Eurographics/SIGGRAPH Symposium on Computer Animation (2003)
D. Breen, M. Lin (Editors)

A 2-Stages Locomotion Planner for Digital Actors

Julien Pettré [1] †, Jean-Paul Laumond [1], Thierry Siméon [1]

[1] CNRS – LAAS (Laboratoire d'Analyse et d'Architecture des Système), Toulouse, FRANCE

Abstract

This paper presents a solution to 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 describes the various components of our solution, from the first path planning to the last animation step. An example illustrates the progression of the animation construction all along the presentation.

Categories and Subject Descriptors (according to ACM CCS):

I.3.7 [Computer Graphics]: Motion Planning, Autonomous Characters, Probabilistic Roadmaps

1. 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 [3]. 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., [5]).

This paper takes advantage of both view points to address virtual human locomotion. It focuses on the following problem: how to automatically compute realistic walking sequences while guaranteeing 3D obstacle avoidance?

State of the Art An efficient approach consists in splitting the problem into two parts [6]. 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 sufficient to address dynamic environments and real time planning.

Locomotion in 3D is investigated in [15]: the workspace is modelled as a multi layered grid. Several types of digital actor bounding boxes are considered according to predefined locomotion behaviours (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 modified by dynamic filters to make it consistent.

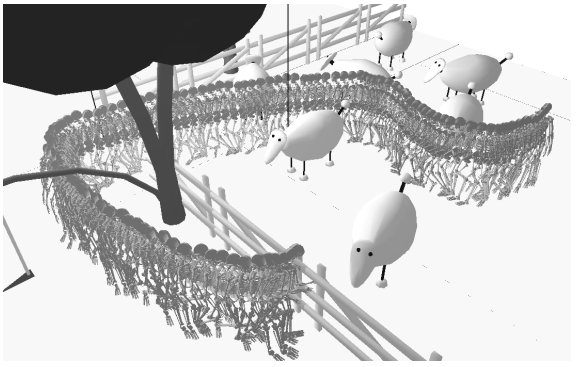

Figure 1: *Walking through the sheep*

Other similar approaches combining path planners and motion controllers have been investigated [12, 2]. None of these approaches addresses the 3D component of the locomotion problem. On one hand, the full integration of the animation process in a planning loop is time consuming. On the other hand, not considering the animation at the planning stage may lead to use bounding boxes, lowering the realism of the followed path (too far from obstacles, or even impossible in

† e-mail: jpettre,jpl,nic@laas.fr

the case of narrow passages). Our objective is a realistic locomotion planner, both considering the quality of the motion and the quality of the followed path. How to reach such an objective with motions as natural as possible?

Architecture The method presented in this paper is an extension of [6] and a continuation of the works introduced in [10]. The paper describes successively the main components of our approach. All of them are illustrated with the example presented in Figure 1 where the digital actor is asked to get out of a flock of sheep and to go in front of a wooden barrier, facing a sheep, and to feed it by passing his hand through the wooden barrier. The inputs of the problem are the 3D description of the environment and the two positions of the actor: the initial, standing in the flock of sheep, and the final one, facing the wooden barrier, feeding the sheep. All the bodies in the environment are considered as fixed obstacles to avoid.

The proposed approach is based on two-level modelling of our 57 degrees of freedom (d.o.f.) actor Eugene. Section 2 details the model: the *active* degrees of freedom gather all the degrees of freedom attached to the legs. The *reactive* degrees of freedom gather all the other ones: they are attached to the upper parts of the body; finally they are labelled with respect of the kinematic chain to which they belong.

A first collision-free path is computed in the 2-dimensional world by using a classical path planner described in Section 3. Then the resulting geometric path is transformed into a trajectory (Section 4). This step, similar to a sampling process, allows to adopt a classical animation data structure: a set of key-frames defining chronologically the successive positions of the actor. Our locomotion controller is described in Section 5. It generates a walking animation for all the previously computed key-frames from a motion capture data set.

Obstacle avoidance is taken into account at two distinct levels. The first planned path is guaranteed to be collision-free for the lower part of the actor body (i.e. the bodies with *active* degrees of freedom). 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 presented in Section 6. It slightly modifies the predefined animation on the reactive degrees of freedom. In such a way the realism of the original animation is preserved at the best.

