Efficient elimination of foot sliding for crowds

A. Beacco[1], B. Spanlang[2] & N. Pelechano[1]

[1]Dept. de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, Spain
[2]Dept. de Personalitat, Avaluació i Tractaments Psicològics, Universitat de Barcelona, Spain

Abstract

This poster presents an efficient Animation Planning Mediator (APM) designed to animate virtual characters in real time for crowd simulation. The APM selects the most appropriate animation clip available for each character and modifies the skeletal configuration to satisfy constraints given by the virtual environment and crowd simulation (CS) module, such as eliminating foot-sliding. Using a reduced number of animation clips, we blend within and between animation clips to increase the number of possible locomotion types and to adjust the animations to the velocity of each agent as indicated by the CS module. A key advantage of our approach is that it can be easily integrated with any existing real-time crowd simulation module working in continuous space.

Keywords: *Crowd Simulation, Animation, Foot Sliding.*

1. Introduction

Crowd simulation in real time for computer graphics applications requires algorithms for not only agent navigation in large virtual environments while avoiding obstacles and other agents, but also rendering complex scenes with 3D articulated figures to enhance realism. Therefore, trade-offs between accuracy and speed are necessary. Although many techniques have been developed for synthesizing the motion of one agent [GMHP04, KGP02], they are not easily extended to large numbers of agents simulated in real time.

In most real time crowd simulation models based on flow fields, social forces, rules, roadmaps or hybrid systems, each agent's root position is updated without taking into account either the underlying animation being played, their torso orientations or the contact points between their feet and the floor [PAB07]. As the overall realism of the animation and rendering improves, artifacts such as foot-sliding become more noticeable.

Other models put effort into avoiding foot-sliding by translating the root based on the animation frame selected at each time step. To be able to run in real time, they employ a small animation graph which discretizes and limits agent movement [ST05, LK06].

We present an Animation Planning Mediator (APM) which selects the best parameters to feed a motion synthesizer at the same time that it feeds back to the crowd simulation module the required updates to guarantee consistency. Our technique allows a large and continuous variety of movement. Our method can be used with any crowd simulation software, since its work is limited to adjusting the root displacement and modifying the skeletal state to satisfy the constraints given.

2. Animation Planning Mediator

Our approach uses a feedback loop where the APM acts as the communication channel between a crowd simulation module, CS, and a character animation and rendering module, CA. The outline of this framework is shown in Figure 1.

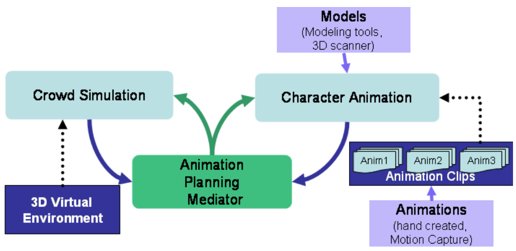

Figure 1: *Framework*

The APM pre-processes the animation clips available to extract information necessary to classify them so that it can be accessed in real time during the simulation. For each animation clip, A_i, the APM calculates its average velocity $\mathbf{v}_{anim_i}$ in m/s (by computing the total distance traveled by the character through the animation clip divided by the total time of the animation clip), as well as the angle α between the torso orientation and the velocity vector $\mathbf{v}_{anim_i}$.

Figure 2 shows the steps that the APM goes through during the real-time simulation, as well as the input/output of each module.

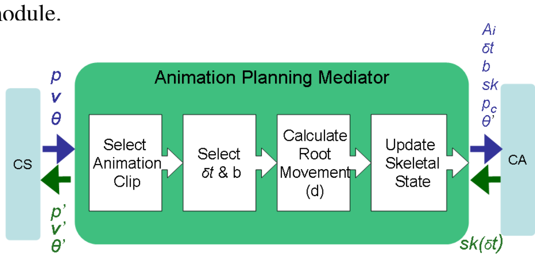

Figure 2: *APM as the communication channel*

Where p, $\mathbf{v}$ and θ are the position, velocity and torso orientation given by the CS; δt, b, and sk are the differential of time to play A_i, the blending factor and the skeletal configuration. Finally p_c and θ' are the computed position and orientation. The APM may need to slightly adjust the parameters given by the CS in order to guarantee smooth and continuous animations.

