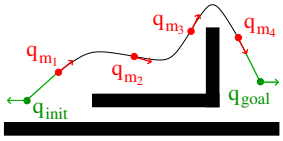
2. The second step is the transformation step. The goal of this step is to get a human-like path P_{opt} from the planned path P_{linear}, using bezier curves (BEZIER_OPTIMIZER(P_{linear})). The different steps of this transformation are detailed on Fig. 6.



(a) The path resulting from the planning step.



(b) Intermediate configurations are added. Orientations of q_{init} and q_{goal} are not changed.



(c) Bezier local paths are created between the intermediate configurations.

Figure 6. Transformation step

3. The third step is the generation of a trajectory with information. The path resulting from the previous step does not have any speed or time information. For animating the character, speed information is computed acoording to speed and acceleration limitations provided by the samples of the motion capture library (PATH_TO_TRAJ(P_{opt})).

4. The fourth step is the animation step (ANIMATE(T_{root})). Using the trajectory computed in the previous step, the actual configurations of the character along the path are generated by interpolating motion captures samples. These samples are chosen in the library depending on the speed along the trajectory.

5. The fifth step is the playback step (PLAY(T_{anim})). The trajectory previously computed is applied to the virtual character. This step is aimed at giving a visual feedback to the user during the computation.

The five main steps of the algorithm are repeated until the user stops the algorithm.

4.2.3 Results

This section show some examples to describe and illustrate the interactive version of the locomotion planner.

On Fig. 7, different steps of the interactive locomotion planner are shown. Fig. (a), (b) and (c) shows the three first iterations of the algorithm. No optimization is visible because the path is to small. The algorithm is tracking the user position. On Fig. (d) and (e), the path is slightly optimized to be more natural looking.

Fig. 8 shows that the collision avoidance properties of the semi-interactive planner are also working with the interactive planner. The user seems to be unexperienced with the device and draws a colliding path. The algorithm manages to avoid the colliding parts and reach the user's position.

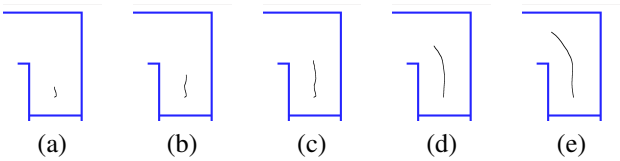
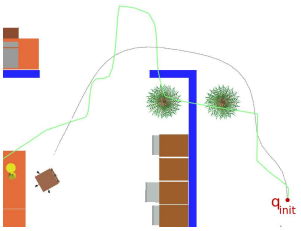
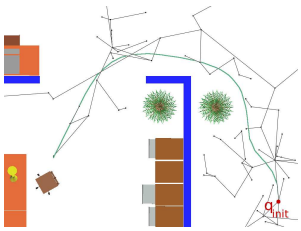


Figure 7. Different steps of the interactive planner, in chronological order



(a) In green, the colliding user path. In black, the path given by the interactive planner tracking the user's position.



(b) In black, we can see the roadmap build by the interactive planner. In green, the final path.

Figure 8. Interactive Locomotion Planner case

One of the problems that occurs with the interactive method is that the collision avoidance is only local and can create blocking situations. For example, when the user is in collision (to ease the guidance by the user, collisions are not forbidden) or is crossing walls, the algorithm may not be able to reach his current position in the virtual scene, and will use the nearest non-colliding position to generate the animation. The user's path may not be followed exactly.

5 Conclusions

In this work, we described a simple interactive method designed for interaction between user and a path planning algorithm. This method is based on pseudo-forces exchange between the algorithm and the user, and on data gathering (labels) from the virtual scene. We saw an illustration on a real industrial case. The method was applied on virtual character animation to compute a locomotion movement. Two cases were covered. The first one, when the goal of the planning query is known, is solved with a semi-interactive planner which can avoid collisions and help an unexperienced user to find a path. The second one uses a fully interactive method to track the user position while animating the character at the same time, locally avoiding collisions.

We should integrate a real haptic device in order to use real forces and contact perception from the virtual scene instead of using only movements speeds. We have to define a normalization forces, in order to let the user decide

all along the process. Visual hints have to be inserted from the labels to have a visual feedback. Concerning the interactive locomotion planner, the algorithm performances depend on the length of the path to animate. In the interactive version, since the path is modified at each iteration, the animation has to be regenerated and the computation time grows with the length of the path. It can be interesting to modify the algorithm so that the computation time is constant with the length of the path by reusing previously computed animations.

References

[1] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo. OBPRM: An obstacle-based PRM for 3D workspaces. In *Workshop on Algorithmic Foundations of Robotics*, 1998.

[2] O. B. Bayazit, G. Song, and N. M. Amato. Enhancing randomized motion planners: Exploring with haptic hints. In *IEEE Int. Conf. on Robotics and Automation (ICRA'2000)*, pages 529–536, april 2000.

[3] V. Boor, M. H. Overmars, and A. F. van der Stappen. The Gaussian sampling strategy for probabilistic roadmap planners. In *IEEE Inter Conf. on Robotics & Automation (ICRA'99)*, pages 1018–1023, 1999.

