Model-Based Human Motion Capture from Monocular Video Sequences

Jihun Park[1], Sangho Park[2], and J.K. Aggarwal[2]

[1] Department of Computer Engineering
Hongik University
Seoul, Korea
jhpark@hongik.ac.kr
[2] Department of Electrical and Computer Engineering
The University of Texas at Austin
Austin, TX 78712
{sh.park,aggarwaljk}@mail.utexas.edu

Abstract. Generating motion and capturing motion of an articulated body for computer animation is an expensive and time-consuming task. Conventionally, animators manually generate intermediate frames between key frames, but this task is very labor-intensive. This paper presents a model-based singularity-free automatic-initialization approach to capturing human motion from widely-available, static background monocular video sequences. A 3D human body model is built and projected on a 2D projection plane to find the best fit with the foreground image silhouette. We convert the human motion capture problem into two types of parameter optimization problems: static optimization and dynamic optimization. First, we determine each model body configuration using static optimizations for every input image. Then, to obtain better description of motion, the results from all static optimizations are fed into a dynamic optimization process where the entire sequence of motion is considered for the user-specified motion. The user-specified motion is defined by the user and the final form of the motion they want. A cost function for static optimization is used to estimate the degree of overlapping between the foreground input image silhouette and a projected 3D model body silhouette. The overlapping is computed using computational geometry by converting a set of pixels from the image domain to a polygon in the real projection plane domain. A cost function for dynamic optimization is the user-specified motion based on the static optimization results as well as image fitting. Our method is used to capture various human motions: walking, pushing, kicking, and handshaking.

1 Introduction and Previous Work

Generating articulated body motion for computer animation is time consuming and computationally expensive. Because an articulated body has many degree of freedom (DOF), its computation is nonlinear, which makes motion generation

A. Yazici and C. Şener (Eds.): ISCIS 2003, LNCS 2869, pp. 405–412, 2003.

difficult. Probably the simplest form of computer-based animation is *key frame animation*, in which the animator denotes the key positions of the body at both the initial and goal configurations. The animation system then smoothly interpolates the intermediate positions between these key positions. In this paper, we present a cost-effective method of motion capture for computer animation using widely-available monocular video sequences with a static background.

Tracking non-rigid objects such as moving humans presents problems, including segmentation of the human body into meaningful body parts, handling occlusion, and tracking the body parts over the image sequence as well as automatic tracking of the first image frame. Approaches for tracking a human body can be classified into two groups: model-based approaches and view-based approaches[1,2]. (Park et al.[3] combined view-based and model-based techniques by processing color video at both pixel and object level.) We present here a model-based singularity-free automatic-initialization approach to capturing human motion that uses computational geometry for model fitting computation. One of the unique contributions of this paper is the ability to automatically track the initial frame[4].

We define animation to be *motion* with *motion control*. This motion can be either computed via *simulation*[5,6] or captured using a motion capture device. We require *motion control* to satisfy kinematic goals/constraints; usually for animation this involves meeting the configuration goals of a body while satisfying constraints. If we want to do *animation*, we must be able to do *motion control*. Basically we need a tool that will consider the entire motion (from the beginning of a motion until the end) to help us to find the *best* motion that suits our animation purpose. In order to do this, we convert our animation problem into a single nonlinear cost function of *(control) variables* with nonlinear equality and inequality constraints. By finding the best values of these variables, we can find the best motion for the animation. The technique of finding local optimum values for a nonlinear cost function with nonlinear equality and inequality constraints is called *parameter optimization*. Our proposed method processes motion capture twice. At the first level, we get the proper model body configuration for an input image. There is no time concept involved in this process, which is called *static optimization*. The body configuration must be interpolated to get motion for computer animation. But the motion we get from image motion capture is not a suitable character motion that satisfies user (animator) motion specifications. For this purpose, we need to modify the input data such that the resulting motion satisfies the user-specified motion. This second-level process considers the entire motion in animation, and is called *dynamic optimization*. Witkin and Kass[5] and Park and Fussell[6] both take user input of initial motion guessing and find the user-specified motion, but use inverse dynamics or forward dynamics, respectively, to improve the quality of the animated motion. The dynamic optimization approach presented here is quite similar; however our approach takes initial motion input using static optimization, and the quality of the entire motion is supported by the image fitting, whereas the others[5,6] are based on

simulation. In order to find the motion that satisfies the user-specific constraint, the degree of image fitting is sacrificed.