Finally, the initial and final positions given as inputs of the whole problem may not be respected due to the modifications to the animation introduced by the warping module. Section 7 shows how to add two planned motions at the beginning and at the end of the animation to the animation in order to respect those positions.

2. Modelling Eugene

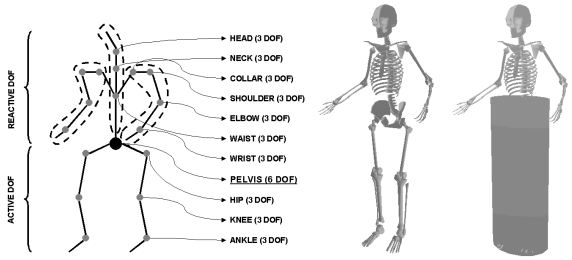

Figure 2: *The digital actor: Eugene*

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 five kinematics chains modelling respectively the arms, the legs and the spine. The pelvis is located in the 3D space with 6 parameters. Its location fixes 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 classification is based on geometric issues dealing with obstacle avoidance. In the absence of obstacle, the walk controller (Section 5) 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 predefined 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 predefined path. Such a tuning is addressed by the warping module (Section 6).

3. Path Planning

The objective of the `Path-Planner` module is to find a locomotion path through the environment between two given configurations of the actor. The locomotion path is an evolution of the position of the actor, thus it only deals with the position x, y of the actor and its orientation θ. The path ensures the collision-free motion of the lower part of the body, i.e. of its bounding cylinder.

The principle of this motion planning step is based on probabilistic roadmaps [4]. The main idea of such motion planners is to capture the connectivity of the collision-free configuration space in a roadmap. In our case, nodes are

collision-free positions and edges collision-free local paths for the cylinder bounding the lower part of the body. Local paths are Bezier curves of the third degree.

As Eugene is assumed to go only forward, the planner computes a directed roadmap. Once the search is performed we get a first locomotion path which is guaranteed to smooth. This first path is then optimised by a classical dichotomy technique maintaining smoothness. As a result, the output of this step is a sequence of local paths: a continuous composition of Bezier curves, i.e. a B-Spline.

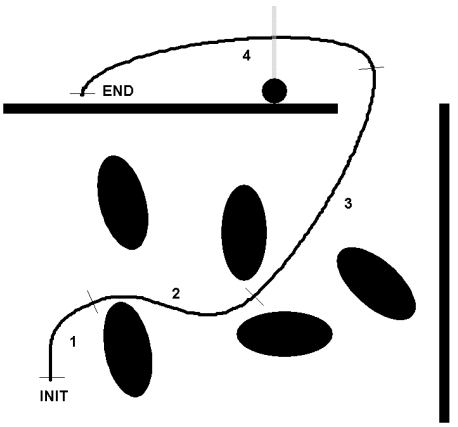

Figure 3: *A path made of 4 Bezier curves*

Figure 3 represents the path obtained for the problem illustrated on the Figure 1. This result is obtained thanks to our implementation, based on Move3D the motion planning platform developed a LAAS-CNRS [16]. The actor first surrounded by sheep, navigates through them in order to get out of the flock, then pass under a tree branch to face the wooden barrier, just in front of the sheep.

4. From Path to Trajectory

The module `Path-to-Traj` transforms the continuous parametric expressions of a 2D path into a discrete set of time stamped positions for the actor along the trajectory, respecting some criteria of maximal velocities and accelerations.

The 2D path is given by the `Path-Planner` module. It consists in a B-Spline, i.e. a continuous sequence of N Bezier curves. Thus, the position $P(u)$ along the path is described by a parametric expression $P(u) = [x(u),\ y(u),\ \theta(u)]^T$, where u is a real number belonging to $[0,N]$.

We want to introduce a time parameterisation by sampling the path. The time scale defined by $t_{i:0\rightarrow m} = i\ \pi$ where π is a constant number (the sampling period). Thus, we can then get a discrete time parametrised expression of P : $P(t_i) = P(u_i)$

We may introduce other discrete variables: v_i and ω_i respectively denoting the tangential and the rotational speeds for each time step, with respect $P(u_i)$, $P(u_{i-1})$, and π. Also, the tangential acceleration $\dot{v}_i(v_i,\ v_{i-1},\ \pi)$ is defined.