To avoid foot-sliding and guarantee that the animation satisfies the constraints given by $\mathbf{v}$ and θ, we need to ensure that the character appears to move according to $\mathbf{v}$ while the foot currently in contact with the floor stays in place, and the torso faces the direction given by θ. This could be done using inverse kinematics, but since we are simulating large crowds, we need a method that can be quickly calculated and applied to hundreds of 3D animated figures in real-time.

At each time step, given $\mathbf{v}$ and θ of the agent, we select the best animation clip, A_i. Knowing the agent's velocity $\mathbf{v}$ and the velocity of the animation $\mathbf{v}_{anim_i}$ we can calculate the δt in order to get the CA to produce the animation frame that best matches the current velocity of the character. If $A_i(t)$ is different from $A_i(t-1)$, then the blending factor b is calculated.

Following this we need to gather the local coordinates of the root and the feet position for the new frame. For the current foot on the floor, we calculate the vector that goes from the foot to the root of the skeleton at this frame $\mathbf{u}_f(t)$ and for the previous frame $\mathbf{u}_f(t-1)$. (Figure 3). The vector $\mathbf{u}_f(t)$ contains all the rotations that happen at the ankle and knee level and thus provides sufficient information about the leg movement. By subtraction of vectors we can easily and rapidly

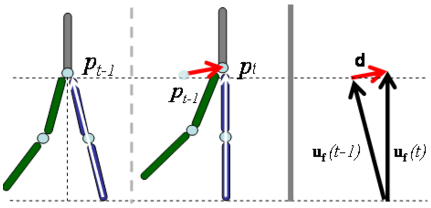

Figure 3: *Root Displacement*

calculate the root displacement $\mathbf{d}$ between frames. Adding the vector $\mathbf{d}$ to the current root position will give us the next position of the root. This efficient technique provides a high level of accuracy without becoming a bottleneck in the process of simulating large crowds, plus it conserves the up and down movement of the character's waist.

Turns in the environment can be achieved by reorienting the figure aligning $\mathbf{v}_{anim}$ with $\mathbf{v}$. Knowing the torso orientation θ given by the CS and the angle α from the animation clip, we can calculate the angle ψ to adjust the skeletal configuration, as shown in Figure 4.

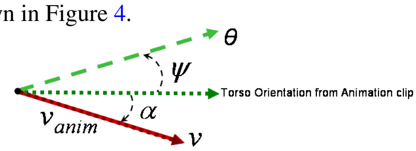

Figure 4: *Torso re-orientation*

Orienting the character in global coordinates according to the velocity vector, $\mathbf{v}$, will also orient the displacement vector $\mathbf{d}$ to move the character according to the CS module, while the skeletal adaptation allows the character to face the torso orientation indicated by the CS. The visual effect of a turn is that the foot on the floor will slightly rotate in place. This is almost unnoticeable to the human eye, especially at high velocities, and thus is a trade-off worth considering as we can achieve turns in any direction without the requirement of having a large database of animations.

3. Results

Our algorithm requires less than 0.13ms to run the APM, of which 0.04ms are employed to calculate the displacement of the root position. These times are calculated per agent and per frame, on a Dual Core 2.67 GHz, Intel processor. The total cost of incorporating the APM into any crowd simulation module is thus 0.13ms times the number of agents. Therefore, the total time required per frame will increase linearly with the crowd size. Despite this small computational cost, the visual results achieved appear to drastically reduce foot-sliding and provide the user with a significantly more realistic experience than would otherwise occur

References

[GMHP04] Grochow K., Martin S. L., Hertzmann A., Popović Z.: Style-based inverse kinematics. *ACM Transactions on Graphics 23*, 3 (2004), 522–531.

[KGP02] Kovar L., Gleicher M., Pighin F.: Motion graphs. *ACM Transactions on Graphics 21, 3* (2002), 473–482.

[LK06] Lau M., Kuffner J.: Precomputed search tree: Planning for interactive goal-driven animation. *ACM SIGGRAPH / Eurographics Symp. on Computer Animation* (2006), 299–308.

[PAB07] Pelechano N., Allbeck J., Badler N.: Controlling individual agents in high-density crowd simulation. *ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (2007), 99–108.

[ST05] Shao W., Terzopoulos D.: Autonomous pedestrians. *ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (2005), 19–28.