

MOCKEY : MOTION CAPTURE AS A TOOL FOR KEYFRAMING ANIMATION

Joëlle Tilmanne¹, Sullivan Hidot², Thierry Raver¹

¹ Laboratoire de Théorie des Circuits et Traitement du Signal (TCTS), University of Mons, Belgium

² Service Informatique (INFO), University of Mons, Belgium

ABSTRACT

This project, initiated in collaboration with the animation studio DreamWall, studies the use of motion capture as a tool for cartoon animators. We implemented an algorithm for the automatic extraction of "keyframes" from the captured motion, and tackled several technical issues concerning the use of motion capture for the project goal.

KEYWORDS

Motion Capture, Keyframe, Motion Analysis

1. INTRODUCTION

Animation consists in the rapid display of a sequence of images, which creates an optical illusion of movement due to the human phenomenon of retinal persistence. The animation process is applied to a lot of different domains, ranging from the coarse motion seen in video games to the precise human like motion of 3D films, and including fields like virtual reality or character animations for human-computer interactions. In this project, we focused on animation for cartoons.

The manual keyframing technique is the oldest and still the main technique used for cartoon animation. An animator manually defines positions and orientations of the character's joints for a set of specific animation times (called key-times), and the final animation is obtained after interpolation of the motion between those key-frames. In traditional animation the assistants of the senior animator perform this interpolation, and in computer animation software calculates the interpolation. In both cases the whole process can take a lot of time: more than one day for a few seconds of animation, as around 25 postures have to be created for each second of animation. It also requires an experienced and skilled animator to create an animation that looks realistic to the human eye, which is much more complicated than one may think. Indeed, animation is not only about creating motion; it is about adding emotions to the motion, in order to make the character look alive. The time and the experience needed grow as the skeletal complexity of the character to be animated increases, which means that this technique is better suited for animating cartoon characters (which generally have a simplified skeleton) than human characters. Despite these drawbacks, keyframe techniques are still the most common ones for 3D movies and games.

Unfortunately, as competition in this field becomes more and more fierce, and with the difference in cost of animators in Europe and in Asia, local studios need to find ways to reduce the time-consuming manual process. This is why in this project we investigated the use of motion capture techniques for keyframe animation. This is not a new idea, as the origins of motion analysis for cartoon animation go back to 1914 with the "rotoscoping" technique, which consists in transforming a filmed scene into a cartoon by using a translucent projection table on which each image of the

film is placed and can then be reproduced as a drawing by the animator using tracing paper. This technique was used for example in Disney's "Snow White".

The "motion capture" approach is similar, as the animation is obtained by modifying, adapting and/or concatenating sequences of 3D positions obtained thanks to motion capture on an actor who performs for real the desired motion for the animation. The principle of motion capture in animation is thus to record the performance of a human in 3D and reproduce it with high fidelity onto a virtual character. Each animation produced is thus truly unique. Motion capture must thus either be performed "on demand" for any specific animation, or a huge motion database has to be available. This lack of flexibility, and the significant costs related to the motion capture process are the main drawbacks of this approach.

In this project, we investigated the use of motion capture not for replacing the animator but as a tool that he can use, for example, for a faster design of the main lines of the motion. In this perspective, we developed methods to extract keyframes from the motion capture data, and to modify the motion captured from a human skeleton to apply it on another skeletal size, even if it is not human-like.

Several attempts have already been made in this direction, with the extraction of keyframes for animation like in the MoCaToon project [11] but also for editing, classifying/indexing [12, 3] and compressing [5] the motion capture data.

The main steps that had to be taken into account in this project are presented here. Section 2 introduces the framework and the technical interface. Section 3 describes the format of the animation data used to be able to share data with the DreamWall animators. Section 4 explains the problems inherent to the motion capture system that we use and how we fixed some of the issues it brings. In Section 5, the algorithm chosen for the keyframe extraction is presented. Section 6 briefly presents another motion data representation closer to the animator's idea of a keyframe that could be further investigated. Finally, Section 7 draws some perspectives and conclusions of this work.

