

Animating Character Images in 3D Space

Bing-Yu Chen*
National Taiwan University

Shih-Chiang Dai†
National Taiwan University

Shuen-Huei Guan‡
Digimax

Tomoyuki Nishita§
The University of Tokyo

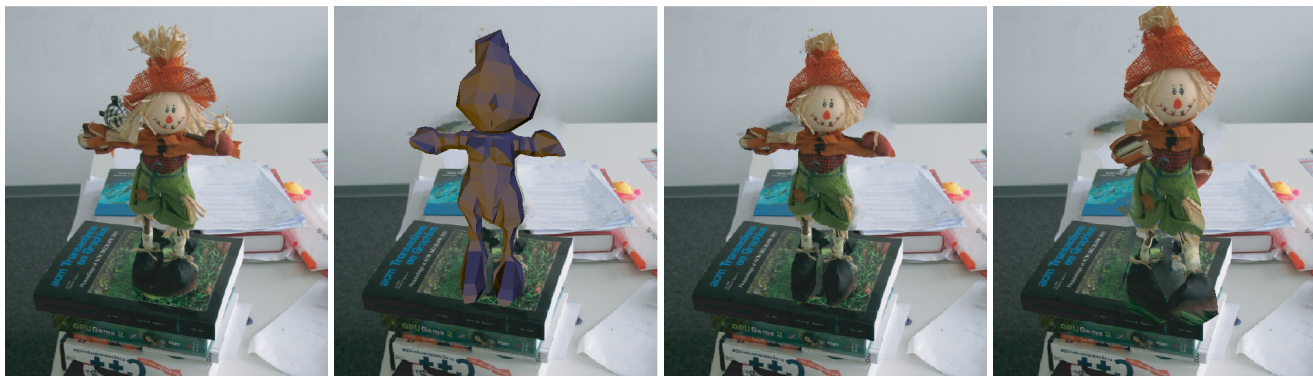


Figure 1: From left to right: the original input image; the template character model after fitting to the character image; the character model textured by the character image; the motion data is applied to animate the character model.

1 Introduction

In this extended abstract, we present a system that allows the user to animate character images in 3D space by applying an existed 3D character model with motion data. The character model with skeleton rigged is used as a template model to fit the silhouette of the character image. After assigning some corresponding points between the character image and template model, the system then fits the model to the image and transfer the colors and patterns of the image to the model as the textures. Finally, the user can apply any motion data to animate the fitted 3D character model in 3D space.

2 Overview

The input of our system is an image and a skeleton rigged 3D character model and a 2D character image. The system first samples some points on the contour of the character image as the vertices in 3D space, and the points form a close loop as a "contour loop" C . Then, the system roughly generates silhouette vertices from the 3D character model, which also form a close loop as a "silhouette loop" S . Then, our model fitting algorithm is applied to fit S to C while considering the original shape of the 3D character model. Finally, we can apply any motion data to the 3D character model or add any visual effect in 3D space.

3 Model Fitting Algorithm

Given a contour loop C extracted from the input character image I and a silhouette loop $S = \{s_i | i = 1, \dots, n\}$, $s_n = s_1$ with n silhouette vertices of the 3D character model T , we have to match S with C . The user first drag $m \leq n$ vertices of the silhouette loop $S' = \{s_{p(j)} | j = 1, \dots, m\} \subseteq S$, $s_{p(m)} = s_{p(1)} = s_1$ to their corresponding points of C manually, where $p(j) = i$ denotes an index mapping from the dragged vertices $s_{p(j)}$ to the silhouette vertices s_i , and $s_{p(1)} = s_1$. Then, the system fits the remaining $n - m$ silhouette vertices to C automatically while satisfying the constraint: $\overline{s_i s_{i+1}} = \overline{s_{p(j)} s_{p(j+1)}} / (p(j+1) - p(j))$, $p(j) \leq i \leq p(j+1)$.

After fitting S to C , we have to deform the shape of T to match that of I while satisfying the constraint (i.e., S'). We can first keep the

z coordinate fixed and only consider the 3D model T as a 2D mesh T_{xy} . Then, as-rigid-as-possible shape manipulation [Igarashi et al. 2005] is used to deform T_{xy} with constrained S' . Through a sparse linear solver, we can fit the remaining vertices of T_{xy} within a sec. After fitting the skin (only $x - y$ coordinates) of T , we still have to fit the original skeleton of T corresponding to the fitted skin.

Before skeleton fitting, we first project T and its skeleton joints onto $x - y$ plane (i.e., T_{xy}), and record each joint position by barycentric coordinate of the triangle which contains the joint. If there exists several triangles contain the same joint, we choose the one nearest to the joint in the original 3D space and belongs to that bone.

After fitting the skin and skeleton, we have transformed T to fit I by adjusting its projected $x - y$ coordinates in 2D space (i.e., T_{xy}). However, the quantity of the third (z) coordinate (or thickness) must also be revised to generate a convincing fitted 3D character model. In our experience, we observed that the distance between the bone and the skin is highly correlated to the average distance between the silhouette vertices $s_i \in S$ and the bone $b_k \in B$ they belong to. Hence, for each b_k we record the average distance d_k to its nearest silhouette loop S before the model fitting. If the vertex s_i belongs to several bones, we compute the average with its bone weight ω_{ik} as $d_k = \sum_i^{n-1} \omega_{ik} d_{ik} / (n - 1)$, $d_{ik} = \|s_i - b_k\|$, where n is the number of $s_i \in S$ and $\omega_{ik} = 0$ if s_i and b_k have no binding relationship. After skin fitting, we can then compute the new average distance d'_k by using the new position of s_i . Then, the new z value of each vertex is scaled by the ratio: $\sum_k \omega_{vk} (d'_k / d_k)$, where ω_{vk} is the binding weight between the vertex $v \in T$ and the bone b_k .

4 Result

Our system is implemented in C++ with OpenGL, and the character deformation is performed by following the standard linear blend skinning (LBS) method. The motion data is downloaded from <http://mocap.cs.cmu.edu/>. Fig. 1 shows the result. The computation time is interactive except inpainting and silhouette cut out. The user interaction time is about 10 min. for a trained user.

References

IGARASHI, T., MOSCOVICH, T., AND HUGHES, J. F. 2005. As-rigid-as-possible shape manipulation. *ACM Trans. Gr.* 24, 3, 1134–1141. (SIGGRAPH 2005 Conf. Proc.).

*e-mail: robin@ntu.edu.tw

†e-mail: jeffrey@cmlab.csie.ntu.edu.tw

‡e-mail: drake.guan@gmail.com

§e-mail: nis@is.s.u.tokyo.ac.jp