All kinematics-based motion tracking methods may be classified into two groups depending the use of inverse kinematics. If no inverse kinematics is used, we call that approach a "forward kinematics-based approach"; otherwise we call it an "inverse kinematics-based approach". Our work is forward kinematics-based, while the paper [7] is inverse kinematics-based. Singularity is inevitable for methods with differential inverse kinematics.

2 Human Body Modeling and Overview of Our System

As shown in figure 1(a), the body is modeled as a configuration of nine cylinders and one sphere. These are projected onto a 2D real projection plane. A sphere represents the head, while the rest of the model body is modeled using cylinders of various radii and lengths. Currently, we use only nine 1-DOF (1 degree-of-freedom) joints plus body displacement DOF, and a vertical rotation to compensate for the camera view. These are our control variables for the cost function. Figure 1(b) shows the initial state of searching for the best overlapping configuration, given the first frame image of video sequence of figure 4. As can be seen in figure 1(b), initial joint angle values of the model body for parameter

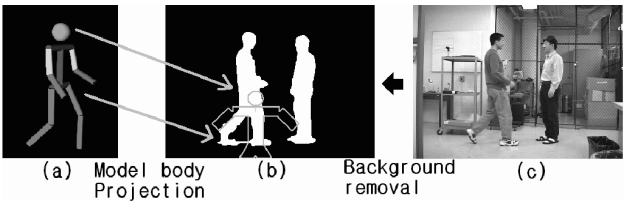
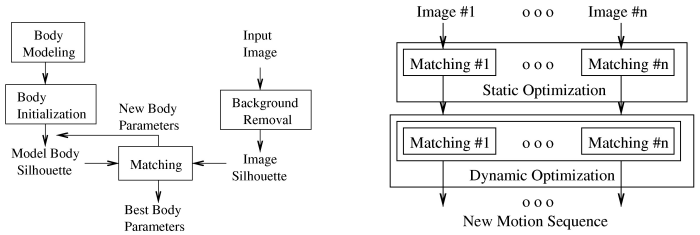


Fig. 1. 3D Model body (a), overlapping between background removed image and projected body (b), and original input image(c).



(a) Computation of a single image fitting. (single static optimization.)

(b) Computation considering all image fittings. (dynamic optimization)

Fig. 2. (a) Process of determining the best fitting model body configuration given an image, (b) finding the best motion considering all image fittings.

optimization are arbitrary. This shows that our initial model body configuration detection for the first image frame is automatic, which is a great improvement over other methods[4].

Figure 2(a) shows a fitting (matching) process (i.e. computing the overlapping between the foreground input image and projected model body silhouette) given an input image. A 3D model body is built. Model body joint angles and displacements are determined, and the body is projected on a 2D projection plane. On the 2D projection plane, we get a model body silhouette. The boxes in figure 2 represent computational processes. The *fitting* process uses parameter optimization [8], which modifies the model body parameters and checks the resulting level of fitting between the image silhouette and the model body silhouette. When the best fitting model body configuration is found for a single image, then the process is done for that image; thus for n input images, we run the fitting computation n times. Figure 2(b) shows the sequence of fitting process tasks. When the fitting computation is completed, we have a model body configuration for each image. Then we have a complete motion sequence for the input images. Static optimization determines the best body configuration for a single input image, while dynamic optimization finds the best motion for a sequence of input images.

3 Background Subtraction

A pixel color at an image location is represented as a 3D vector $(R, G, B)^T$ in the RGB color space. A background model is built for every pixel using 30 background images that are captured when no person is contained in the camera view. In order to handle shadows effectively, we used normalized RGB models. Average background color vectors for image location are computed. Vector angles between the average background color vector and a given color vector to be determined are computed to determine if they are part of the foreground or background. A foreground image is obtained by thresholding and applying morphological filters to remove small regions of noise pixels.

4 Forward Kinematics-Based Cost Function for Parameter Optimization

In this section, we present our cost function used in the overlapping area computation for both static optimization and dynamic optimization. A projected model body is assumed to have an affine transformation[7]. This is mathematically acceptable if an orthographic camera model is used and the model body moves parallel to the projection plane. Given a set of joint angles and body displacement values, the forward kinematics function computes points of body segment boundaries. The projection matrix then projects the computed boundary points on a 2D projection plane, which will be compared to a foreground image silhouette. The overlapping area computation in the real number domain

makes the derivative-based parameter optimization possible. Refer to the authors' papers [3,9] for more detailed explanation. The differences are that we do not use body part information as well as distance maps. Because the input is a vector of joint DOF values, this computation is purely forward kinematics-based and thus presents no singularity problem[3,9].

