Motion Frame Omission for Cartoon-like Effects

Maki Kitamura*, Yoshihiro Kanamori†, Jun Mitani†, Yukio Fukui†, Reiji Tsuruno*
*Kyushu University †University of Tsukuba

Keywords—Limited Animation, Motion Capture.

I. Introduction

Limited animation is a hand-drawn animation style that holds each drawing for two or three successive frames to make up 24 frames per second. This style was originally developed for 2D hand-drawn animation to reduce costs, but now it has matured and even been used in 3DCG keyframe animation to express unique aesthetic effects.

Traditional hand-drawn animation techniques such as shape deformation for exaggeration and holding frames are adequate for expressing motions of cartoon characters. Conversely, applying motion capture data to cartoon characters is inadequate as John Lasseter, the film director at Pixar and Walt Disney Animation Studios, stated as follows [1].

> *"Motion capture from human actors will always look realistic... for a human. But apply that motion to a chicken and it will look like a human in a chicken suit."*

However, in recent games featuring cartoon characters, motion capture data are often used to reduce costs.

In this paper, we propose a simple method for automatically converting motion capture data to imitate the unique expressions of limited animation. Especially, we focus on the following two features related to motion timing; one is that limited animation does not show subtle motion because it omits almost-static inbetween frames. The other is that it exaggerates motion speeds by omitting inbetweens, which is called "*nakanashi*" in Japanese [2]. Given motion capture data, our method holds each pose up to three frames to reduce subtle motion while it also omits poses in fast motion to exaggerate the speed. The accompanying movie with animating cartoon characters demonstrates that our method can imitate the expressions of limited animation.

II. Related Work

Most of the previous techniques for simulating cartoon animation styles are related to exaggeration by shape deformation [3]–[5]. These exaggeration techniques are orthogonal to our method; they do not control motion timing but can be combined with our method for better cartoon-like expressions.

In the field of 3D character animation, continuous control of motion timing has been actively studied [6]–[8]. These methods exaggerate motion by changing the duration time for each keyframe without deleting frames. On the other hand, our method employs discontinuous control of motion timing; our method modifies an input motion by holding or omitting frames to imitate the limited animation style as done in traditional hand-drawn animation.

The method proposed by Morishima et al. [9] is considered to be the most related to ours. Their method tackled to emulate the limited animation style by distilling keyframes as vertices in a motion curve, where their method tries to maximize the total length of the polyline that approximates the motion curve by connecting vertices. However, their method is inadequate to control local frame rates, e.g., holding an inbetween for two or three frames. Our method can not only control such local frame rates but also imitate inbetween omission (i.e., nakanashi) by deleting frames with sufficiently fast motions.

III. Algorithm

Our method reduces subtle motion by imitating frame holding and exaggerates motion speed by imitating inbetween omission. We assume that an input motion consists of a single action such as pitching and swinging. In case of a complex motion with compound actions, we segment the motion into individual actions manually. More sophisticated handling is left as future work.

A. Definition of Motion Speed

Suppose an input motion consists of N frames and a pose in each frame has a skeleton with n joints. The input motion is assumed to be subsampled to 24 frames per second, and the duration time for each frame is the same, i.e., 1/24 seconds. Let $\mathbf{p}_i^k \in \mathbb{R}^3$ be the position of joint i ($i = 1, 2, \ldots, n$) at k-th frame ($k = 1, 2, \ldots, N$). We calculate the sum of squared distances (SSD) of joint positions between k-th and $(k-1)$-th frames, and use it as the motion speed at k-th frame.

$$SSD(k) = \sum_{i=1}^{n} \|\mathbf{p}_i^k - \mathbf{p}_i^{k-1}\|^2 \tag{1}$$

B. Inbetween Omission for Speed Exaggeration

To imitate inbetween omission, we exaggerate motion speed by deleting frames with the maximum speed (Fig. 1). Specifically, we first find k_{max}-th frame with the maximum SSD, and then delete the frame. Because this deletion changes the total number of frames, we try to keep the total number by duplicating a frame whose SSD is the minimum so that the overall motion changes as little as possible. This is done by finding k_{min}-th frame whose SSD is the minimum, and then duplicating $(k_{min}-1)$-th frame. However, naive repetition of this duplication makes the motion completely stop. We therefore constrains the number of successive duplicated frames up to three. If the frame is already duplicated more than three times, we find a frame whose SSD is the next smallest. We iterate the above procedure until a user-specified number of frames are deleted. Note that the duplication for keeping the total number of frames is just an option; it can be omitted if the remarkable change of the total number of frames is

