3D Collision Avoidance for Digital Actors Locomotion

Julien Pettré, Jean-Paul Laumond, Thierry Siméon
LAAS - CNRS
7, av. du Colonel Roche
31077 Toulouse Cedex 4 - FRANCE
jpettre,jpl,nic@laas.fr

***Abstract*— This paper presents some evolutions over the locomotion planning problem for digital actors. The solution is based both on probabilistic motion planning and on motion capture blending and warping. The paper particularly focuses on a new collision avoidance technique: while the legs and the pelvis of the digital actor follow a planned path, the animation of the upper part of the body is updated for 3D collision avoidance purposes.**

I. INTRODUCTION

Computer animation for digital actors is usually addressed by two research communities along two complementary lines: Computer Graphics community puts emphasis on realism of the motion rendering [1]. Realism can be obtained by motion imitation thanks to motion capture technologies. More recently, Robotics community tends to provide digital actors with capacity of action planning mainly in the area of motion planning (e.g., [2]).

This paper deals with combining both view points in the framework of virtual human locomotion. It focuses on the following problem: how to automatically compute realistic walking sequences while guaranteeing 3D obstacle avoidance?

An ef£cient approach consists in splitting the problem into two parts [3]. From the obstacle avoidance geometric point of view the digital actor is bounded by a cylinder. A 2D motion planner automatically computes a collision-free trajectory for the bounding cylinder. Then a motion controller is used to animate the actor along the planned trajectory. Performances are good enough to address dynamic environments and real time planning.

Locomotion in 3D is investigated in [4]: the workspace is modelled as a multi layered grid. Several types of digital actor bounding boxes are considered according to prede£ned locomotion behavior (walking, crawling...). Once a path is found in the grid, cyclic motion patterns are used to animate a trajectory along that path. The animation is then modi£ed by dynamic £lters to make it consistent.

Other similar approaches combining path planners and motion controllers have been investigated [5], [6], [7]. None of these approaches addresses the 3D component of the locomotion problem as it is illustrated in Figure 1. In the absence of obstacle, all the degrees of freedom of a digital actor are controlled by the locomotion behavior (e.g., the cyclic motions of the arms are naturally induced by the walk). Walking with a bar in cluttered environments may require the actor to control the motions of his arms independently from the legs. How to reach such an objective with motions as natural as possible?

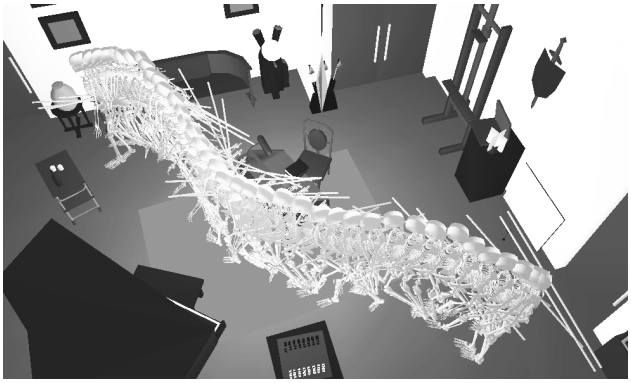


Fig. 1. A locomotion plan in a living-room

Our approach is an extension of [3]. A £rst collision-free path is computed in the 2-dimensional world by using a classical path planner. Then the geometric path is transformed into a trajectory in order to apply a motion controller generating walking animation from a motion capture data set. Obstacle avoidance is taken into account at two distinct levels. The £rst planned path is guaranteed to be collision-free for the lower part of the actor body (i.e. the pelvis together with the legs). Then by applying the motion controller along that path, all the degrees of freedom of the actor are animated. Collision checking is then applied on the whole body. Only the upper parts of the body can be in collision. When collisions occur in subsequences of the animation, each of these subsequences is processed with a warping technique [8]. The warping technique is applied only to the degrees of freedom to be corrected to avoid collisions. It just slightly modi£es the prede£ned animation on these degrees of freedom. In such a way the realism of the original animation is preserved at the best. If the warping step fails, then the original path is invalidated, and a new search is performed by the path planner.

