ICAT 2003
December 3-5, Tokyo, JAPAN
THE VIRTUAL REALITY SOCIETY OF JAPAN

Motion Generation Using Motion Mining System

Seongmin Baek, Il-Kwon Jeong, Inho Lee

Digital Actor Research Team, ETRI
161 Gajeong-dong, Yuseong-gu, Daejeon, Korea
{baeksm, jik, leeinho}@etri.re.kr

Abstract

In this paper, we present a method that generates a path that has no collision with the obstacles by using the environment information, and automatically creates natural motions of characters that are confined to the path. Our Method is based on two kinds of parameters: Environment and Behavior parameters. The behavior parameters include velocity, motion status such as walking and running, and preference. The environment parameters include road and obstacles. A collision-free path is generated by these parameters, and a motion mining system generates the motion that has motion status and velocity that a user inputted while following a path. Our system predicts and avoids a collision by rule-based approach.

Key words: motion-generation, path-generation, motion parameter, motion mining, collision avoidance

1. Introduction

These days a human-like character is widely used in the multi-media contents such as games and movies, and especially the scenes that many characters are appearing are increased. These many characters, the digital extras, are not important being but play a role in increasing the reality of contents. However, it is a hard labor and a waste of time if the animator controls the each character using existing traditional method, that is, key-frame animation. Suppose the scenes as follows, for example. There are one hundred characters and they follow a certain path. Some have fast movement, others move slowly. The faster overtakes the slower. The characters, which are moving, must avoid the obstacles. Although their motions are all 'walking', an animator must modify the motions because it looks unreal if all their movements are the same. Therefore, a new method is needed for creating these extras. In this paper, we present the method that generates a path, which has no collision with the obstacles using environment information, and creates automatically the new motion following the path by motion-mining system. We define motion mining as editing and selecting proper motions automatically from motion database that consists of a number of short motion clips such as walking, jogging, and running. However, the motion trajectory controls overall motion of a character and does not deal with collisions among characters. Therefore we present how to avoid a collision with other characters.

Our system consists of three processes: a path generation using parameters, a motion generation following a path, and collision avoidance. Figure 1 shows an outline of the system.

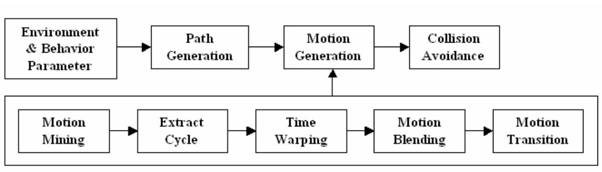


Fig 1. System overview

The structure of the paper is as follows. In section 2, we describe previous work on motion generation technique and section 3 explains how to generate a path that controls a digital actor by parameters. We introduce a motion generation method using motion mining system in section 4, and section 5 contains collision avoidance rule. Experimental results are presented in section 6.

2. Previous Work

The techniques for controlling a character are growing increasingly popular. Many systems have been suggested for controlling a virtual human but crowd systems not yet. F. Multion [7] surveyed the set of techniques developed in animating human walking. In his survey, he focused on the evolution from inverse kinematics (IK) to dynamics and review motion editing technique based on motion capture data. Tolani et al. [6] solved the IK problem of a human arm and leg by analytic solution. A numerical method of IK relies on an iterative process to obtain a solution and Rose et al. [4] solved non-linear optimization problem with various constraints. Lee et al. [10] presented a technique that adapts a motion of a character by using a hierarchical curve fitting and inverse kinematics solver. Bruderlin et al. [1] presented a hybrid approach to animate the human locomotion, which combines goal-directed and dynamic motion control. Tak et al. [15] proposed motion balance filtering, which corrects an unbalanced motion to a balanced one while preserving the original characteristics as much as possible. The above techniques do not consider many characters, however.

Ulicny [3] developed the crowd simulation for interactive virtual environments such as VR training system for urban emergency situations. Farenc [12] proposed a new architecture for simulating virtual humans in complex environment and Kallmann [11] designed 'smart object' based on a complete definition and representation of interactive objects. Sulivan et al. [5] presented crowd and group simulation with level of detail (LOD). These techniques, however, focused on mainly behaviours in specific environment.