2. MOCAP SYSTEM PROGRAMMING ARCHITECTURE

In previous Numediart projects, a solution for controlling the acquisition and displaying movements on a 3D virtual character [7] had already been developed with the Animazoo IGS-190 Motion Capture System and its SDK [1]. The chosen software architecture divided the tasks onto two hosts, the acquisition tasks on one and the data-processing tasks on the other. The two host communicate by network. We started from this basis and studied how we could interface motion capture tools with the NeuroToon 3D engine [8]. The first step of a framework for processing and transmitting the data frames acquired by motion capture tools was then implemented in C++.

To improve reusability and adaptability of the components, we



Figure 1: Eric de Staercke and Emmanuel Guillaume in “Bla-Bla”.

proposed a distributed architecture based on modules. Each one possesses the following functionalities:

- An input (instrumentation, file, network receiver, ...)
- One or more data processing functions.
- One or more outputs (display, network sender, ...)

We created thus block-components reproducing the functionalities of the previous works. This architecture enables us to customize tools very fast and to make easy the inclusion of these components into more complex softwares like multi-thread systems. This architecture was already used for an industrial application during the project, as it permitted to adapt a software solution used in the musical comedy “Bla-Bla”. This show (December 2009), directed and written by Bernard Halut, staged 3D virtual animations in real time interaction with comedians and puppets and is shown in Figure 1.

3. ANIMATION MOTION DATA FORMAT

In order to permit the animators to exploit the signals extracted from an acquisition tool, we implemented an output function with the Autodesk FBX technology [2]. This standard data format is compatible with different software packages used by animators. We applied the SDK proposed by Autodesk to import and to manipulate the data and the FBX files. The component loads a 3D skeleton/character model. Next, it analyzes the geometric and structural properties of the model, adapts the data frames of the motion capture recording, encodes and writes the result in a FBX file. The system can select the frames that we want to store in the final FBX file. This is destined to keep only the keyframes. Visualization of the results is possible thanks to a QuickTime plug-in as shown in Figure 2. If the time intervals are not constant between two samples, this technology provides position interpolation during the visualization.

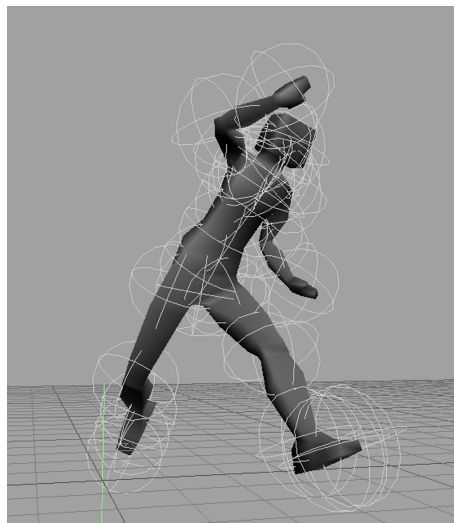


Figure 2: FBX Character Visualization.

4. DATA CORRECTION

The quality of the keyframes is important, as it will determine the quality of the animation. As the keyframes will be detected from motion capture data, the first step is to insure that the mocap data does not present abnormalities.

The IGS-190 inertial motion capture suit used in this project is based on sensors including accelerometers, gyroscopes and magnetometers, and on a virtual skeleton of the actor whose size is fitted in a calibration step. The pose of the skeleton is thus accurately monitored, but the global position of the actor can only be estimated as there is no global positioning system. The size of each segment of the skeleton is known. If we consider that the lowest part of the body is in contact with the ground and can thus be taken as a known reference coordinate, we can calculate the position of the root (hips) from frame to frame.

That estimation is done by knowing the size of the skeleton and especially the legs, and by considering that the lower part of the body is in contact with the ground and is thus static and considered as the known reference coordinate used for the calculation of the root (hips) position.

That estimation is a good approximation of reality but the system has its limits and sometimes the lowest point of the body according to the motion capture is not truly the lowest point or that lowest point is not in contact with the ground (running, jumping, stairs climbing, etc). Some of those situations can be avoided and for others like jumping, an approximation is evaluated by the system. The most obvious error is the one that occurs when the actor is walking and that the wrong foot is considered in contact with the ground. This situation leads to a sliding motion and is obviously inaccurate when looking at the recorded motion.

In addition to that, a second source of problems is the adaptation of motion captured data to a skeleton that is different from the actor’s skeleton. This is an essential problem for our study, as motion capture is useful only if it can be used for the animation of characters that do not have exactly the same morphology as the human actor.

