

Motion Tracking for Volumetric Motion Capture Data

Derek Roberts*, Ying Zhu†

*Kentucky State University derekmroberts@gmail.com

†Georgia State University yzhu@gsu.edu

Abstract—Motion capture is often used to study the biomechanics of human motion. The first step in this type of research is to process and retrieve biomechanical data from the motion capture data. In this paper, we introduce a new method to dynamically track joint locations in volumetrically captured human motion sequences. The challenge is to identify corresponding components in a sequence of 3D meshes that not only transform and deform but may also have different vertex structures. To solve this problem, we have developed a method that uses user-selected vertices from the surface data, derives joint locations, and attempts to track their positions frame by frame.

I. INTRODUCTION

Motion capture is often used to study human motion, with applications in areas such as sports medicine. We are interested in studying the motion of athletes (such as baseball pitchers) for performance analysis, injury prevention, and rehabilitation evaluation. However, the first step is to retrieve biomechanical data from the motion capture data. There are four types of motion capture systems: electromagnetic, optoelectronic, inertial based, and image-processing based systems. In our project, we use a new volumetric motion capture system 4DViews [1], which captures the motions of characters and produces a sequence of high-resolution, time-varying polygon meshes. In this project, the volumetrically captured single-character polygon mesh sequence contains meshes that range from 45,306 polygons to 75,378 polygons, averaging at 65,667 polygons per frame. This volumetric motion capture system, an advanced image-processing based system, offer a less physically invasive way to capture motion. A suit or apparatus (e.g., mark) is not required, and thus the performance is more genuine and natural, a particularly useful feature for capturing motions in sports.

However, volumetric motion capture data present a new challenge. Marker-based motion capture systems create animated skeleton models, with the anatomical joints marked. A volumetric motion capture system creates animated 3D polygon meshes, which present the 3D motions much more accurately but without the clear marking of the joints. In addition, the polygon meshes move and deform from frame to frame. Therefore, we need to automatically identify joints (such as shoulder, elbow, and wrist) frame by frame. This is a problem of time-varying shape registration [2].

We have developed a method to use the surface data in a volumetrically captured animation sequence to accurately and dynamically derive sub-surface joint motion. The proposed system has the user calibrate joint locations on the initial mesh by selecting vertices. These vertices are then updated frame by frame by a kinematics-based point projection estimation system. Each frame is keyframed in order to animate these updated joint locations. This project is still a work-in-progress. The main issue with the current approach is that the point projection animation system drives the vertex selection slowly away from the local user calibrated joint location on the mesh.

II. RELATED WORKS

In numerous studies of motion analysis, the methods of motion capture have involved traditional marker-based motion capture, which typically involves a tight and restrictive suit with reflective markers placed on the outside of the suit to identify joints. The tight suits and apparatus used in these methods can restrict the range of motion [3] [4] and capture motion that is not representative of the performer's intended performance.

Computer vision-based motion tracking has made significant advancements in recent years [5] [6]. In computer-vision based systems, people can wear regular clothes and move naturally. Computer-vision based systems can be divided into two categories: model-based tracking and feature-based tracking [4]. In model-based tracking, entire 3D models are reconstructed from single or multi-view 2D image sequences. In feature-based tracking, key points are tracked for each frame of the image sequence, and skeleton models are often created. Although feature-based tracking seems well suited for biomechanical analysis because joints are already identified, there are some issues with accuracy. For example, feature-based tracking cannot accurately track pronation, which is often used in pitching baseballs or serving tennis balls. Many feature-based tracking systems are designed to detect full-body poses (e.g., walking or dancing), which can tolerate a large margin of error. However, for more precise biomechanical analysis, such inaccuracies pose problems. Therefore, we are interested in developing a method to identify and track human joints from the volumetric capture data. We do not start from 2D images, but the animated 3D meshes already reconstructed by a markerless motion capture system, such as 4DViews. Therefore, we are not dealing with a computer vision problem, but a time-varying shape registration problem [2]. The animated 3D mesh can help identify small local rotations such as pronations. The added temporal constraint may help speed up the shape registration process. The proposed system would work with processed volumetric animation data stored in an Alembic [7] open-source file.

