21st International Conference on Pattern Recognition (ICPR 2012)
November 11-15, 2012. Tsukuba, Japan

A Puppet Interface for the Development of an Intuitive Computer Animation System

Narukawa Hiroki
The University of Tokyo
narukawa@itl.t.u-tokyo.ac.jp

Natapon Pantuwong
The University of Tokyo
na@itl.t.u-tokyo.ac.jp

Masanori Sugimoto
The University of Tokyo
sugi@itl.t.u-tokyo.ac.jp

Abstract

In this paper, we introduce a puppet interface for the development of an intuitive animation system. Our puppet interface does not require any special devices and any type of puppet can be used. The puppet interface is developed by attaching ten visible markers onto a puppet. The user can manipulate the pose of the puppet interface to produce the desired motion in front of a camera. The puppet's joint angle information is captured by the camera and used to retrieve a suitable motion from a motion database. The retrieved motion data is transferred to a 3D character model to generate an animation. To avoid the occlusion problem, we propose an algorithm that estimates the joint angle by determining the position and rotation of markers adjacent to the occluded marker. Experiments confirmed that our proposed algorithm can generate correct results, although some occluded markers remained.

1. Introduction

未練なく散も桜はさくら哉 Several methods have been proposed to reduce the difficulty of the computer animation creation process and one of the most widely used strategies is the reuse of motion data. Prerecorded motion data can be transferred to a 3D model to control an animation using a motion retargeting algorithm [3]. Recently, several public libraries of motion capture data have been made available [6], which facilitate the reuse of motion data. However, the large volume of motion data that is available makes it difficult to identify an appropriate motion sequence in a database.

In this paper, we introduce a puppet interface that can be used for the development of an intuitive computer animation system. In contrast with previous work [10], our puppet interface does not require special devices such as orientation sensors. We develop a vision-based system for use with a puppet and we attached ten visible ARToolKit [5] markers, as shown in Figure 1. Any puppet can be used as the interface. The ARToolKit markers allowed us to capture rotation information from each of the puppet's joints using only one camera. A user can pose the puppet to generate the requisite motion while the application identifies the motion data based on the user's input. The animation can then be rendered after the retrieved motion data is retargeted to the user-selected 3D model. A vision-based puppet interface has been proposed before [2], but this earlier system did not overcome the occlusion problem. Our algorithm overcomes this problem by determining the position and rotation of markers adjacent to the occluded marker, before estimating rotation information for the occluded marker. We implemented a motion retrieval algorithm and applied it to the computer animation system. The user input could be either keyframe information or continuous frame information.

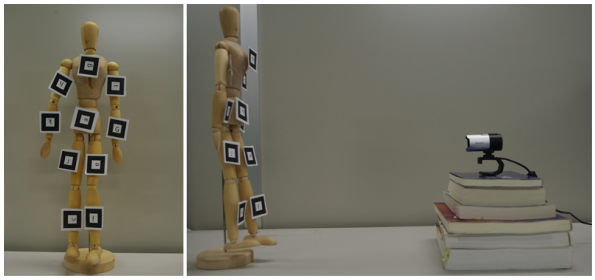

Figure 1: The puppet with ARToolKit markers attached.

2. Related Work

Several interfaces have been proposed for reducing the difficulty of searching for suitable motion data in motion databases, including sketch-based, performance-based, and tangible interfaces.

[9] used a sketch-based interface that allowed users to create several key poses with 2D sketches. These key poses were determined from keyframes, which were interpolated to produce a motion clip. However, this interface may not be suitable for novice users because it requires drawing skills.

Performance-based interface systems allow users to perform their desired motion and create a motion clip. User motions can be recorded using several different devices such as a camera [8] or pressure sensors [12]. This is a straightforward strategy, but it requires a large amount of physical effort by the user during the performance of the desired motion. Some motions are also difficult for users to perform, such as rolling in the air while jumping.

Tangible interfaces might be more suitable because they do not require any special skills or physical effort, and puppet-shaped devices are commonly used for this type of interface. Several researchers have developed puppet interfaces with different types of sensors [10]. These devices provide reliable results, but they also require special devices. Thus, an alternative strategy is to attach trackers or markers to a physical puppet. [1] attached magnetic trackers to the head and limbs of a wooden mannequin and joint rotation information could be acquired by mapping pose data from the animated character using inverse kinematic techniques. However, this strategy also requires special devices with a complicated setup process, which may not be suitable for novice users. [2] attached colored markers to all the important joints on a doll and the rotation information from each joint was retrieved using a 3D reconstruction technique. This technique required two cameras to capture the doll's pose and it required careful system setup. However, the results could be unreliable if some markers were occluded. Thus, this type of system is unsuitable for continuous frame-based input, because of the marker occlusion problem.