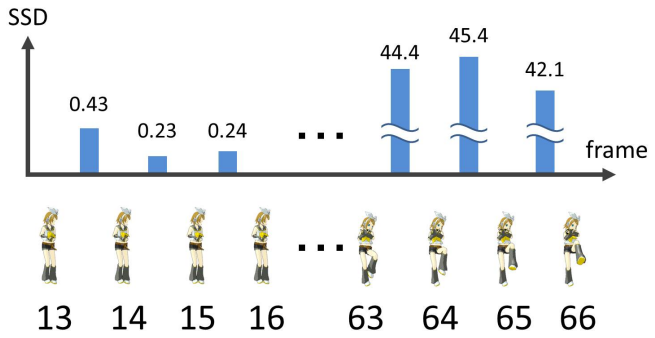

Fig. 1. A graph of SSD between each pair of two successive frames. The vertical axis shows SSD while the horizontal axis shows frames. In this example, 65-th frame will be omitted because $SSD(65)$ is the largest, and then 14-th frame will be duplicated because $SSD(15)$ is the smallest.

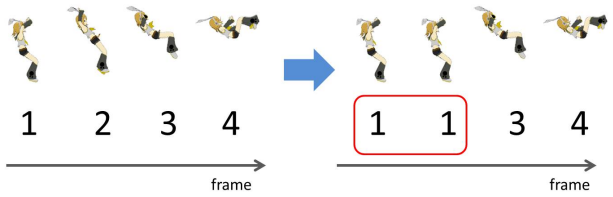

Fig. 2. Holding frames. Because $SSD(1)$ is the smallest, we copy the first frame to the second in order to reduce subtle motion.

permitted. We summarize the whole process in Algorithm 1 where #DUPLICATED(k) denotes the number of successive duplicated frames for k-th frame.

Algorithm 1 Inbetween omission.

$N_{io} \leftarrow$ user-specified number of inbetween omission
for $i \leftarrow 1$ to N_{io} **do**
  $k_{max} \leftarrow \mathrm{argmax}\, SSD(k)$
  delete k_{max}-th frame
  $k_{min} \leftarrow \mathrm{argmin}\, SSD(k)$ s.t. #DUPLICATED($k-1$) < 3
  duplicate $(k_{min}-1)$-th frame
end for

C. Holding Frames for Reducing Subtle Motions

We reduce subtle motion by imitating frame holding (Fig. 2). We find k_{min}-th frame with the smallest SSD, and we copy $(k_{min}-1)$-th frame to k_{min}-th frame. Similarly to inbetween omission, we constrain the number of successive copied frames up to three. We repeat this until a user-specified number of frames are copied. We summarize the whole process in Algorithm 2 where #COPIED(k) denotes the number of successive copied frames for k-th frame.

D. Discussion

In the process of inbetween omission, our method currently deletes only a single frame per iteration. An alternative way might delete a user-specified number of frames with the largest SSD simultaneously. However, such simultaneous deletion yields smaller visual difference from the input motion than ours because the selected frames by such deletion are not always successive but tend to be discrete. Conversely, our method tends to delete successive frames, which results in strong exaggeration of motion speed. Fig. 3 shows a comparison between the simultaneous approach and our method.

Algorithm 2 Holding frames.

$N_{hf} \leftarrow$ user-specified number of holding frames
for $i \leftarrow 1$ to N_{hf} **do**
  $k_{min} \leftarrow \mathrm{argmin}\, SSD(k)$ s.t. #COPIED($k-1$) < 3
  copy $(k_{min}-1)$-th frame to k_{min}-th frame
end for

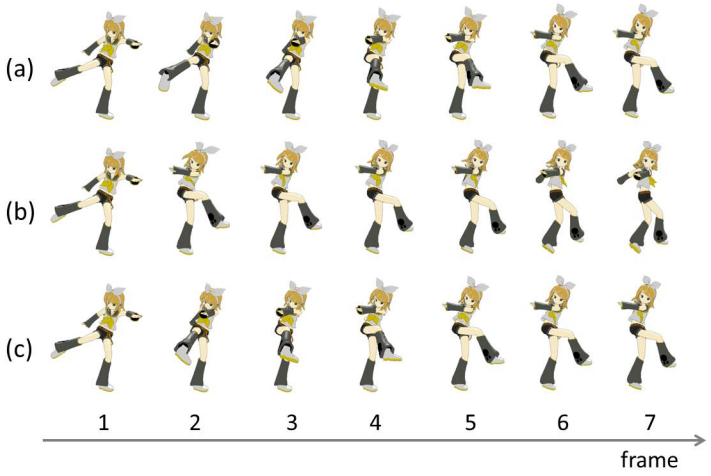

Fig. 3. Comparison of (a) original motion data, (b) our method and (c) simultaneous deletion. In our method, frames from second to fifth will be omitted, while only the second frame is omitted in simultaneous deletion. Consequently, our method exaggerates the motion speed more strongly than simultaneous deletion.