The first problem that we have considered is the ground con-

tact. In motion capture, no physical constraints force the foot to lie on the ground. If the system is properly calibrated, the original data recorded will show a skeleton whose feet touch the ground at appropriate times while walking. But nothing ensures that when the motions are applied to other skeletons there will be ground contact. Furthermore, a different skeleton does not only affect the feet contact with the ground. The global displacement of the skeleton is stored in the motion capture data file as the positions of the hips in the 3D space, but if the skeleton has a different size, the global displacement is no longer the same. It must be recalculated given the angles between the legs segments, the lengths of those segments and the instants when the feet touch the ground. Post processing of the data is thus needed to make it usable for animation applications.

A first algorithm has been developed for detecting which part of the body is in contact with the ground for each frame with respect to its height and speed, and recalculating from that contact point the position of the root of the skeleton (the hips). But it has to be further studied and improved as some “glitches” are still present in the corrected motion. Some adaptations of the angles have also to be implemented for skeletons which have a very different length ratio than human skeletons (like the very long feet of a cartoon character for instance, which should bend less than a human feet would do).

A solution to that would be for a user to check the sequence and decide which part of the body is in contact with the ground. From that information, the algorithm can recalculate the position of the hips and thus of the whole skeleton.

Trying to make the algorithm more autonomous, we considered a case for which we have a special interest: walk sequences. In that case, we know that while the left foot is swinging, the right one is on the ground and vice versa. We developed a new algorithm that detects when a leg stops bending forwards and starts bending backwards by analyzing the rotation of each upper leg around the hips axis. By detecting the peaks of these signals, we were able to detect which foot is on the ground and then to apply our correction to the whole walk sequence. The first part of Figure 3 shows the segmentation of a whole walk sequence into “left leg step” (blue) and “right leg step” (red) based on the rotation of each upper leg. In the second part of the figure, the height of the right and left foot during the same sequence are shown.

Those corrections are the first steps towards clean and adaptable data, and significantly improve the quality of the data. Even if the most important part of the process is to pay as much attention as possible during the recording process, the algorithms we have developed here enabled us to correct errors and approximations of the system, and should be developed further.

5. KEYFRAMES EXTRACTION

In this section we focus on keyframe extraction from motion data recorded by a system which gives the spatial coordinates during movement. In comparison to video keyframing, there are few studies about this in the scientific literature (see [4, 6, 12] for the most recent). We chose an approach of keyframing based on a mutual information measure [12] for three reasons: ease of implementation, speed of implementation, and, of course, good results.

Generally speaking, the method is based on displacement histograms between frames. From these histograms we deduced the probabilities of each histogram outcome for the frame f (denoted by $B_f(r)$ with the same notations of the article), the joint probability of a marker having level r in frame f , and also discretization

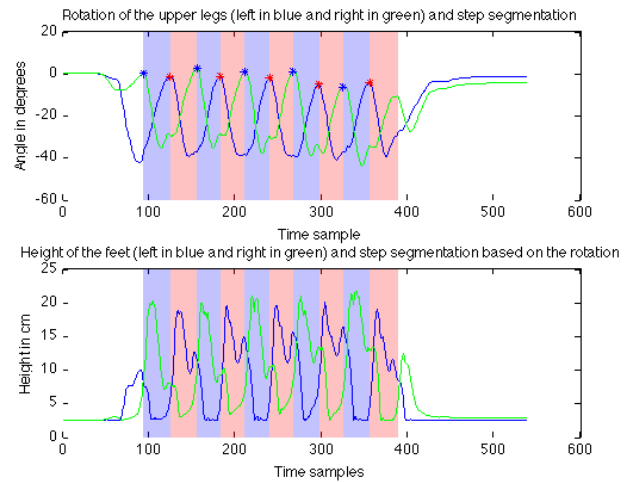


Figure 3: Segmentation of a walk sequence.

s in frame $f + 1$ (denoted by $C_f(r, s)$). Mathematically, we obtain the mutual information I_f^X for the X coordinate:

$$I_f^X = - \sum_{r=1}^n \sum_{s=1}^n C_f^X(r, s) \log \left(\frac{C_f^X(r, s)}{B_f^X(r)B_f^X(s)} \right)$$

where n is the number of levels of the displacement histograms. Finally, we compute the total information I_f of the displacement from frame f to $f + 1$:

$$I_f = I_f^X + I_f^Y + I_f^Z$$

In [12], the authors also propose a normalization computed with a mean average, in order to accent local changes. Keyframes are obtained by considering the significant maxima of this normalization (called “localized mutual information”). This approach highlights the directional changes compared to the methods based on the magnitude of the velocity [4]. In our point of view, it appears to be an interesting technique as the results are very close to what we could find with an intuitive way.