3. The proposed puppet interface

The proposed puppet interface comprised a puppet with visible markers attached. We used ARToolKit markers in our implementation. Each ARToolKit marker was attached to the puppet as shown in Figure 2. ARToolKit markers provided the transformation matrix for each marker, which could be used to calculate the rotation angle of each joint. For example, Figure 2 shows the transformation matrix of the 6^{th} marker, which can be used to calculate the rotation angle of the right shoulder joint. The 5^{th} marker was attached to calculate the rotation angle of the puppet's spine. We considered the position and rotation information of the root marker to determine the global position and rotation of the puppet.

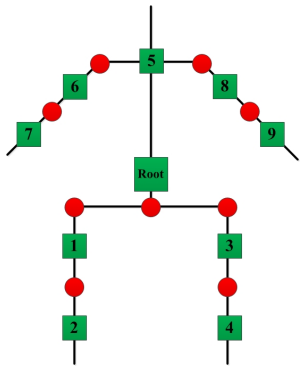

Figure 2: Position of each marker attached to the puppet. Circles indicate the positions of joints while squares indicate the positions of the markers.

We assume that the transformation matrix of the i^{th} marker is $\mathbf{M}_i$ while the transformation matrix of its parent marker is $\mathbf{M}_{parent(i)}$. These transformation matrices are provided in the ARToolKit library. The rotation matrix, $\mathbf{X}_i$, of a marker can be calculated using the following equation.

$$\mathbf{X}_i = \mathbf{M}_i \mathbf{M}_{parent(i)}^{-1} \tag{1}$$

The parent of the root marker is the root marker itself, so $\mathbf{X}_{root} = \mathbf{M}_{root}$.

The rotation matrix, $\mathbf{X}_i$, expresses the rotation information for each marker in terms of the Euler angle in YXZ order. Let $(\mathbf{X}_i)_{rc}$ be the value of the $\mathbf{X}_i$ matrix in the r^{th} row and the c^{th} column. We can extract the rotation of the x-, y-, and z-axes from this matrix using the following equations.

$$\theta_z = \sin^{-1}(\mathbf{X}_i)_{21} \tag{2}$$

$$\theta_x = \tan^{-1}\frac{(\mathbf{X}_i)_{23}}{(\mathbf{X}_i)_{22}} \tag{3}$$

$$\theta_y = \begin{cases} \frac{\sin^{-1}(\mathbf{X}_i)_{31}}{\cos\theta_z} & \text{if } (\mathbf{X}_i)_{11} \geq 0 \\ -\frac{\sin^{-1}(\mathbf{X}_i)_{31}}{\cos\theta_z} & \text{if } (\mathbf{X}_i)_{11} < 0 \end{cases} \tag{4}$$

If $\cos\theta_z = 0$, θ_x and θ_y are derived using the following equations.

$$\theta_y = 0 \tag{5}$$

$$\theta_x = \tan^{-1}\frac{(\mathbf{X}_i)_{32}}{(\mathbf{X}_i)_{33}} \tag{6}$$

The equations above cannot be used when some markers are occluded. Thus, we propose an algorithm

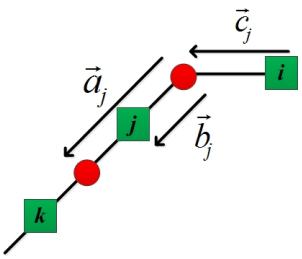

Figure 3: Three vectors are used for rotation matrix estimation when a marker is occluded.

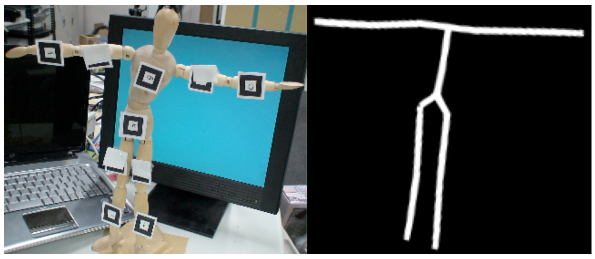

Figure 4: The proposed algorithm can estimate the rotations of all the occluded markers correctly.