Additive variables are used to account for the following criteria:

1. $u_0 = 0$ and $u_m = N$,
2. $v_0 = 0$ and $v_m = 0$,
3. $0 < v_i < V_{max}$,
4. $-R_{max} < \omega_i < R_{max}$,
5. $-A_{max} < \dot{v}_i < A_{max}$,
6. m is minimal.

where m is the necessary number of time steps to achieve the trajectory sampling. V_{max}, R_{max} and A_{max} are user defined. The sampling rate, (i.e. the time period π) can also be defined.

The transformation of a path into a trajectory respecting velocity and acceleration constraints is a classical problem in Robotics (e.g., [11, 8]). We do not develop the whole procedure.

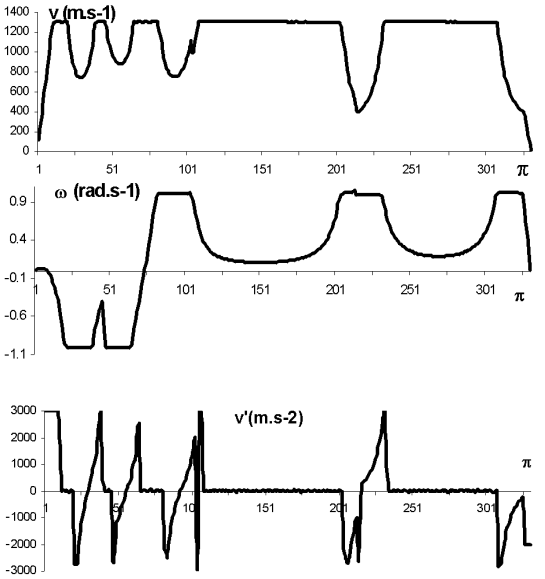

Figure 4: *Characteristic profiles: v, ω and $\dot{v}$*

Figure 4 illustrates the results of the `Path-to-Traj` step over the 2D-Path of the Figure 3. The v_i and ω_i speed evolutions are illustrated. The acceleration $\dot{v}_i$ is displayed at the bottom figure. Note that always one of the previously defined criteria reaches its bound: the solution is optimal.

5. Locomotion Controller

The module `Locomotion-Control` transforms a set of time parametrised positions into a walk sequence. It is based

on a motion capture blending technique. Therefore, it is composed of two elements: a motion capture library and a locomotion controller (see [9, 13, 19, 17, 14] for some descriptions of similar controllers, and [10] for a more detailed description of this controller).

In order to solve a locomotion problem as illustrated on Figure 1, the library is filled with only one type of locomotion: the walk, grouping several similar motion capture examples. The similarity is estimated with respect of different criteria: same skeleton, same motion structure and same behaviour (walking). So the difference between each motion cycle can be characterized by their average (tangential and rotational) speeds $(\tilde{v},\ \tilde{\omega})$. More precisely, motion captures are expressed in the frequency domain using Fourier expansions. $[x,\ y,\ \theta]$ parameters are modified as positioning errors around a virtual point moving at $(\tilde{v},\ \tilde{\omega})$. Such a transformation allows to make these parameters cyclic as the rest of the joint angles evolutions. Transformed variables are noted $[\Delta x,\ \Delta y,\ \Delta\theta]$.

The input of the locomotion controller is the set of time stamped positions computed by the module `Path-to-Traj`. Thus, the parameters $[P(t_i),\ v_i,\ \omega_i]$ of each time step t_i are considered. A motion blending formulæ is computed for each time step, from which a motion cycle MC_{t_i} is created. The characteristics speeds $(\tilde{v},\ \tilde{\omega})$ of MC_{t_i} are equal to $(v_i,\ \omega_i)$.

A single configuration q_i is the extracted from MC_{t_i}. $P(t_i)$ is considered to project q_i on the followed trajectory. Projection is computed by adding $[\Delta x,\ \Delta y,\ \Delta\theta]$ issued from MC_{t_i}, to the input parameters given by $P(t_i)$. The other angular values are replaced. The extraction ensures the continuity of the motion between the frames i and $i-1$ and as a result, over the whole trajectory.