The proposed approach is based on two-level modelling of our 57 degrees of freedom (d.o.f.) actor Eugene

(Section II). The *active* degrees of freedom gather all the degrees of freedom attached to the legs: they are processed by the motion controller without any possible adaptation once a path is £xed. The *reactive* degrees of freedom gather all the other ones: they are attached to the upper parts of the body. Reactive d.o.f. are processed both by the motion controller and by the warping module. Section III presents the global architecture of the approach and summarizes the tools we have previously developed in this framework: the walk controller [9] as well as the path planner [10]. Section IV focuses on the warping technique which is the core module of the global system.

II. MODELLING EUGENE

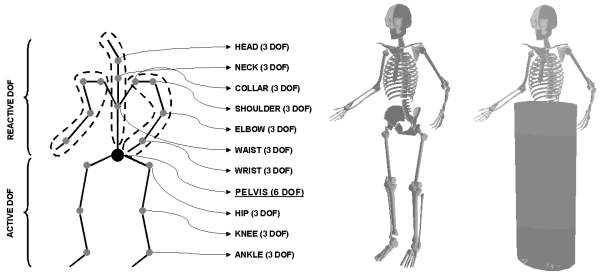


Fig. 2. The digital actor: Eugene

Eugene is the name of our digital actor (Figure 2). He is made of 20 rigid bodies and 57 d.o.f. The pelvis is the root of £ve kinematics chains modelling respectively the arms, the legs and the spine. The pelvis is located in the 3D space with 6 parameters. Its location £xes the location of Eugene in the environment. All the remaining 51 d.o.f. are angular joints.

Two classes of bodies are considered. Pelvis and the legs are responsible for the locomotion. All the 24 corresponding d.o.f. are said to be *active* d.o.f. The 27 other ones are said to be *reactive* d.o.f. They deal with the control of the arms and the spine.

Such a classi£cation is based on geometric issues dealing with obstacle avoidance. In the absence of obstacle, the walk controller (see below) has in charge to animate all the 51 angular d.o.f. along a given path. Due to the closed kinematic chain constituted by the ground, the legs and the pelvis, any perturbation on the active d.o.f. would affect the position of the pelvis, and then the given path. This is why we want the prede£ned path guaranteeing collision avoidance for all the bodies of the legs. Leg bodies and pelvis are then gathered into a bounding cylinder and the path planner (see below) computes collision-free paths for that cylinder. Possible collisions between obstacles and the upper part of Eugene are processed by tuning only the reactive d.o.f. without affecting neither the active d.o.f. nor the prede£ned path. Such a tuning is addressed by the warping module below.

III. PLANNING STEPS AND MODULES

The input data of the motion planner are a 3D environment, two (initial and £nal) con£gurations for Eugene and a walk controller allowing to steer Eugene along a prede£ned path. The output is an animated sequence showing Eugene walking to reach his goal while avoiding all the obstacles. The various steps are the following ones:

1) Plan a collision-free path for the cylinder bounding the legs and the pelvis: module `Path-Planner`,
2) Transform the path into a trajectory according to the velocity and acceleration bounds of the walk controller: module `Path-to-Traj`,
3) Animate Eugene along the trajectory: module `Walk-Control`. The animation appears as a sequence of key-frames.
4) Partition the animated sequence into alternate subsequences of collision-free key-frames and colliding key-frames. On each colliding subsequence apply the `Warping` module.

The role of the warping module is to plan a collision-free motion for the concerned reactive d.o.f. without changing the active ones along the subsequence of colliding key-frames. When the `Warping` module fails in computing a collision-free motion, the initial path computed at Step 1 is invalidated and a new path search is performed. The following modules have been implemented within Move3D [11], a software platform dedicated to motion planning.

`Path-Planner` The £rst step deals with collision-free path planning for a cylinder moving among 3D obstacles. This is a classical problem that can be solved by numerous approaches [12]. The module we have implemented is based on a probabilistic roadmap approach [13]. A roadmap is a graph aimed to capture the topology of the con£guration space of a given mechanical system. Roadmap nodes are collision-free con£gurations. A steering method being given to compute a path between two con£gurations, two nodes are adjacent if the path computed by the steering method is collision-free. According to the symmetry property of the steering method, a roadmap is either a directed graph or an undirected graph. For our application (human locomotion) the computed path should be smooth enough. Bézier curves of the third degree are used. This gives rise to directed roadmaps, and the probabilistic roadmap builder was adapted to produce directed graphs.