Regarding the order of which inbetween omission or frame holding should be performed first, we currently process inbetween omission first. We also experimented with the reversed order and confirmed that the order does not matter. This is because the frames omitted by frame holding do not affect those omitted by inbetween omission and vice versa.

IV. RESULTS

We implemented our prototype using C++ and Qt library, and ran on a PC with Intel Core i7-2760QM 2.40GHz CPU. We used publically-available motion capture data obtained from Mocapdata.com [10] and CMU Graphics Lab Motion Capture Database [11]. We used 3D models[1] included in MikuMikuDance [12]. All data are captured in 24 frames per second, and these figures show partial frames only. We strongly recommend watching the supplemental movie to confirm the resultant effects of our method.

[1]Kagamine Rin is copyrighted by Crypton Future Media, INC.

TABLE I. THE STATISTICS OF OUR EXPERIMENTS

Name	Total number of frames	N_{io}	N_{hf}	Processing time[ms]
Pitch	129	4	33	69
Kick	92	4	30	28
Tennis	88	4	29	27
Baseball swing	120	6	40	63

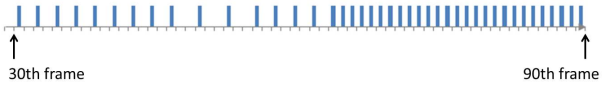

Fig. 8. The keyframes corresponding to the partial frames of Fig. 4.

Figs. 4, 5, 6 and 7 show our results. In these figures, upper rows show original motion capture data while lower rows show our results. In Fig. 4, the motion speed of the right arm is clearly exaggerated by omitting original frames from third to sixth to imitate inbetween omission. Fig. 5 shows the result of a kick motion, where the motion speed of the right leg is clearly exaggerated by omitting original frames from second to fifth. In the result of tennis swing motion of Fig. 6, the motion speed of the right arm is clearly exaggerated by omitting original frames from second to fifth. Fig. 7 shows the baseball swing motion. In this example, the original frames from second to seventh are omitted to exaggerate the motion speed of the both arms.

Fig. 8 shows the keyframes corresponding to the partial frames of Fig. 4. The blue bar at k-th frame means that k-th frame is an original frame (i.e., not duplicated or copied, what we call a keyframe here), whereas no bars mean that the corresponding frames are duplicated or copied from a keyframe. Fig. 8 demonstrates that frames with a small amount of motion are held while fast-moving frames are not. We can confirm that subtle motion is reduced successfully by watching the motion as an animation.

Table I shows the statistics including the total number of frames, the number of inbetween omission N_{io}, the number of frame holding N_{hf} and processing time of each motion. In our experiments, we found that around five is just right for N_{io}. If $N_{io} > 5$, the motion no longer looks continuous. We also found that one-third of the total number of frames is adequate for N_{hf}.

V. CONCLUSION

In this paper, we have proposed a simple method for automatically converting motion capture data into the limited animation style. Among the features of limited animation, we focused on inbetween omission and frame holding for discrete timing control. Inbetween omission was imitated by omitting frames with the largest SSD while frame holding was reproduced by copying frames with the smallest SSD to their corresponding successive frames. The accompanying video successfully demonstrated that our method can imitate the limited animation expressions.

Currently our method handles motions not in the screen space but in the object space. However, the relative speed of motion changes according to the distance between the character and the camera; even if a frame has a very large SSD in the object space, the SSD in the screen space might be very small when the camera is far from the character. In contrast, a small motion in the object space might be quite large in the screen space if the camera is closed to the character. Therefore we would like to consider the motion speed in the camera coordinate system in future work.

Another interesting directions of future work include combination of shape deformation for cartoon animation [4] and automatic determination of the optimal numbers of inbetween omission and frame holding.

ACKNOWLEDGEMENT

We would like to thank Shigeru Kuriyama and Tomohiko Mukai for their helpful feedback. This work was supported by the Japan Society for the Promotion of Science (JSPS).

REFERENCES

[1] J. Lasseter, "Tricks to animating characters with a computer," *SIGGRAPH Comput. Graph.*, vol. 35, no. 2, pp. 45–47, 2001.

[2] T. Ozawa, *Anime sakuga no shikumi (in Japanese)*. Works Corpolation, 2004.