We will present here two examples of keyframe extraction on two different type of data. In those examples, the localized mutual information is computed with one second window size and we set $n = 256$ discretization levels. In addition to the procedure, we impose that the expected keyframes are sufficiently distributed in time. This kind of trick allows to avoid sets of very close keyframes which concerns the same keypose.

The first type of example data presented here was captured with the inertial IGS-190 system. We study a sequence of capoeira movement consisting of 1062 frames with a sample frequency of 120 Hz. The second example data is a movement with a lower complexity. A dancer has been equipped with only 15 sensors fastened on head, torso, arms, wrists, hands, pelvis, hips, legs and feet. For this split-jump dance figure executed in the side way, 92 data frames were recorded with a sampling frequency of 25 Hz.

Results for the capoeira sequence are shown at Figures 4 for the localized mutual information (red circles correspond to the local minimas) and 5 for the corresponding keyframes, as we chose to detect 8 keyframes out of the 1062 data frames. This example uses Matlab Motion Capture toolbox for plotting keyframes [9].

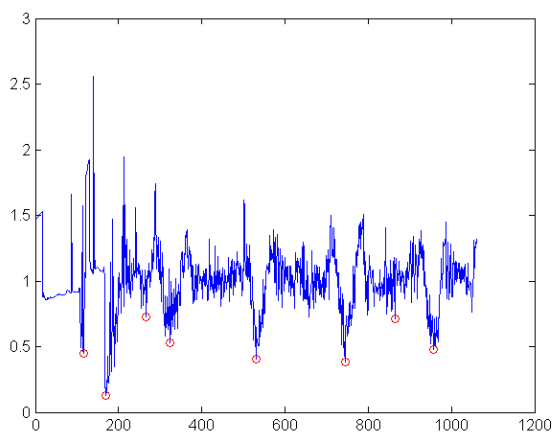


Figure 4: Localized mutual information of a sequence of capoeira.

Figure 6 shows the localized mutual information for the split-jump movement and Figure 7 gives the obtained keyframes. In this case, 4 keyframes were detected from the 92 data frame.

In both cases the keyframes detected provide a good summary of the motion.

6. CHARACTER SKETCHING WITH CURVES

Keyframe is not only a dynamic notion. If we can detect some important variations in the animation curves from a motion capture recording session, an animator would not necessarily choose these extreme positions as key poses.

From our discussions with animators, it appeared that a “keyframe” could not only be defined as a dynamic motion. The animators also take into account some kind of esthetic criterion to choose these poses. In order to consider this aspect, we tested a method to summarize the character with curves. We used Bezier curves, two for the bodies and one for the arms as shown in Figure 8. We obtained a first positive assessment from the animators for this test and its potentiality to help detecting some interesting positions.

The curves may better represent the idea of the motion that animators see than the complete set of motion capture data. This motion representation should thus further investigated, as it may be an interesting way of taking into account some of the esthetic criterion on which animators base their keyframe choice.

7. CONCLUSIONS AND PERSPECTIVES

The use of motion capture for cartoon animation is a hot topic and much criticism is centered around it: the loss of artistic touch when motion capture is used instead of pure manual keyframing, the many drawbacks of motion capture that limit this use, the fact that it is intended to replace artists animators by computers, etc. As is often the case, the controversial method is not all good nor all bad. Even if a few decades ago some people believed motion capture was the “miracle tool” which would allow making realistic animations easily and for no cost; this opinion has now changed [10]. For some well defined cases and when it is used well, motion capture does have its advantages over classical animation.