that estimates the rotation information of the occluded marker based on information from its adjacent markers. Three vectors are used by the calculation. Vector $\overrightarrow{a}$ is the vector from the joint under consideration to its child. Vector $\overrightarrow{b}$ is the vector from the joint under consideration to its marker. Vector $\overrightarrow{c}$ is the vector from the parent joint of the joint under consideration. Figure 3 shows these three vectors for the joint associated with the j^{th} marker. We calculate these three vectors for each joint that is detected. The distance between joints is also important information, because it allows us to estimate an occluded marker. This distance information is measured manually in the current implementation. However, we plan to automate this process with an automatic rigging algorithm [11], which will calculate a skeleton structure appropriate to the input 3D model automatically. We can use a 3D scanner system to acquire a 3D model of the physical puppet. The automatic rigging algorithm can then be used to extract the skeleton structure of the puppet. The distances between the joints can then be calculated from the skeleton structure.

Assume that j^{th} marker in Figure 3 is occluded. Let the i^{th} marker be the parent of the j^{th} marker, while the j^{th} marker is the parent of the k^{th} marker. Let $\overrightarrow{m}$ be the position of a marker detected by the ARToolKit. We can then estimate the rotation matrix for this marker using the following calculation.

If the DOF of the j^{th} marker is 1 and the DOF of the k^{th} marker is 3, the rotation matrix of the j^{th} marker, $\mathbf{X}_j$, can be calculated using the following equation.

$$\mathbf{X}_j(\overrightarrow{a_j}+\overrightarrow{c_k}) = \mathbf{M}_i^{-1}(\overrightarrow{m_k}-\overrightarrow{m_i}-\mathbf{M}_i(\overrightarrow{a_i}-\overrightarrow{b_i}+\overrightarrow{c_j})-\mathbf{M}_k\overrightarrow{b_k}) \quad (7)$$

Equations 3 and 4 can be used to extract the rotation angles from the rotation matrix, $\mathbf{X}_j$.

If the DOF of the j^{th} marker is 3 and the DOF of the k^{th} marker is 1, we can calculate the rotation matrix of the k^{th} marker, $\mathbf{X}_k$, using the following equation.

$$\mathbf{X}_k(\overrightarrow{m_k}-\overrightarrow{m_i}-\mathbf{M}_i(\overrightarrow{a_i}-\overrightarrow{b_i}+\overrightarrow{c_j})-\mathbf{M}_k\overrightarrow{b_k}) = \mathbf{M}_k(\overrightarrow{a_j}+\overrightarrow{c_k}) \quad (8)$$

Thus, we can calculate the rotation matrix of the j^{th} marker, $\mathbf{X}_j$, by substituting $\mathbf{X}_k$ in Equation 1 ($\mathbf{X}_k = \mathbf{M}_k\mathbf{M}_j^{-1}, \mathbf{X}_j = \mathbf{M}_j\mathbf{M}_i^{-1}$) and the rotation angles can be extracted by solving Equations 3 and 4.

If the $root$ marker is occluded, we can calculate the transformation matrix of this marker, $\mathbf{M}_{root}$, using the 1^{st} and 5^{th} markers in Figure 2. First, we calculate the rotation matrix of the 5^{th} marker, $\mathbf{X}_5$, using the following equation.

$$\mathbf{X}_5(\overrightarrow{m_5} - \overrightarrow{m_1} - \mathbf{M}_1\overrightarrow{b_1} - \mathbf{M}_5\overrightarrow{b_5}) = \mathbf{M}_1(\overrightarrow{c_5} - \overrightarrow{c_1}) \quad (9)$$

The transformation matrix of the $root$ marker, $\mathbf{M}_{root}$, can be obtained by solving Equation 1 ($\mathbf{X}_5 = \mathbf{M}_5\mathbf{X}_{root}^{-1}$). The position of the $root$ marker, $\overrightarrow{m_{root}}$, can be calculated using the following equation.

$$\overrightarrow{m_{root}} = \overrightarrow{m_5} + \mathbf{M}_{root}\overrightarrow{b_{root}} - \mathbf{M}_{root}\overrightarrow{c_5} - \mathbf{M}_5\overrightarrow{b_5} \quad (10)$$

Figure 4 shows that our proposed algorithm can estimate the rotation angles of all the occluded markers correctly.

4. Motion data indexing and retrieval

The CMU motion database [6] was used in our study. We indexed each motion by determining all the joints that were affected visibly during the motion. For example, the hip and knee joints were affected visibly during kicking and walking motions. We used this information for indexing because users cannot manipulate all the puppet's segments during posing with continuous frame input. All the motions in the database that shared similar visibly affected joints with the input data were selected. Dynamic Time Warping [4] was performed using the input information and all the frames for each selected motion. We calculated the dissimilarity between the input data and each selected motion based on the Canberra distance [7]. The motion with the lowest dissimilarity score was retrieved and transferred to the 3D model, using a motion retargeting algorithm [3] to generate the animation.