[3] S. Chenney, M. Pingel, R. Iverson, and M. Szymanski, "Simulating cartoon style animation," in *Proceedings of the 2nd international symposium on Non-Photorealistic Animation and Rendering (NPAR)*, 2002, pp. 133–138.

[4] J. Wang, S. M. Drucker, M. Agrawala, and M. F. Cohen, "The cartoon animation filter," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 1169–1173, 2006.

[5] Y. Li, M. Gleicher, Y.-Q. Xu, and H.-Y. Shum, "Stylizing motion with drawings," in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA)*, 2003, pp. 309–319.

[6] E. Hsu, M. da Silva, and J. Popović, "Guided time warping for motion editing," in *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA)*, 2007, pp. 45–52.

[7] J. McCann, N. S. Pollard, and S. Srinivasa, "Physics-based motion retiming," in *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA)*, 2006, pp. 205–214.

[8] J.-y. Kwon and I.-K. Lee, "An animation bilateral filter for slow-in and slow-out effects," *Graph. Models*, vol. 73, no. 5, pp. 141–150, 2011.

[9] S. Morishima, S. Kuriyama, S. Kawamoto, T. Suzuki, M. Taira, T. Yotsukura, and S. Nakamura, "Data-driven efficient production of cartoon character animation," in *SIGGRAPH 2007 sketches*, 2007.

[10] *Mocapdata.com*, http://www.mocapdata.com/.

[11] *CMU Graphics Lab Motion Capture Database*, http://mocap.cs.cmu.edu./.

[12] *MikuMikuDance*, http://www.geocities.jp/higuchuu4/.

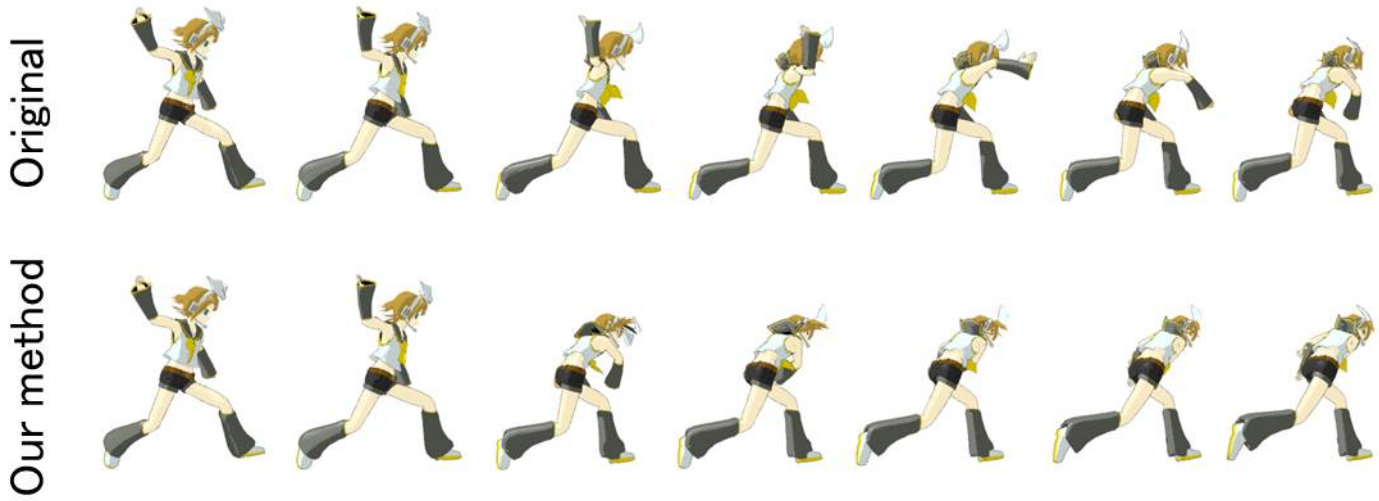

Fig. 4. Partial frames of the pitch motion. In our result (below), the motion speed of the right arm is exaggerated by deleting frames from the third to the sixth from the original motion (above). The total number of frames is 129, $N_{io} = 4$ and $N_{hf} = 69$.

Fig. 5. Partial frames of the kick motion. The motion speed of the right leg is exaggerated. The total number of frames is 92, $N_{io} = 4$ and $N_{hf} = 30$.

Fig. 6. Partial frames of the tennis motion. The motion speed of the right arm is exaggerated. The total number of frames is 88, $N_{io} = 4$ and $N_{hf} = 29$.

Fig. 7. Partial frames of the baseball swing motion. The motion speed of the both arms is exaggerated. The total number of frames is 120, $N_{io} = 6$ and $N_{hf} = 40$.