[4] G. Bottesi, J.-P. Laumond, and S. Fleury. A motion planning based video game. Technical Report 04576, LAAS-CNRS, 2004.

[5] G. Burdea. Haptics issues in virtual environments. In *Computer Graphics International*, pages 295–302, 2000.

[6] C. Chabal, C. Megard, and L. Sibile. Emm-3d : a virtual environment for evaluating maintenability from cad models. In *Laval Virtual*, 2005.

[7] L. Chen and C. G. Brown. A 3d mouse for interacting with virtual objects. In *IEEE Inter. Symposium on Circuits and Systems*, 2005.

[8] J. Chestnutt, P. Michel, K. Nishiwaki, J. Kuffner, and S. Kagami. An intelligent joystick for biped control. In *IEEE Inter. Conf. on Robotics and Automation*, may 2006.

[9] J. Chestnutt, K. Nishiwaki, J. Kuffner, and S. Kagami. Interactive control of humanoid navigation. In *IEEE/RSJ Inter. Conf. on Intelligent Robots and Systems (IROS'09)*, pages 3519 –3524, 2009.

[10] E. Ferré, J. Laumond, G. Arechavaleta, and C. Estevés. Progresses in assembly path planning. In *Int. Conf. on Product Lifecycle Management (PLM'05)*, pages 373–382, july 2005.

[11] D. Flavigné. *Planification de mouvement interactive: coopration humain-machine pour la recherche de trajectoire et l'animation*. PhD thesis, LAAS-CNRS, Université Paul Sabatier, September 2010.

[12] D. Flavigné, M. Taïx, and E.Ferré. Interactive motion planning for assembly tasks. In *IEEE Int. Symposium on Robot and Human Interactive Communication (RO-MAN 09)*, pages 430–435, 2009.

[13] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics & Automation*, 12(4):566–580, June 1996.

[14] L. Kovar, M. Gleicher, and F. Pighin. Motion graphs. In *Inter. Conf. on Computer Graphics and Interactive Techniques*, pages 473 – 482. ACM SIGGRAPH, 2002.

[15] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006. Available at http://planning.cs.uiuc.edu/.

[16] S. M. LaValle and J. J. Kuffner. Rapidly-exploring random trees: Progress and prospects. In *Proceedings Workshop on the Algorithmic Foundations of Robotics (WAFR'00)*, 2000.

[17] Z. Liangjun, Y. Kim, and D. Manocha. A hybrid approach for complete motion planning. In *IEEE/RSJ Inter. Conf. on Intelligent Robots and Systems*, 2007.

[18] N.Ladeveze, J.Y.Fourquet, B.Puel, and M.Taïx. Haptic assembly and disassembly task assistance using interactive path planning. In *IEEE Virtual Reality (IEEE VR 09)*, 2009.

[19] J. Pettré. *Planification de mouvements de marche pour acteurs digitaux*. PhD thesis, LAAS-CNRS, Université Paul Sabatier, September 2003.

[20] J. Pettré, T. Simeon, and J.-P. Laumond. Planning human walk in virtual environments. In *IEEE/RJS Inter. Conf. on Intelligent Robots and Systems (IROS'02)*, 2002.

[21] J. Rosell, C. Vazquez, A. Perez, and P.Iniguez. Motion planning for haptic guidance. *Journal of Intelligents. Robotics Systems*, 53(3):223–245, 2008.

[22] B. Salomon, M. Garber, M. Lin, and D. Manocha. Interactive navigation in complex environments using path planning. In *Symposium on Interactive 3D graphics (I3D)*, pages 41–50. ACM SIGGRAPH, 2003.

[23] Z. Shiller, K. Yamane, and Y. Nakamura. Planning motion patterns of human figures using a multi-layered grid and the dynamics filter. In *IEEE Int. Conf. on Robotics and Automation (ICRA'2001)*, pages 1–8, 2001.

[24] Y. Su, W. Zhu, and T. Yu. Virtual assembly platform based on pc. In *Inter. Conf. on Audio, Language and Image*, 2008.

[25] Z. Sun, D. Hsu, T. Jiang, H. Kurniawati, and J. H. Reif. Narrow passage sampling for probabilistic roadmap planning. *IEEE Transactions on Robotics*, 21(6):1105–1115, 2005.

[26] T. Truong. *Un modele de locomotion humaine unifiant comportements holonomes et nonholonomes*. PhD thesis, LAAS-CNRS, Université Paul Sabatier, July 2010.

[27] T. Truong, D. Flavigné, J. Pettre, K. Mombaur, and J.-P. Laumond. Reactive synthesizing of human locomotion combining nonholonomic and holonomic behaviors. In *IEEE Inter. Conf. on Biomedical Robotics and Biomechatronics (BioRob)*, 2010.

[28] N. Ye, P. Banerjee, A. Banerjee, and F. Dech. A comparative study of assembly planning in traditional and virtual environments. *IEEE Transactions on Systems, Man, and Cybernetics*, 29(4):546 –555, 1999.