`Walk-Control` The walk controller (see [9] for details) computes the values of all the degrees of freedom of the digital actor when applying a 2-dimensional control (v, ω), v and ω being the linear velocity and the angular velocity respectively. This is the critical module that addresses the motion realism constraint. The controller is built on a motion capture library using a classical blending

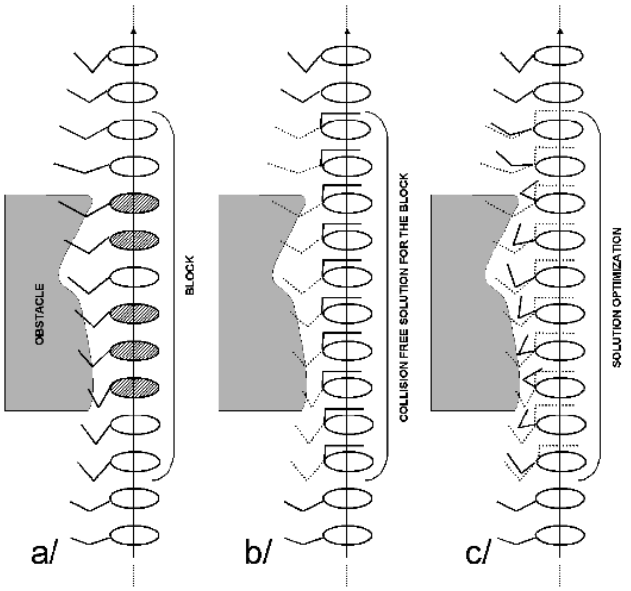


Fig. 3. Warping method steps

technique [14], [15], [16], [17] consisting in merging recorded motion data to generate new animations. The output of this module is an animation buffer (i.e. sequence of key-frames).

`Path-to-Traj` The transformation of a path into a trajectory respecting velocity and acceleration constraints is a classical problem in Robotics (e.g., [18]). In our application the function of `Path-to-Traj` is to provide a sampling of the path into points (x_i, y_i) compatible with the application of a control (v_i, ω_i).
The following section presents the original `Warping` module we have developed to address 3D collision avoidance.

IV. THE WARPING MODULE

The goal of the `Warping` module is to locally modify the animation of the upper bodies of Eugene (arms and spine) when collision occur in the animation produced by Step 3. The output of Step 3 is a sequence of key-frames which a complete speci£cation of all the 57 d.o.f.

Each key-frame of the sequence is scanned and a collision test is performed. At this level only the bodies of the upper part of Eugene may be in collision. Leg bodies as well as pelvis are necessarily collision-free. If a collision exists, the frame is marked with a label (*left-arm, right-arm* or *head-spine*) according to the body involving a collision with the obstacles. All the marked frames are gathered into connected subsequences. Subsequences are extended to create blocks absorbing collision-free frames in the neighborhood of the colliding subsequences (Figure 3-a). Such a subsequence extension is considered to provide smooth motions able to anticipate the corrective actions to be done. Each connected frame block is then processed independently.

Let us consider the example of a block with a *left-arm* label. The method consists in choosing a set of d.o.f. of the left arm at random until the left arm do not collide anymore. A new collision-free frame block is then created for which the d.o.f. of the left arm are modi£ed using this set of values (Figure 3-b).

Now we apply a warping procedure considering the two blocks: the original and the modi£ed one. Such a procedure is classical in graphics to modify a sequence of key-frames. By construction the two blocks have the same number of key-frames. The warping procedure consists in interpolating the reactive d.o.f. of the left arm. The parameters of the interpolation are controlled by the collision-checker in order to provide a new con£guration for the left arm which is as close as possible from the original con£guration while being collision-free (Figure 3-c).

V. RESULTS

Move3D, developed at LAAS-CNRS, has been used as support platform for implementing our method. This section presents some results obtained concerning the collision avoidance method described in the section IV. For legibility reasons, the results are illustrated with only few con£gurations. Exactly, one frame over three is used, over a small part of the whole computed animation.

