Using Motion Capture Data to Optimize Procedural Animation

Chang-Hung Liang
Computer Science Department
National Chengchi University
g9606@cs.nccu.edu.tw

Tsai-Yen Li
Computer Science Department
National Chengchi University
li@nccu.edu.tw

1 Introduction

Procedural animation [1][3] has the advantage of being able to generate flexible animations according to high-level parameters while motion capture-based (MoCap-based) methods [2] have the advantage of creating realistic animations. In this paper, we attempt to use the data from motion capture to automatically find an optimal set of parameters that can be used to produce realistic animations for procedural animation. These sets of procedural parameters can then be interpolated to produce desired animations by taking into account the constraints in a specific scenario. The experimental results show that the proposed method can not only produce plausible animations but also accommodate environmental constraints by adjusting high-level procedural parameters.

2 System Overview

We model the problem of generating natural-looking motions with animation procedures as an optimization problem by comparing the procedure-generated motion with the motion-captured motion. We use a walking procedure as an example conduct our experiments. The walking motion is modeled with four keyframes. A total of 194 motion attributes are used to define the keyframes and inbetweens. Twenty six of them are defined for keyframes such as step length, step height, and arm swaying angle. The remaining parameters (168) are used to control the timing (ease in/out) of the motions between every two keyframes.

An overview of the system is shown in Figure 1. The auto-tuner module is the main component of the system used to search for the optimal set of motion parameters in the animation procedure for the creation of realistic motions. The objective function for the optimization process is defined as

$$f(A) = \sum_{i=1}^{n} d(P_A[i], M[i]), \tag{1}$$

where A is a given set of procedural parameters, $P_A[i]$ and $M[i]$ are the ith frame in the animations generated by the procedure and the MoCap data, respectively, and d is a distance function computing the weighted sum of the discrepancies on the corresponding joint locations between two motions.

We have used the Simulated Annealing (SA) algorithm to solve the optimization problem. The search space is defined by the 194 procedural parameters, each of which is bounded in a reasonable interval to confine the search space. The goal of the search is to find the optimal solution minimizing $f(A)$ of eq. (1). An example (left hip angle) of comparison between the motions generated by the procedure and by the MoCap data before and after the optimization is shown in Figure 2.

The objective of finding the optimal set of procedure parameters is to make the generated motion look as natural as the MoCap motions. Nevertheless, the main advantage of generating animations with procedures is on the flexibility of creating a good variety of motions according to the requirements of a given scenario. For example, the system should be able to generate the walking motion according to the step length and step height required in a specific scene. However, the parameters in the animation procedure are not likely to be independent. Varying only some parameters often results in an unnatural motion.

In order to account for the possible unnaturalness resulting from varying parameters, we propose to apply the method to many sample motions of the same type from the MoCap database and then interpolate the obtained parameter sets according to the given independent parameters such as step length and step height. Assume that the parameter set for walking is defined as $P_i(a_1, a_2, a_3, \ldots, a_{194})$, where i (=1..m) is the index of a sample motion in the

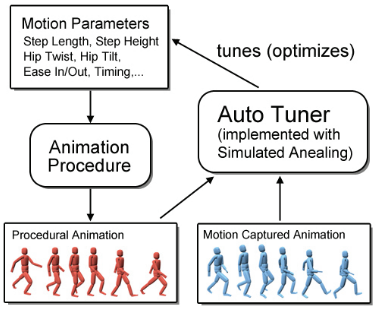
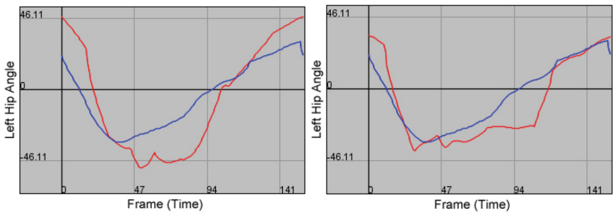


Figure 1. System overview



(a) (b)

Figure 2. Comparison of a joint angle (red: procedural, blue: MoCap) (a) before optimization and (b) after optimization

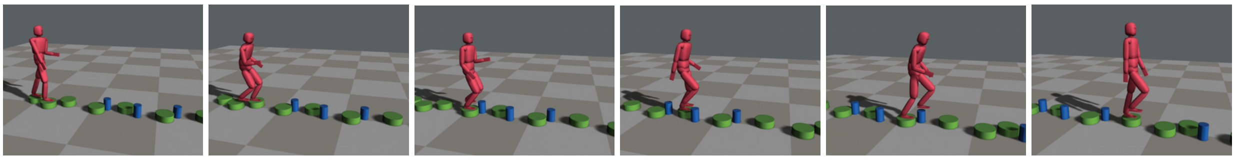


Figure 3. An example animation showing how adaptive motions can be generated according to the requirement of the environmental constraints

MoCap database for walking. Suppose that the parameters a_1 and a_2 represent step length and step height, respectively. We hope to compute the dependent parameters (a_3, … , a_{194}) by interpolating on the given independent parameters (a_1, a_2). In order to find the closest samples of a given query point for interpolation, we first perform Delaunay Triangulation on the sample points. Then we find the triangle where the query point is located and use the three vertices of the triangle to linearly interpolate the dependent parameters.

In Figure 3, we show an example of walking motion generated by the procedure with the need of using different step lengths and step heights along the path to satisfy the environmental constraints. When the desired step lengths and step heights are determined in a given scenario, the system computes the remaining parameters according to this specification by interpolating nearby sample points.

3 CONCLUSIONS

Different approaches to the generation of character animations have their own pros and cons. In this work, we attempt to combine the advantages of the MoCap-based approach and the procedure-based approach. We use MoCap data to enhance the realism of the motions generated by procedural animation in an automatic fashion. The enhancement is modeled as an optimization problem on the animation parameters. The obtained parameters for different motion samples in a MoCap database can be further interpolated to obtain the desired procedural parameters for a target specification. The tedious and time-consuming parameter tuning in the design of an animation procedure can therefore be avoided while the flexibility of procedural animation is kept.

REFERENCES

[1] Bruderlin, A. and Calvert, T.W. (1996). Knowledge-Driven Interactive Animation of Human Running. *Proceedings of the Conference on Graphics Interface 1996*, pages 213-221, 1996.

[2] Kovar, L., Glercher, M., and Pighin. F. (2002). Motion Graphs. *Proceedings of ACM SIGGRAPH 2002*, pages 473-482, 2002.

[3] Perlin, K. (2003). Building Virtual Actors Who Can Really Act. *Proceedings of the 2nd International Conference on Virtual Storytelling*, pages 127-134, 2003.