Snapshots of some small parts of the locomotion controller output are illustrated on Figure 5. Note that the actor model allows us to guarantee collision free motion for the lower part of the body. However, some collisions exist between the upper part of the body and the environment. This is illustrated on the two images: between the hand and the sheep's on the top image, and between the actor's head and the branch on the bottom image.

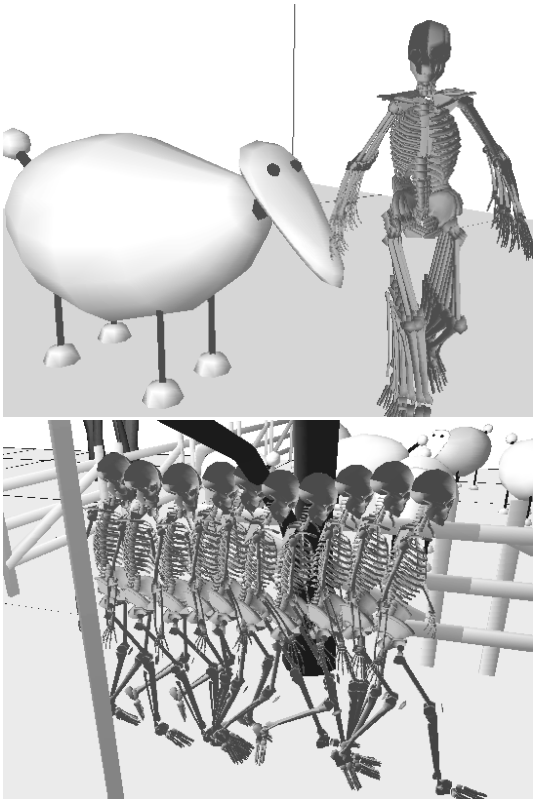

Figure 5: *Residual collisions*

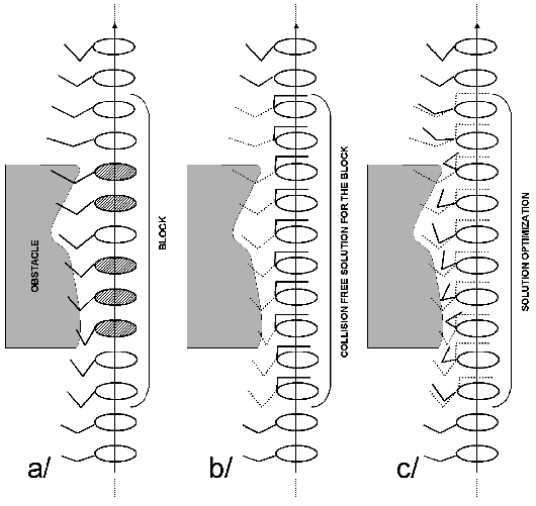

Figure 6: *Warping method steps*

6. Warping to Avoid

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 the module `Locomotion-Control`. At this stage, the animation is a sequence of key-frames which a complete specification 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 neighbourhood of the

colliding subsequences (Figure 6-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 modified using this set of values (Figure 6-b).

Now we apply a warping procedure considering the two blocks: the original and the modified one. Such a procedure aimed to modify a sequence of key-frames is a classical one in graphics [20, 1]. By construction the two blocks have the same number of keyframes. 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 configuration for the left arm which is as close as possible from the original configuration while being collision-free (Figure 6-c).