Recently, research efforts have developed a probabilistic model for motion synthesis based on the motion capture data [8][9][13][14]. Lee et al. [8] showed that a connected set of a human-like character can be generated from non-linear sequences of motion, automatically organized for search, and used real-time control of an avatar using three interface techniques: selecting from a set of available choices, sketching a path, and acting out a desired motion in front of a camera. Kovar et al. [9] constructed a directed graph called a 'motion graph' that includes connections among the database for creating realistic, controllable motion. The motion graph consists of original motions and automatically generated transitions. Li et al. [13] described 'motion texture' for synthesizing complex human character that is statistically similar to the original motion captured data. Motion texture can be manipulated by modifying the details of a specific motion at the texton level or by generating a new motion at the distribution level. Pullen et al. [14] discussed a method for creating animations by setting a small number of keyframes and used motion capture data to enhance the animation.

3. Path Generation

A motion path plays an important role in controlling all the motions of characters, that is, a guideline that indicates a direction and position a character should move. A path is made of Hermit-curve with a number of control points.

The environment and motion parameters are required to generate a path. A user can input the environment information such as roads and obstacles through user interface such as mouse, and assign start and end points of characters (fig 2).

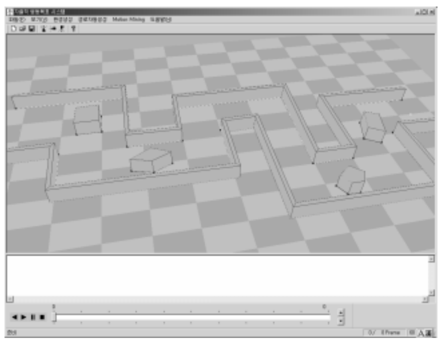


Fig 2. Create the environment – road, obstacles

A user is able to set the behavior parameter such as velocity, weight, status, and preference. The velocity parameter controls the moving velocity of a character (e.g. slow or fast). The velocity and weight values become larger, a radius of rotation of a character becomes larger. The status parameter determines whether the motion is large or small and is connected with multi-resolution filter [2]. The preference parameter is used for making decisions. For example, when applying a collision avoidance, whether the character turns to the left or the right, whether the character avoids obstacles far or near, or whether the character waits for the moving obstacle to go pass or not are determined by using the preference parameter.

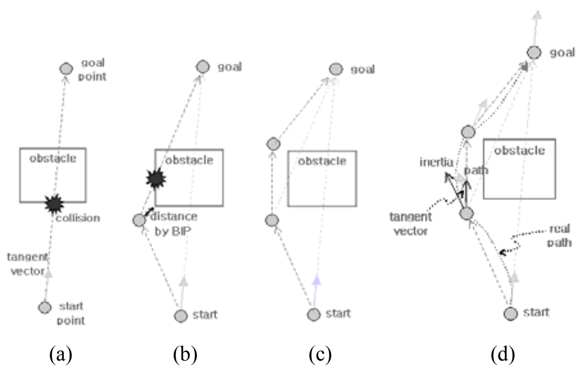


Fig 3. Curve Generation

Figure 3 shows the method that generates a path. First, the system checks a collision between start position and end position. If not exist, the path is the line that connects two points: otherwise, the system inserts the new control points (fig 3(b)(c)). When inserting a control point, the system refers the environment and the behaviour parameter as mentioned before. This process is repeated until a collision does not occur and the curve that is generated by inserted control points is the trajectory that a character follows. With the placed control points, tangent vector is determined at each control point as follow.

$$\mathbf{V}_t = (\text{ norm }(\mathbf{C}_{i+1} - \mathbf{C}_{i-1}) + \text{norm }(\mathbf{C}_i - \mathbf{C}_{i-1})\,) \times \nu \times \omega \quad (1)$$

($\mathbf{V}_t$: tangent vector, $\mathbf{C}_i$: ith control point, ν : velocity, ω : the weight of character)

4. Motion Generation

The system generates an appropriate motion following the generated path and applies parameters such as the velocity, motion status, and style at this time. It should create motion automatically using motion blending and transition techniques because motion database includes short motion clips only.

4.1 Motion Mining

We define motion mining as editing and selecting proper motions automatically from motion database that consists of a number of short motion clips such as walking, jogging, and running. If a user inputs motion status such as 'Walk' or 'Run' and its velocity then the system selects proper motion automatically from motion database. If motion database has no exact motion, motion-mining system generates a motion that is similar to motion status and velocity that a user inputted by blending a set of motion clips. For example, a user may input that motion status is 'walk' and velocity is 4m/s. Mining system searches the motion whose status is 'walk' and whose velocity is 4m/s from motion database. If the motion does not exist in the database, mining system makes a new motion from similar motions.