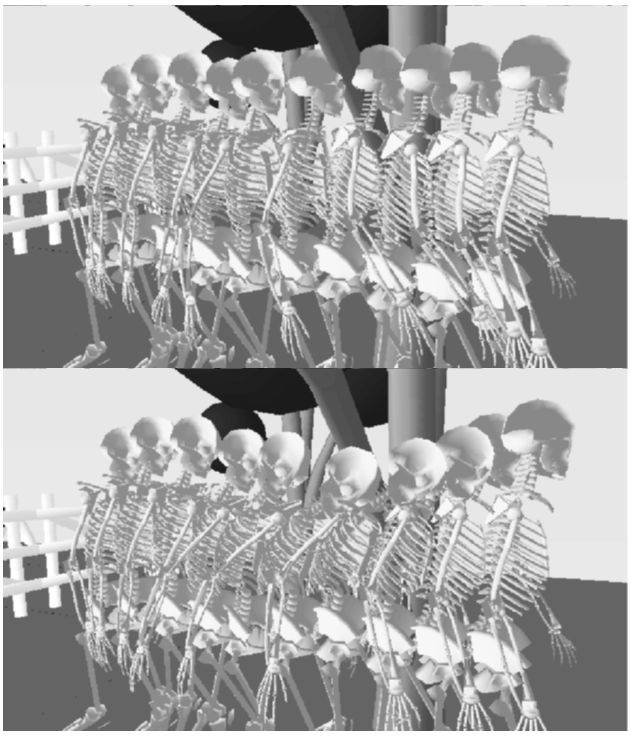


Fig. 4. Avoiding a tree branch

a) Careful with that branch, Eugene [19]: The £rst example is a human locomotion under a branch of a tree. The £rst image of the Figure 4 illustrates the result of the walk controller (i.e. when the trajectory is computed, optimized, sampled and animated). As expected, the lower part of the actor is collision free along the motion. But a collision occurs between the head and the branch; this is visible on the sixth con£guration drawn from the left.

The kinematic chain *head-spine* is detected in collision. The result of the warping procedure is illustrated on the second image at the bottom. Note that the new motion is realistic, and its amplitude is quite minimal. Also note that in this example, for one frame in collision, six are £nally modi£ed to avoid the obstacle. This correspond to the extension of the colliding subsequence into a block allowing an anticipation of the corrected motion.

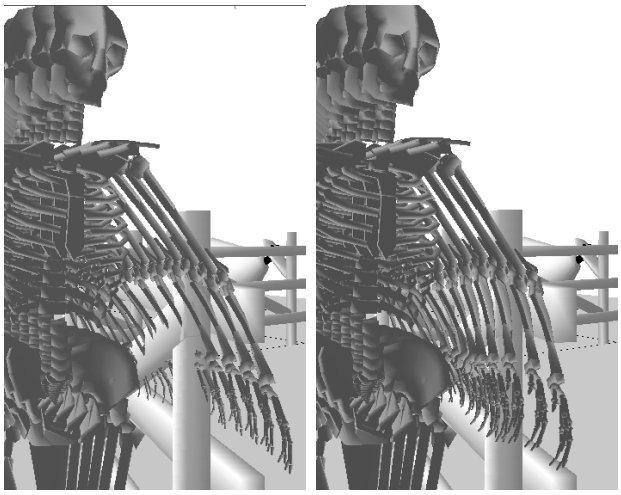


Fig. 5. Avoiding a wooden barrier

b) Careful with that barrier, Eugene: The second example deals with a collision between the left arm against a wooden barrier. The collision is illustrated on the left image of the Figure 5. Once again, a solution is found and illustrated on the right image.

The actor has “decided” to move its hand in front of him while walking. Some other solutions exist. As the search of a solution is not deterministic, each execution of the method rise to different solutions. The arm could have pass over the barrier. There is no criteria to score a solution. The user estimates the quality of the result. He may call the warping module again to evaluate another solution. He may also impose the con£guration randomly chosen by the warping module.

c) Careful with that chair, Eugene: Figure 6 illustrates that the model can be easily modi£ed for simulating the carriage of an object. The object does not

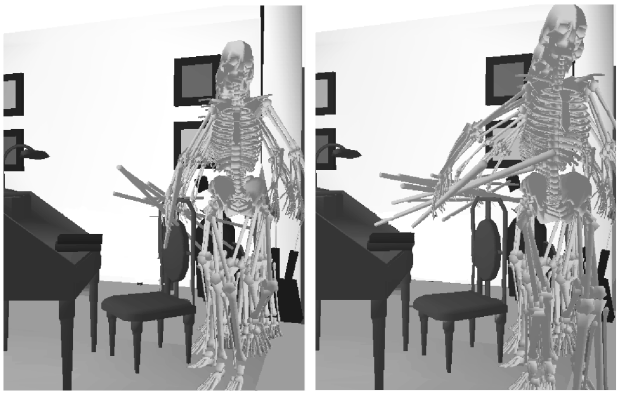


Fig. 6. Avoiding a desk, handling an object

in¤uence the result of the walk controller. But, as the bar is integrated to the right arm, the collision of the bar against the chair (see the left image of the Figure 6) is treated as explained previously. On the right image, a solution is illustrated: the bar goes over the chair and the desk.

d) Initial and £nal steps of a locomotion task: Our walk controller has a speci£city: it respects strictly the initial and the £nal con£gurations given as inputs of the problem. This means that while starting or ending the locomotion, the actor progressively adopt those con£gurations.