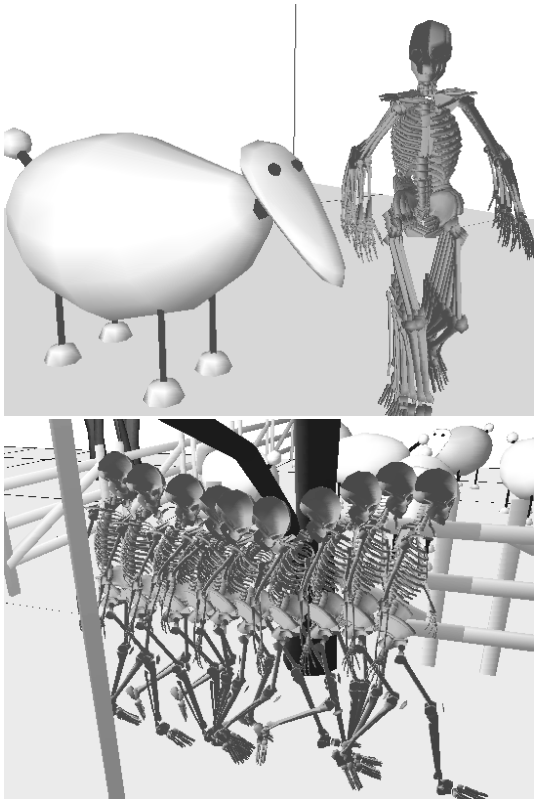

Figure 7: *Avoided branch and sheep*

Figure 7 illustrates the result of the `warping` module, over the two parts illustrated in Figure 5 (output of the `Locomotion-Control`). Note that the realism of the motion is preserved, and that the movement allowing to avoid the collision is quite minimal.

7. Initial and Final Positioning

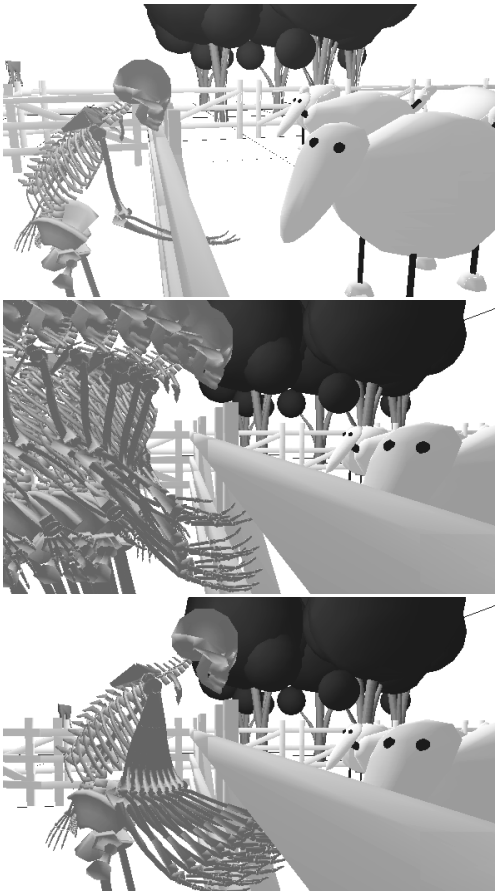

Figure 8: *Feeding a sheep*

Our walk controller has a specificity: it strictly respects the initial and the final configurations given as inputs of the problem. This means that while starting or ending the locomotion, the actor progressively adopt those configurations.

Remember that in the case of Figure 1, 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 final position (where the actor feeds the sheep) is respected but some collisions exist. Then collisions are avoided by applying the warping module. This leads to modify the motion of the arm. As it concerns the last frames in the animation, the final position is affected: the hand stays away from the barrier, and the actor cannot feed the sheep anymore: see the middle image of the Figure 8.

In order to still reach the final position, a single query

probabilistic motion planner [7] is used between the last modified configuration and the final (and desired) configuration. The result is then sampled and added at the end of the animation. The additive frames are illustrated on the bottom image of the Figure 8.

8. Discussion

The whole number of key-frames generated by the example of Figure 1 is 331. Computing times consumed on this example are distributed as follow :

1. Path search: 0.84 *s* (with a precomputed roadmap),
2. Path optimisation: 3. *s*,
3. Path to trajectory: 0.9 *s*,
4. Locomotion controller: 0.3 *s*,
5. Warping: 0.74 *s* + 0.87 *s* (collisions identification and their solution + warping itself),
6. Initial and final positioning: 0.98 *s*,

The computation times indicated above were measured on a standard Unix workstation † and would therefore be much lower using a high end computer. One can note that path optimisation is the more time consuming step of our locomotion planner. An effort is currently done to improve the efficiency of this step which currently uses a generic dichotomic method to optimise the path; the steering method for digital actors is quite particular and should be optimised in a different manner. Also note that this is a "still in development" and a non-optimised code.