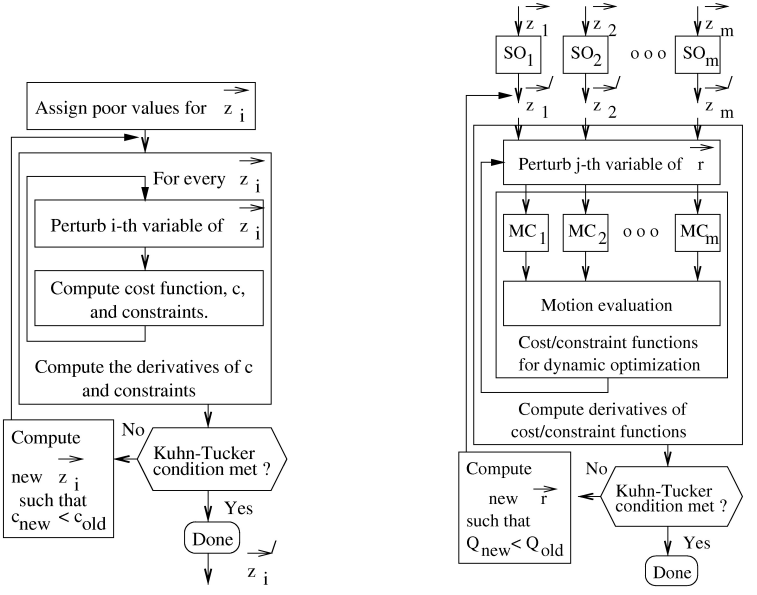
We compute the image overlapping area using a computational geometry-based method on each pixel. An image silhouette is one that is converted from the 2D integer pixel domain to a real domain such that the resulting image silhouette becomes a jagged-edge polygon with only horizontal and vertical edges. Projected objects are either polygon-shaped or circular. The resulting polygon(s) may even have holes. We compute the polygon intersection between the image silhouette and the model silhouette. Our computation is a modified version of Weiler-Atherton's polygon intersection/union computation [10]. We found the best overlapping configuration using the GRG2 [8] optimization package with the cost function discussed above.

5 Parameter Optimization

A parameter optimization problem is of the following form: minimize a nonlinear cost function $c(\bar{z})$ subject to $g_i(\cdot) = 0, 1 \leq i \leq l, q_j(\cdot) \leq 0, 1 \leq j \leq m$ where l is the number of equality constraints, m is the number of inequality constraints, and $g(\cdot)$ and $q(\cdot)$ are nonlinear constraint functions, where $\cdot$ is a generic variable(s), The parameter optimization problem solver we used is GRG2[8]. Nonlinear functions usually have many local optima as well as ridges.

5.1 Static Optimization

Given an image, we preprocess to get a foreground image silhouette. Then, to find the best model body configuration for the image silhouette, we solve parameter optimization. In solving the parameter optimization, we define a cost function that computes the degree of the overlapping between the foreground input silhouette and a projected 3D model body silhouette. We call this process static optimization. Figure 3(a) shows the processing flow of a static optimization for the *i-th* image. The input variable vector $\bar{z}_i$ consists of body DOF variables, body displacement as well as joint angles. Because we have n body DOF variables, we have n variables for static optimization. For m input images, we run the static optimization m times. We initialize $\bar{z}_i$ with poor (arbitrary) input values. Figure 1(b) shows the initial search state for the best overlapping configuration, given arbitrary input values. For every control variable, the static optimization perturbs each variable to compute derivatives of constraint functions and the cost function and then checks for the Kuhn-Tucker condition[8], which guarantees a local minimum while satisfying constraints.



(a) Static optimization. (b) Dynamic optimization.

Fig. 3. Static and dynamic optimization.

5.2 Dynamic Optimization

To find a motion over an entire sequence of input video sequences, we need dynamic (global) optimization. We use the term *dynamic* to mean considering the entire image sequence. Figure 3(b) shows the process of dynamic optimization. SO_i is static optimization process for *i-th* image. Because we have m images, we run static optimization m times. Each static optimization has n variables. For m input images, we have total $n \times m$ variables for dynamic optimization. Static optimization results, $z_i^{'}$, for all image sequences become the initial guessing values for dynamic optimization. Figure 4 shows the subject with the model figure superimposed for a sequence containing a pushing motion. We used overlapping area computation as a cost function for static optimization, shown in figure 5(a-c). The example of a cost function of dynamic optimization presented in figure 5(d-f) includes non-violent motion evaluation as well as the degree of overlapping between image and model silhouettes. (By *violent motion* we mean motion with large absolute velocity/acceleration values.) MC_i is a process of computing the degree of overlapping between the model silhouette and the *i-th* image silhouette. The computation results we get from every MC_i are interpolated using C^2 continuous Natural Cubic Spline. Then, the motion evaluation process evaluates the model figure motion. The optimizer will find the best solution, $\bar{r}^{'}$, a collection of the model body configurations for all image silhouettes.