In the case of the Figure 7, the actor is asked to feed the virtual sheep. For that its hand must go on the other side of the barrier. When the walk controller is applied, the £nal position (where the actor feeds the sheep) is respected but some collisions exist.

Then collisions are avoided. This leads to modify the motion of the arm. As it concerns the last frames in the animation, the £nal position is affected: the hand stays away from the barrier, and the actor cannot feed the sheep anymore. The result is illustrated on the middle image of the Figure 7. In order to still reach the £nal position, a single query probabilistic motion planner [20] is used between the last modi£ed con£guration and the £nal (and desired) con£guration. The result is then sampled and added at the end of the animation. The additive frames are illustrated on the bottom image of the £gure.

A more constrained example of the positioning problem is illustrated on the Figure 8. The goal position of the actor is de£ned with the arm passing through the hole of a plank. The actor walks until he is in front of the plank then he passes its left arm in the hole of the plank.

e) Computational time: We give the computing times corresponding to the £gures of this paper. All the examples have been realized with a sampling rate of 25 frames per

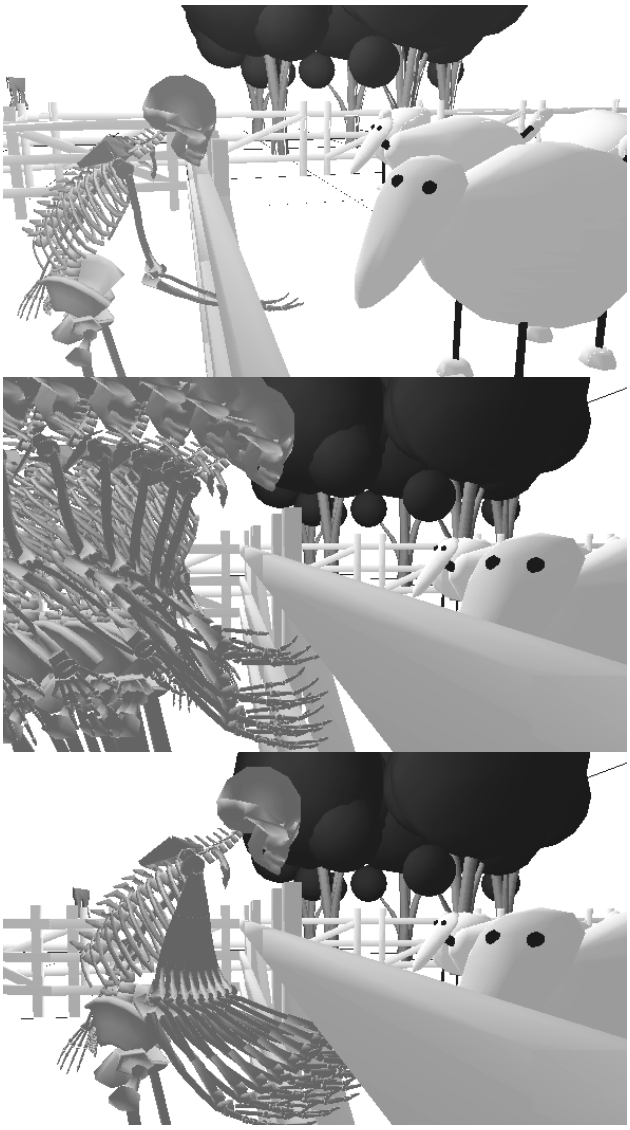


Fig. 7. Feeding a sheep

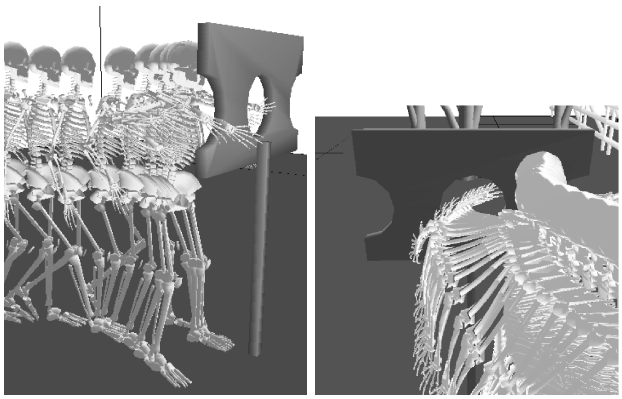


Fig. 8. Another example of an animation positioning the actor at the end of the trajectory