More generally, computing times are sensitive to several elements. First, the complexity of the environment. The use of precomputed roadmaps allow to preserve the performance of the path search in the query phase. Nevertheless, environment complexity extends the time necessary to solve collisions in the warping module and to plan motions in the last positioning module.

Figure 9 illustrates a more complex environment: a living room. To increase the complexity of the locomotion task, we added a long bar in the right hand of Eugene. This version of the model is named CWTB-Eugene (*Careful with that bar Eugene* [18]). So the problem for CWTB-Eugene is now to handle a bar while crossing the living room. In order to avoid the desk, the piano and other furniture, CWTB-Eugene carries the bar in front of him. Detailed views appear in Figure 9.

Time consumed by the `Path-To-Traj` module depends on the user limits given. The more constraining they are, the more this module is consuming. Also, the total number of frame is critical. Indeed, the `Warping` module will test the collision existence for each frame. Over more simple problems, such as illustrated on Figure 10 (with an alternative skinned version of Eugene), the whole result is obtained in less than a second.

† Sun Blade 100, proc. UltraSPARC-IIe 500 MHz, 768 Mo RAM

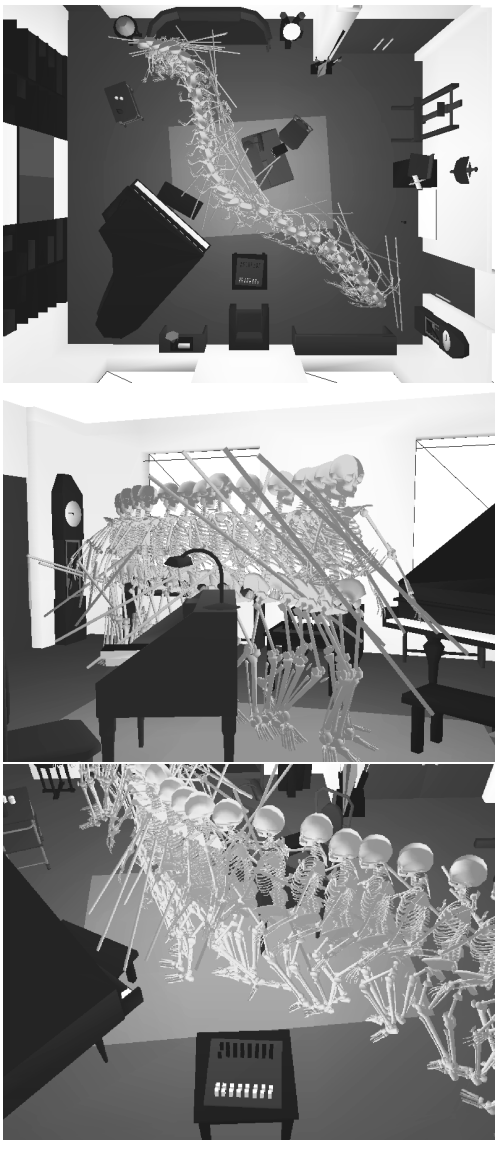

Figure 9: *Back home*

9. Conclusion

We have presented a solution for digital actors locomotion problems. This paper insists on the modularity of the approach. Each component can be modified or replaced with ease. The example detailed along the paper illustrates the role of each component through the locomotion planning process and demonstrates the realism of the result.

Accounting for the 3D model of the environment is a specificity of our solution. Through our results, the navigation close to obstacles and their avoidance, thanks to little movements of the upper body, gives an illusion of a real in-

Figure 10: *Skinning Eugene*

teraction between the digital actor and the digital environment.

Our locomotion planner is still to be enhanced: we want to introduce the ability for the digital actor to change its locomotion behaviour by crouching, crawling, etc. (See [9] for an overview of the human walk animation problem). This objective raises some new needs for our solution: the extension of the content of our motion library, the introduction of rules to change from a behaviour to another. Some approaches of those problems exist in the literature. But a main part of our architecture is already designed to deal with such problems. Those enhancements are planned in our future works. Finally the scope of the approach at its current stage is restricted to walking tasks on flat floor. Future works will integrate rough terrains as well as stairs.