For searching appropriate motions, mining system must have motion information such as motion status (walk, run, and so on), velocity, motion style (brisk or gloomy), and constraints with ground. Especially, the constraint of a foot is important information and is obtained by calculating the moving distance of a foot from a previous frame to a current frame, that is, the foot becomes a constraint condition at a current frame if the moving distance of a foot or ankle joint is within threshold

$$\begin{gathered} d(\mathrm{p}_i, \mathrm{p}_{i-1}) = \| \mathrm{p}_i - \mathrm{p}_{i-1} \|^2 \\ \mathit{if}\,(d < \mathit{threshold}) \ \text{constraint} \end{gathered} \tag{2}$$

Here, p_i is a foot position at ith frame and if the moving distance between previous position and current position of foot is within threshold then the foot becomes constraint at the current frame. If two feet are on the ground simultaneously, then this motion status is 'walking', otherwise, this motion status comes under 'running'. Whether a character moves or not is estimated by a moving distance of the center of mass (COM). A character is regarded as standing if COM exists only within the pre-defined radius. The velocity of a character is obtained by a moving distance and elapsed time. We may presume that a motion style is brisk or gloomy using the standard deviation of joint values.

4.2 Motion Blending

We first perform time-warping process to synchronize the motion clips. We then calculate the weights of the motion clips. Next, we compute the target motion by blending them with respect to their weights. Finally, we adapt the blended posture to the trajectory.

We should address time warping problem to interpolate a set of motions because short motion clips are different with number of frames, starting posture and so on. We solved it by constraint based approach because the important constraint of locomotion such as walking, running has some connection with ground in general. To blend a set of motions, the system selects a frame that has a left foot constraint and extracts the cycle, that is, repeated section by a measure of posture similarity between frames, and rearranges the motion frames that the frame, which the right foot takes off the ground, becomes start frame. The generated clips are converted to those at generic time (fig4) .

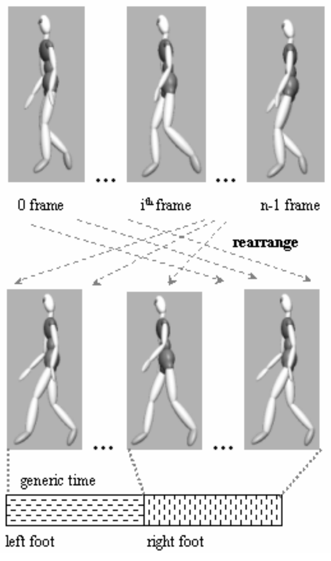


Fig 4. Time warping

We can generate a posture at a given time by blending the corresponding postures of a set of motions at the same generic time. We used a quaternion interpolation method, which transforms the orientation data into vector to compute their weighted sum, and then transforms the result into the orientation space. The main connection between unit quaternion and vector is the logarithm and exponential functions.

$$\mathbf{q}_k = \exp\left(\sum_i \log(\mathbf{q}_{i^*}{}^{-1} \mathbf{q}_{ik}) \otimes w_i \right) \quad (3)$$

$$\alpha_i = \left(\frac{1}{(v_w - v_i)} \right)^2, \ N = \sum_i \alpha_i, \ w_i = \alpha_i / N$$

Here w_i is a weight of the motion clip, v_w is desired velocity and v_i is ith character's velocity. α_i is square of inverse of this difference and w_i is obtained by dividing by N that total sum becomes one.

Posture similarity means how much one posture is similar to other posture. This equation is a similarity equation. We estimate they are very similar if the sum of this term is within threshold. Where coefficient delta is joint weight value.

$$d(p_i, p_j) = \sum_{k=0}^{n-1} \delta_k \, \| \log(q_{j,k}^{-1} q_{i,k}) \|^2 \quad (4)$$

Although we find the similar posture, a transition might be noticeable if the motion sequence simply jumps from one posture to another. Therefore, we interpolate a connection part using 'Qsquad' method that is spherical cubic interpolation of quaternion.

The directions of these motions use those of COM of a character. The direction vector of COM is changed in accordance with the tangential vector of the curve. And it is not until the total moving distance of a character is same with curve's length that motion generation process finishes (fig 5).