second, running on a Sun Blade 100 (proc. UltraSPARC-IIe 500MHz, 768Mo RAM). The total number of key-frames (Total KF) composing the animation and the number of frames constituting the colliding blocks (Block KF) are given, the computing times consumed by each module in seconds.

	Fig. 4	Fig. 5	Fig. 6	Fig. 7
# Total KF	85	64	53	69
`PathPlanner`	0.03*s*.	0.05*s*.	0.03*s*.	0.04*s*.
`PathToTraj`	0.32*s*.	0.22*s*.	0.19*s*.	0.34*s*.
`WalkControl`	0.1*s*.	0.07*s*.	0.05*s*.	0.07*s*.
# Block KF	35	29	53	17
`Warping`	0.53*s*.	0.23*s*.	3.27*s*.	0.28*s*.
Final Step	0*s*.	0*s*.	0*s*.	0.62*s*.

`Path-Planner` steps are very shorts as the given problems do not need a complex navigation in the environments. The `Path-to-Traj` problem consumes more computing time. The complexity of this step is very sensitive to the constraints (maximum velocities and accelerations). The `Walk-Control` step only depends on the number of key-frames in the animation, as the same model is used.

Note that the `Warping` module is largely more expensive in the case of the Figure 6. This is due to the more complex environment (the living room), and to the high number of frames in collision (due to the carriage of the bar and the proximity of the chair).

A problem concerning the £nal step, in order to respect the position of the arm of the actor passing through the barrier, appears only on the example of the Figure 7. The computing time includes the single query path search, and its sampling.

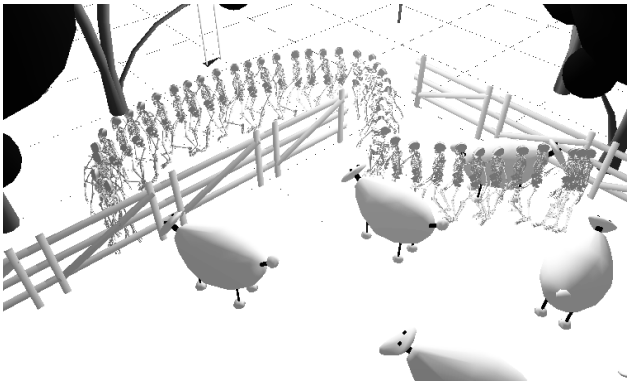


Fig. 9. A locomotion plan through the sheeps

Figure 9 illustrates a navigation problem. In this example an initial trajectory is found quickly (0.48*s*.). The `Warping` module computing time is essentially dedicated to the collision identi£cations (2*s*). Two blocks are formed: one for a collision against the sheep, another one against

the barrier. The solution search and the warping itself took 1.3*s*.

VI. CONCLUSION

A solution for planning the locomotion of a digital actor through a virtual environment has been presented in this paper. The solution is based on a PRM motion planner, on a motion capture blender and on an collision avoidance technique. The method is demonstrated over several examples, with satisfying results. The inputs of the problem are precise con£gurations, possibly constrained, and paths through narrow passages can be found. The implementation of the method is modular and controllers can be added, removed or replaced to test different solutions.

At this stage, the locomotion is limited to a ¤at terrain, the lower part of the body is not reactive, and the solution to avoid an obstacle while walking is limited to a single *sub-con£guration*. These limitations give some directions for the work to come. For example, the lower part of the body should be reactive: some solutions are described in the literature ([21]) and can be used as a base for a more sophisticated controller, allowing the walk on non-¤at terrains.

In a near future, we want to elaborate an intelligent strategy for constructing the initial roadmap. This could lead to more realistic trajectories with less optimizations on it, lowering the computing time. At last, the motion library needs to be extended with different types of walks (crawl, side steps, etc...). Our architecture can already mix several behaviors in a same plan but some adaptations are however needed to compute some transition phases. This would allow to £nd paths in even narrower passages.