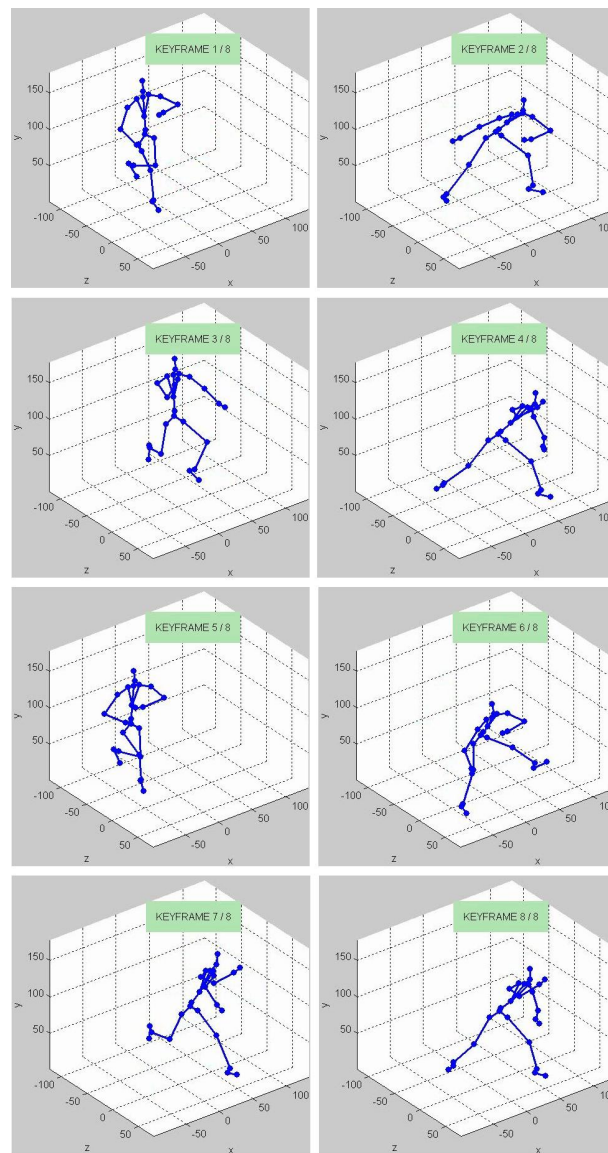


Figure 5: Keyframes detection for the sequence of capoeira.

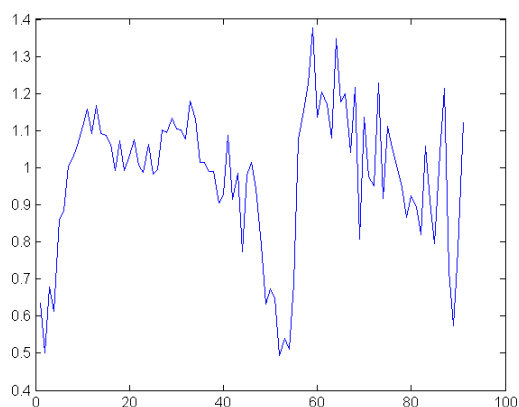


Figure 6: Localized mutual information of a split-jump movement.

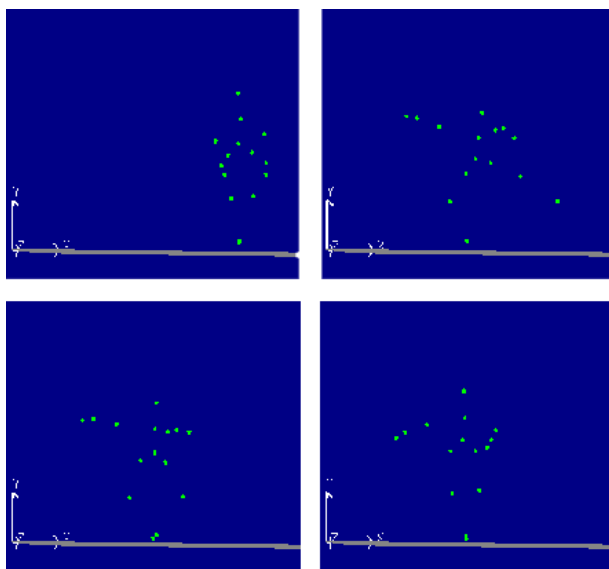


Figure 7: Keyframes detection for the split-jump sequence.