Fig. 4. The subject with the model figure superimposed, shown over a pushing motion.

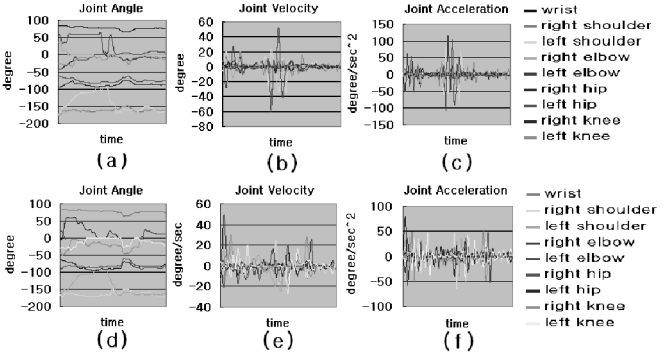


Fig. 5. Joint angles, velocities, and accelerations of the pushing motion. Results from static optimization(a,b,c), and results from dynamic optimization(d,e,f).

6 Experimental Results

The 2D-based human motions we have studieded include pushing, kicking, walking and handshaking. The example shown in figure 4 has a static background with one person (the left) pushing another (right) person away. The white line shows the result of union of every model body part. The holes in the resulting body (the polygon-like object) indicate our geometry union works well. As long as there is no heavy occlusion between the subjects, motion tracking is satisfactory. Using joint angular values from each frame, we perform motion interpolation using the Natural Cubic Spline. These velocity and acceleration values are based on the motion interpolation. Despite the presence of violent motion in the middle of the sequence, as can be seen in the graphs, the tracking is excellent. The violent motion of the left shoulder and left elbow joint can be observed from the velocity and acceleration graphs of Figure 5(b,c). Figure 5(d-f) shows the same motion solved using dynamic optimization. The results of static optimization are used as the input values for dynamic optimization. The part of the cost function is to reduce any violent velocity and acceleration in the static optimization-based motion. As you can see, the violent motion is quite reduced. Figure 6 shows walking (departing), shaking hands, and kicking motions, respectively.

Fig. 6. The subject with the model figure superimposed, shown over (a) a walking motion, (b) a hand-shaking motion, and (c) a kicking motion.

Acknowledgements. This work was partially supported by grant No. 2000-2-30400-011-1 from the Korea Science and Engineering Foundation. We thank Ms. Debi Prather.

References

1. Aggarwal, J., Cai, Q.: Human motion analysis: a review. Computer Vision and Image Understanding **73** (1999) 295–304
2. Gavrila, D.: The visual analysis of human movement: a survey. Computer Vision and Image Understanding **73** (1999) 82–98
3. Park, J., Park, S., Aggarwal, J.K.: Human motion tracking by combining view-based and model-based methods for monocular video sequences. Lecture Notes in Computer Science (2003 International Conference on Computational Science and Its Applications) **2669** (2003)
4. Bregler, C., Malik, J.: Tracking people with twists and exponential maps. In: IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara, California (1998) 8–15
5. Witkin, A., Kass, M.: Spacetime constraints. Computer Graphics **22** (1988) 159–168
6. Park, J., Fussell, D.: Forward dynamics based realistic animation of rigid bodies. Computers and Graphics **21** (1997)
7. Huang, Y., Huang, T.S.: Model-based human body tracking. In: International Conference on Pattern Recognition. (2002)
8. Lasdon, L., Fox, R., Ratner, M.: Nonlinear Optimization using the Generalized Reduced Gradient Method. Department of Operations Research 325, Case Western Reserve University (1973)
9. Park, S., Park, J., Aggarwal, J.K.: Video retrieval of human interactions using model-based motion tracking and multi-layer finite state automata. Lecture Notes in Computer Science (2003 Intl. Conf. on Image and Video Retrieval) **2728** (2003)
10. Hill, F.: Computer Graphics. Macmillan (1990)