Acknowledgement: The environment of the living room in the examples shown in Figure 9 and 10 have been provided by Daesign.

References

1. A. Bruderlin and L. Williams. Motion signal processing. *Proc. SIGGRAPH'95*, 1995.
2. M. Choi, J. Lee, and S. Shin. Planning biped locomotion using motion capture data and probabilistic roadmaps. *ACM transactions on Graphics*, 2003.
3. R. Earnshaw, N. Magnenat-Thalmann, D. Terzopoulos, and D. Thalmann. Computer animation for virtual humans. *Computer Graphics and Applications*, 1998.
4. L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. 1994.
5. Y. Koga, K. Kondo, J. Kuffner, and J.-C. Latombe. Planning motions with intentions. *Computer Graphics*, vol. 28(Annual Conference Series):395–408, 1994.
6. J. Kuffner. Goal-directed navigation for animated characters using real-time path planning and control. *CAPTECH*, pages 171–186, 1998.
7. J. Kuffner and Steven LaValle. Rrt-connect : An efficient approach to single-query path planning. *IEEE Int. Conference on Robotics and Automation*, 2000.
8. F. Lamiraux and J.-P. Laumond. From paths to trajectories for multi-body mobile robots. *5th Int. Symp. on Experimental Robotics*, 1997.
9. F. Multon, L. France, M.-P. Cani, and G. Debunne. Computer animation of human walking: a survey. *Journal of Visualization and Computer Animation*, 1999.
10. J. Pettré, T. Siméon, and J.-P. Laumond. Planning human walk in virtual environments. *Int. Conf. on Intelligent Robots and Systems (IROS'02)*, 2002.
11. M. Renaud and J.Y. Fourquet. Time-optimal motions of robot manipulators including dynamics. *The Robotics Review n. 2*, 1992.
12. C. W. Reynolds. Steering behaviors for autonomous characters. *Proceedings of 1999 Game Developers Conference*, 1999.
13. C. Rose. *Verbs and Adverbs : Multidimensionnal motion interpolation using radial basis functions.* PhD thesis, Princeton University, June 1999.
14. B. Salmon, M. Garber, M. Lin, and D. Manocha. Interactive navigation in complex environments using path planning. *ACM S. on Interactive 3D Graphics*, 2003.
15. Z. Shiller, K. Yamane, and Y. Nakamura. Planning motion patterns of human figures using a multi-layered grid and the dynamics filter. *IEEE Int. Conf. on Robotics and Automation*, 2001.
16. T. Siméon, JP. Laumond, and F. Lamiraux. Move3d: a generic platform for motion planning. *4th Int. Symposium on Assembly and Task Planning, Japan.*, 2001.
17. M. Unuma, K. Anjyo, and R. Takeuchi. Fourier principles for emotion-based human figure animation. *Proc. od SIGGRAPH'95*, 1995.
18. R. Waters, R. Wright, N. Mason, and D. Gilmour. Careful with that axe, eugene. *Pink-Floyd Ummagumma*, 1969.
19. D. Wiley and J. Hahn. Interpolation synthesis for articulated figure motion. *VRAIS '97*, 1997.
20. A. Witkin and Z. Popovic. Motion warping. *Proc. SIGGRAPH'95*, 1995.

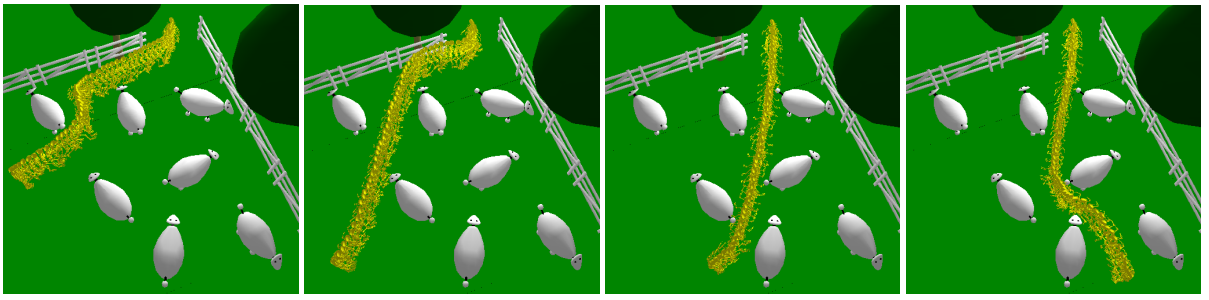

Figure 11: *4 different examples of global trajectories. The initial position is modified while the final is kept identical.*