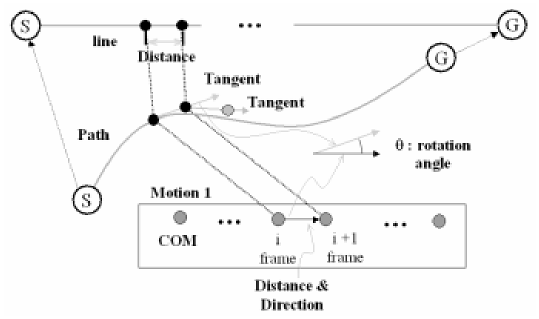
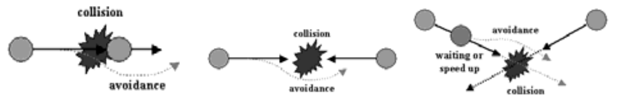


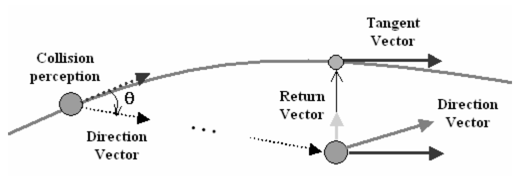
Fig 5. A distance and direction of a character

5. Collision Avoidance

The motion trajectory controls overall motion of a character and does not deal with collisions among characters. First, consider how we avoid collisions in our human society for collision avoidance. It is very complex but it can be defined by some simple rules. In general, one is reluctant to be far away from one's path. Therefore, one just goes ahead if the other goes out of way to avoid a head-on collision. In case of overtaking, one prefers a wide side or follows the other if obstacles (or other characters) exist somewhere near. In case that a collision occurs while proceeding to different direction, people pass on the backside or speed up; otherwise speed down or wait generally (fig 6-a).



(a)



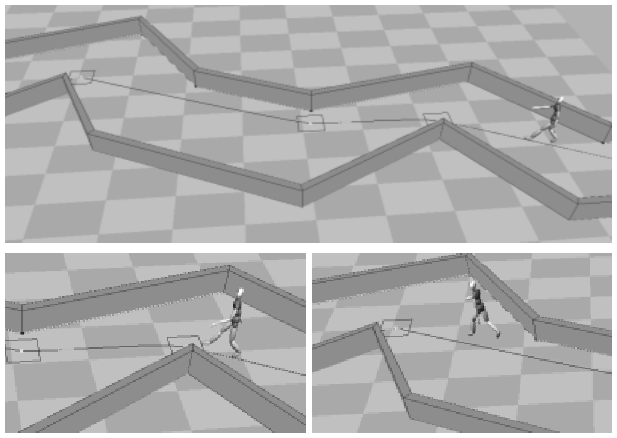
(b)

Fig 6. Collision avoidance

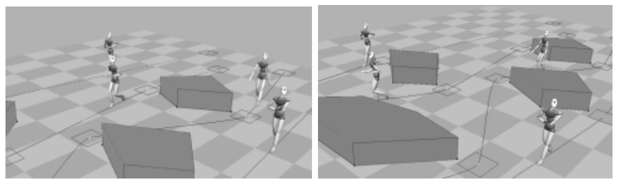
We define the window that may estimate the situation within next some frames using current information such as a direction, velocity, and acceleration. When a collision is predicted within a defined window, the system performs a slight adjustment to avoid a collision through preference parameter. If a collision situation is not dissolved, it tries the opposite side. In case of remaining a problem, a character stops for a while. A character returns to own path using a tangent vector of a trajectory and a direction vector toward a curve when avoiding a collision (fig 6-b).

6. Results

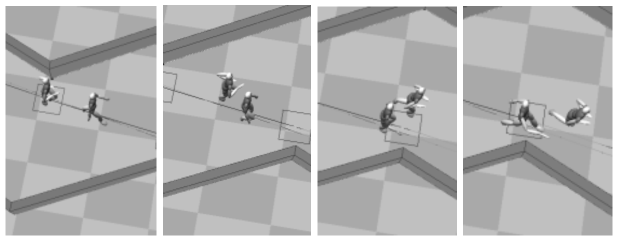
Figure 7 shows the characters following the paths. A character walks along the road as shown fig 7-(a). Figure 7-(b) presents that the characters moves while avoiding obstacles on the ground. Some characters run and others walk by motion status and velocity that a user input. A user can alter the velocity of a character at one's own accord and the character runs after walking or walks after running in accordance with the motion status. The number of motion frames and a moving distance of a character are determined by a given curve trajectory. Fig 7 (c) and (d) show collision situations. As show in figure, a character leaves own trajectory to avoid a collision with other character and then returns to a path.