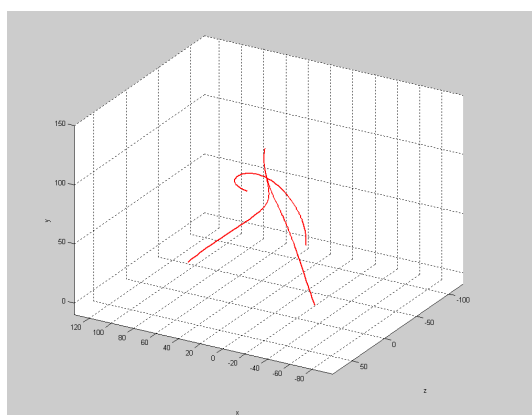


Figure 8: Skeleton sketching with Bezier curves.

There is still no evidence that motion capture could be an efficient solution to speed up the production of cartoon animation but, in this project, we highlighted two important dimensions that define the keyframes. The first dimension is the one represented in the algorithm we implemented: time and the evolution of the motion over time. We showed that using this dimension enabled a first “keyframe” extraction that was useful even if the keyframes are not exactly the ones that an animator would pick up. The second dimension is the esthetic criterion mentioned by the animators. A more complex algorithm should thus be implemented in the future to refine keyframe detection, and to include some kind of esthetic criterion to better represent the an animator’s definition of a keyframe.

One way of better taking into account the “keyframe” as they are seen by animators would be to collect a database of keyframes chosen by animators, and use that database to train a detection algorithm. Such a solution would enable us to model the animators’ esthetic criterion without expecting them to write a formal definition of what they see as a keyframe. The curves-based representation of the skeleton should also be further investigated as a way of summarizing the motion and highlight its esthetic representation. A keyframe detection based on this data representation should thus be implemented.

In this project, we also suggested a partial solution to the problem of correcting and adapting the motion capture data, but this is a large topic that has to be further investigated, as physiological limits and ground contact constraints should be respected for any animation.

8. ACKNOWLEDGMENTS

numediart is a long-term research program centered on Digital Media Arts, funded by Région Wallonne, Belgium (grant N°716631). The authors would like to thank the comedian Sebastien Marchetti for his participation in a motion capture session.

9. REFERENCES

9.1. Scientific references

- [3] A. Baak, M. Müeller, and H. Seidel. “An efficient algorithm for keyframe-based motion retrieval in the presence of temporal deformations”. In: *1st ACM international Conference on Multimedia information Retrieval* (2008). Pp. 451–458. P.: 119.
- [4] F. Bevilacqua, J. Ridenour, and D.J. Cuccia. “3D motion capture data: motion analysis and mapping to music”. In: *Workshop/Symposium on Sensing and Input for Media-Centric Systems* (2002). P.: 121.
- [5] E. Bulut and T. Capin. “Key Frame Extraction from Motion Capture Data by Curve Saliency”. In: *Computer Animation and Social Agents* (2007). P.: 119.
- [6] T. Kim, S.I. Park, and S.Y. Shin. “Rhythmic-motion synthesis based on motion-beat analysis”. In: *ACM Transaction on Graphics (TOG)* 22.3 (2003). Pp. 392–401. P.: 121.
- [7] C. Mancas-Thillou et al. “Sensor-Based Mini-Opera”. In: *QPSR of the numediart research program* 1.1 (2008). P.: 119.
- [8] M. Mancas et al. “Matrix: Natural Interaction Between Real and Virtual Worlds”. In: *QPSR of the numediart research program* 2.1 (2009). P.: 119.

- [10] A. Menache. *Understanding motion Capture for Computer Animation and Video Games*. Morgan Kauffman Publishers Inc., San Francisco, CA, USA, 1999. P.: 122.
- [11] S. Morishima et al. “Data-driven efficient production of cartoon character animation”. In: *ACM SIGGRAPH 2007 Sketches* 76 (). P.: 119.
- [12] C.K.F. So and G. Baciú. “Entropy-based motion extraction for motion capture animation: Motion Capture and Retrieval”. In: *Computer Animation and Virtual Worlds* 16.3-4 (2005). Pp. 225–235. Pp.: 119, 121.

9.2. Software and technologies

- [1] “Animazoo company”. URL: www.animazoo.com/. P.: 119.
- [2] “Autodesk Company”. URL: www.autodesk.com/. P.: 120.
- [9] “MATLAB Motion Capture Toolbox”. URL: www.cs.manchester.ac.uk/~neill/mocap/. P.: 121.