VII. ACKNOWLEDGEMENTS

This work has been partly supported by the IST European Project 39250 MOVIE.

The environment of the living room in the example shown in Figure 1 and 6 has been provided by Daesign.

Thanks to F. Forges from Ex-Machina who gave us the motion capture data used in our examples.

Some videos of the results are available at this web page: *http://www.laas.fr/∼jpettre/*.

VIII. REFERENCES

[1] Rae Earnshaw, Nadia Magnetat-Thalmann, Demetri Terzopoulos, and Daniel Thalmann. Computer animation for virtual humans. *IEEE Computer Graphics and Applications*, pages 20–23, September/October 1998.

[2] Y. Koga, K. Kondo, J. Kuffner, and J.-C. Latombe. Planning motions with intentions. *Computer Graphics*, vol. 28(Annual Conference Series):395–408, 1994.

[3] James J. Kuffner Jr. Goal-directed navigation for animated characters using real-time path planning and control. *CAPTECH*, pages 171–186, 1998.

[4] Z. Shiller, K. Yamane, and Y. Nakamura. Planning motion patterns of human £gures using a multi-layered grid and the dynamics £lter. *IEEE Int. Conf. on Robotics and Automation*, 2001.

[5] D. Raulo, J.M. Ahuactzin, and C. Laugier. Controlling virtual autonomous entities in dynamic environments using an appropriate sense-plan-control paradigm. *Proc. of the 2000 IEEE/RS. International Conference on Intelligent Robots and Systems*, 2000.

[6] C. W. Reynolds. Steering behaviors for autonomous characters. *Proceedings of 1999 Game Developers Conference*, 1999.

[7] Min Gyu Choi, Jehee Lee, and Sung Yong Shin. Planning biped locomotion using motion capture data and probabilistic roadmaps. *ACM transactions on Graphics, Vol. 22(2)*, 2003.

[8] A. Witkin and Z. Popovic. Motion warping. *Proc. SIGGRAPH'95*, 1995.

[9] Julien Pettré, Thierry Siméon, and Jean Paul Laumond. Planning human walk in virtual environments. *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'02)*, 2002.

[10] T. Siméon, J.P. Laumond, and C. Nissoux. Visibility based probabilistic roadmaps for motion planning. *Advanced Robotics Journal 14(6)*, 2000.

[11] T. Siméon, JP. Laumond, and F. Lamiraux. Move3d: a generic platform for motion planning. *4th Internation Symposium on Assembly and Task Planning, Japan.*, 2001.

[12] Jean-Claude Latombe. *Robot Motion Planning*. Boston: Kluwer Academic Publishers, Boston, 1991.

[13] Lydia Kavraki, Petr Svestka, Jean-Claude Latombe, and Mark Overmars. Probabilistic roadmaps for path planning in high-dimensional con£guration spaces. Technical Report CS-TR-94-1519, 1994.

[14] Franck Multon, Laure France, Marie-Paule Cani, and Gilles Debunne. Computer animation of human walking: a survey. *The Journal of Visualization and Computer Animation*, 10:39–54, 1999.

[15] Charles F. Rose. *Verbs and Adverbs : Multidimensionnal motion interpolation using radial basis functions*. PhD thesis, Princeton University, June 1999.

[16] D. Wiley and J. Hahn. Interpolation synthesis for articulated £gure motion. *D. J. Wiley and J. K. Hahn. Interpolation synthesis for articulated £gure motion. In VRAIS '97, pages 156–160, Albuquerque, New Mexico, 1997.*, 1997.

[17] A. Bruderlin and L. Williams. Motion signal processing. *Proc. SIGGRAPH'95*, 1995.

[18] M. Renaud and J.Y. Fourquet. Time-optimal motions of robot manipulators including dynamics. *The Robotics Review n. 2*, 1992.

[19] Roger Waters, Richard Wright, Nick Mason, and David Gilmour. Careful with that axe, eugene. *Pink Floyd – Ummagumma*, 1969.

[20] James Kuffner and Steven LaValle. Rrt-connect : An ef£cient approach to single-query path planning. *IEEE International Conference on Robotics and Automation*, 2000.

[21] Shih kai Chung and James K. Hahn. Animation of human walking in virtual environments. *Computer Animation*, pages 4–15, 1999.