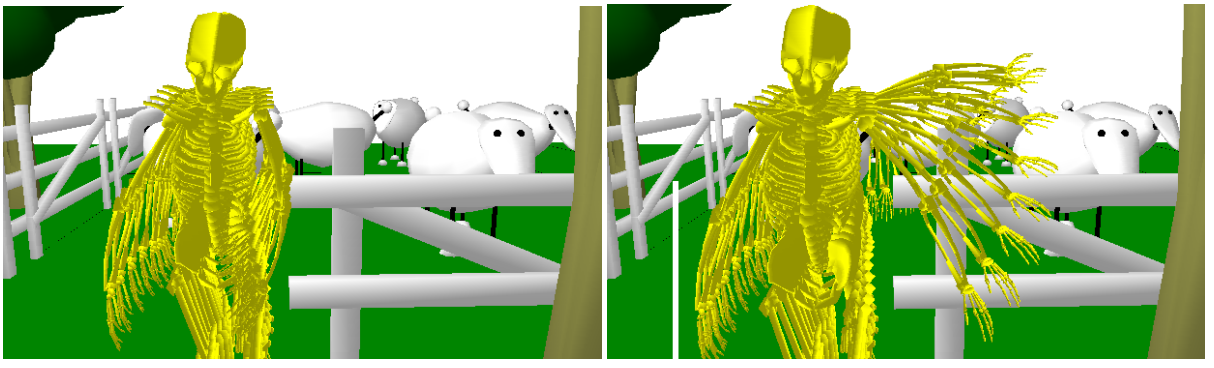

Figure 12: *By rerunning the warping module only, different solutions are explored as the collision free motion is randomly searched. On the left image, the arm goes in front of Eugene, while on the right image the arm goes over the barrier.*

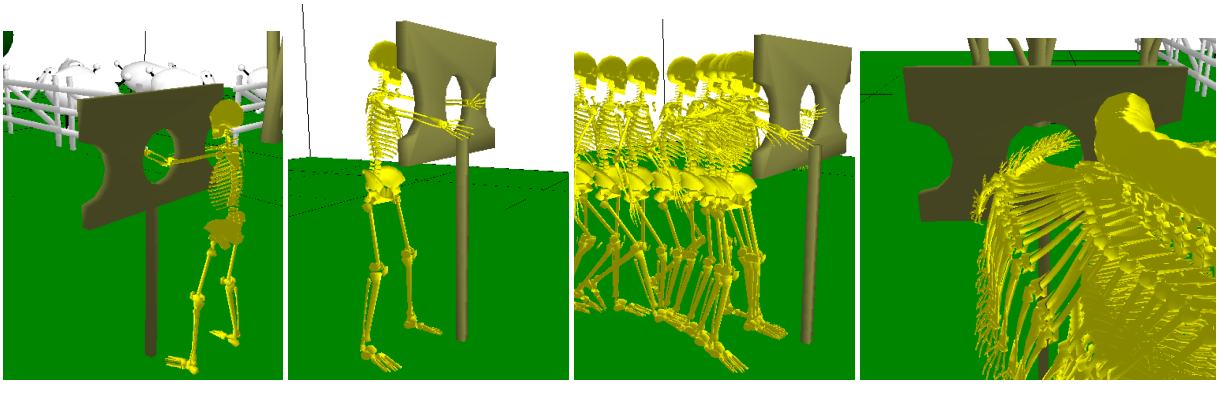

Figure 13: *A more constrained goal position (see the two images on the left) is illustrated on this example. The two images on the right illustrate the solution given by the planner (the end of the locomotion plus the final positioning motion)*