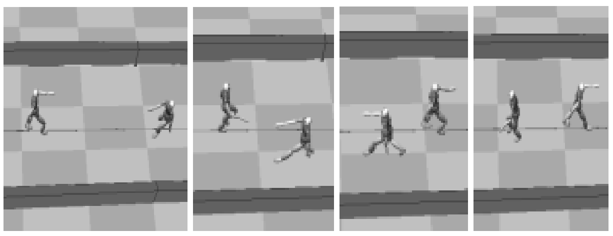
(a) A path following for one character



(b) Paths following without collisions



(c) Overtaking



(d) Face to face

Fig 7. Examples

We have presented a new approach for generating an appropriate path and motion automatically by using the environment and behavior parameters and motion mining system. We have showed a collision avoidance method that applies rules as humans do. This system has two merits; the motion of a character is similar to that of a human because the motion is based on the motion captured data and our system is fast and easy because the motion is controlled by high-level parameters. This system can be expanded to the crowd system because it is able to create many analogous motions with a few motions, apply to lots of characters, and avoid a collision naturally based on human rules.

References

1. A. Bruderlin and Calvert, T. W., "Goal-directed, dynamic animation of human walking", In Computer Graphics (Proceedings of SIGGRAPH 89), vol. 23, pp. 233-242.
2. A. Bruderlin and L. Williams, "Motion Signal Processing", In Proceeding of SIGGRAPGH 95, 1995, pp. 97-104
3. B. Ulicny and D. Thalmann., "Crowd simulation for interactive virtual environments and VR training systems", Proc. Eurographics Workshop on Animation and Simulation '01, 2001, pp. 163-170
4. C. Rose, B. Guenter, B. Bodenheimer, and M. F. Cohen., "Efficient generation of motion transitions using spacetime constraints", In Proceeding of SIGGRAPH 96, 1996, pp. 147-154.
5. C. Sulivan, J Cassell, H Vilhjalmsson, S. Dobbyn, C. peters, W. Leeson, T. Giang and J. Dingliana, "Crowd and Group Simulation with Levels of Detail for Geometry, Motion and Behaviour", *Third Irish Workshop on ComputerGraphics.* 2002.
6. D. Tolani and N.I. Badler., "Real-time inverse kinematics of the human arm", Presence, 5(4), 1996, pp. 393-401
7. F. Multon, L. France, C. Gascuel and G. Debunne., "Computer Animation of Human Walking: a Survey", The Journal of Visualization and Computer Animation volume 10, 1999, pp 39-54
8. J. Lee, J. Chai, P. S. A. Reitsma, J. K. Hodgins, and N. S. Pollard, "Interactive control of avatars animated with human motion data", *In Proceedings of ACM SIGGRAPH*, 2002, pp. 491-500
9. L. Kovar, M. Gleicher, and F. Pighin, "Motion Graph", In Proceeding of SIGGRAPH 02, 2002, pp. 473-482
10. J. Lee and S. Y. Shin., "A hierarchical approach to interactive motion editing for human-like figures", In Proceeding of SIGGRAPH 99, 1999, pp. 39-48.
11. M. Kallmann and D. Thalmann, "Modeling Behaviors of Interactive Objects for Real Time Virtual Environments", Journal of Visual Languages and Computing 13(2), 2002, pp. 177-195
12. N. Farenc, S. R. Musse, E. Schweiss, M. Kallmann, O. Anue, R. Boulic and D. Thalmann., "A Paradigm for Controlling Virtual Humans in Urban Environment Simulations", Applied Artificial Intelligence 14(1), 2000, pp. 69-91
13. Li, Y., Wang, T., and Shum, H–Y, "Motion texture: A two-level statistical model for character synthesis", In Proceedings of SIGGRAPH 02, 2002, pp. 465-472
14. Pullen, K. and Bregler, C., "Motion Capture assisted animation: Texturing and synthesis", In Proceedings of SIGGRAPH 02, 2002, pp. 501-508
15. S. Tak, O. Song, and H. Ko, "Motion Balancing Filtering", In Eurographics 00, 2000
16. Y. Kang, H. Cho and E. Lee, "An Efficient Control over Human Running Animation with Extension of Planar Hopper Model", The Journal of Visualization and Computer Animation, 10(4), 1999, pp. 215-224