5. Experimental results

We conducted an experiment to test the performance of our proposed puppet interface. Four participants created keyframe-based input data using the puppet for three motions: walking, kicking, and throwing. Each motion was produced five times with nonoccluded and occluded marker examples. Two markers were occluded in the occluded marker examples. In the experiment, the correct motion was retrieved with the lowest dissimilarity score for all the test input data when there were no occluded markers. If occluded markers were present, 95% of the input kicking data could be used to retrieve the correct motion with the lowest dissimilarity score. This was because the leg movement during a kicking motion was very similar to the leg movement during a walking motion. However, the correct motion could be found at the second position in the list, which was ranked in ascending order of the dissimilarity score. The accuracy was 100% for walking and throwing examples.

We also conducted an experiment with continuous frame-based input. We created each input motion, and 80% and 70% of the input data could be used to retrieve the correct motion with the lowest dissimilarity score with nonoccluded and occluded markers, respectively. The accuracy was slightly lower than the keyframe-based input because of the ARToolKit tracking process. In some frames, the ARToolKit tracking algorithm produced a result with a high level of error when we captured motions continuously. However, the correct motion was always found among the first three motions in the dissimilarity score listed.

Based on the experimental results, the retrieved motion could be transferred directly to the 3D model without any confirmation by users when a keyframe-based input is used without any occluded markers. However, a set of possible motions should be retrieved initially when occluded markers are present, or if continuous frame-based input is used, and the final output can be produced after user confirmation.

We only used pose information in this study and users needed to specify the style of motion manually. However, the proposed puppet interface can also be used to extract style information. For example, users could modify leg movements at different speeds with walking motion inputs. This speed information can then be used to specify the style of the walking motion that is retrieved. Therefore, users can input pose and style information into the system simultaneously, using the proposed puppet interface.

6. Conclusion

In this paper, we introduced a puppet interface for use in motion retrieval and with a computer animation system. User inputs into the system were generated by posing the puppet. The puppet's joint rotations were acquired using the ARToolKit. The marker occlusion problem was also addressed. The experimental results confirmed the performance of the proposed puppet interface. In future studies, we aim to improve the marker tracking algorithm to improve the reliability of the system with continuous frame-based inputs.

Acknowlegement

The authors thank Dr. Yasuyuki Matsushita (Microsoft Research Asia) for his constructive discussions and valuable suggestions. The research is sponsored by Microsoft Research Collaborative Research Projects (MSR CORE7).

References

[1] I. Barakonyi and D. Schmalstieg. Augmented reality agents in the development pipeline of computer entertainment. In *Proc. of ICEC'05*, pages 345–356, 2005.

[2] T.-C. Feng et al. Motion capture data retrieval using an artistś doll. In *Proc. of ICPR'08*, pages 1 –4, dec. 2008.

[3] E. S. L. Ho et al. Spatial relationship preserving character motion adaptation. *ACM Trans. Graph.*, 29(4):33:1–33:8, July 2010.

[4] E. Hsu et al. Guided time warping for motion editing. In *Proc. of SCA '07*, pages 45–52, 2007.

[5] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proc. of IWAR'99*, Oct. 1999.

[6] C. G. Lab. Cmu graphics lab motion capture database. `http://mocap.cs.cmu.edu/`.

[7] G. N. Lance and W. T. Williams. Computer programs for hierarchical polythetic classification (similarity analyses). *The Computer Journal*, 9(1):60, 1966.

[8] J. Lee et al. Interactive control of avatars animated with human motion data. *ACM Trans. Graph.*, 21(3):491–500, July 2002.

[9] Q. L. Li et al. Motionmaster: authoring and choreographing kung-fu motions by sketch drawings. In *Proc. of SCA '06*, pages 233–241, 2006.

[10] N. Numaguchi et al. A puppet interface for retrieval of motion capture data. In *Proc. of SCA '11*, pages 157–166, 2011.

[11] N. Pantuwong and M. Sugimoto. A fully automatic rigging algorithm for 3d character animation. In *SIGGRAPH Asia 2011 Posters*, pages 30:1–30:1, 2011.

[12] K. Yin and D. K. Pai. Footsee: an interactive animation system. In *Proc. of SCA '03*, pages 329–338, 2003.