III. PROPOSED METHOD

Georgia State University has installed a 4DViews volumetric motion capture system. The outputs of the 4DViews system are animated 3D meshes in the Alembic file format (Fig. 1). The algorithms for constructing the 3D meshes are proprietary. Some consecutive frames may share the same 3D mesh, and then a new 3D mesh is generated for the subsequent frames, and so on. The 3D mesh may be deformed as the animation progresses. Because the motion capture is markerless, there is no marking of the human body components in the 3D meshes. In computer-vision based methods, 2D images are

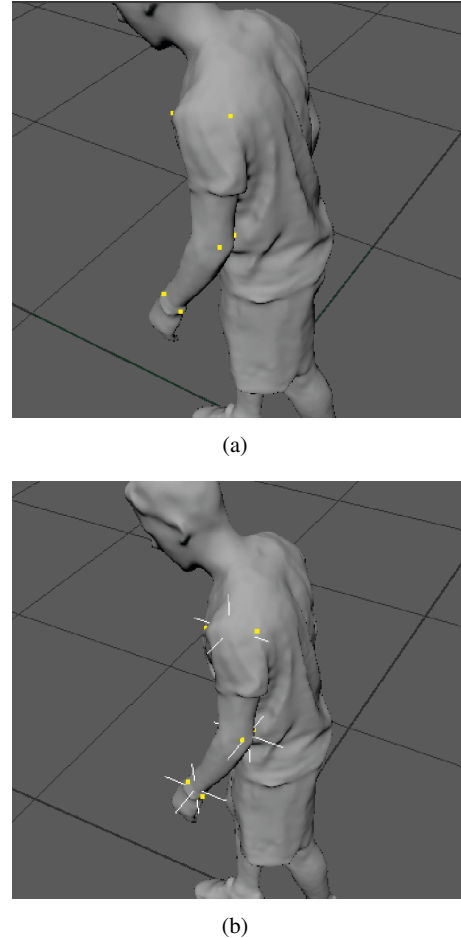


Fig. 1. Vertex pairs (a) are selected and then the joint locators (b) are moved to the three-dimensional midpoint between the corresponding vertex pairs.

processed to identify the locations of the major joints. However, in our case, 3D geometry objects are processed to identify the locations of the major upper body joints (shoulder, elbow, wrist). This is a time-varying shape registration problem [2].

Our idea is to manually mark the locations of the joints on the 3D meshes for the first few frames (Fig. 1) and then estimate the locations of the joints in the subsequent 3D meshes by their motion trajectory. Therefore, the temporal constraints are used to help solve the shape registration problem. The specific steps are described below.

First, vertex pairs are selected on either side of a joint such that the midpoint between the vertices would accurately represent the joint location. After each pair is selected, the pair is put into a single

set so that the number of vertex pair sets matches the number of joints. For the first two frames, these vertex pairs are either selected by name, if the mesh is still the same in the second frame, or manually calibrated if it is not. This way there is enough velocity data to calculate acceleration for the displacement formula used for point projection estimation later.

After the third frame, the vertex pair locations are updated by point projection estimation (Fig. 2).

The projection estimation itself is based on a slight adjustment to basic kinematics displacement. The position of the projected point $\vec{p}_i$ is equal to the sum of the vector of the current point $\vec{c}_i$, the instantaneous velocity $\vec{v}_i$, and the acceleration $\vec{a}_i$ divided by 2.

$$\vec{p}_i = \vec{c}_i + \vec{v}_i + \frac{\vec{a}_i}{2} \quad (1)$$

Instantaneous velocity $\vec{v}_i$ is calculated as the difference of the current point $\vec{c}_i$ and the previous point $\vec{c}_{i-1}$.

$$\vec{v}_i = \vec{c}_i - \vec{c}_{i-1} \quad (2)$$

Acceleration $\vec{a}_i$ is calculated as the difference between the current instantaneous velocity $\vec{v}_i$ and the previous $\vec{v}_{i-1}$.

$$\vec{a} = \vec{v}_i - \vec{v}_{i-1} \quad (3)$$

For each vertex, the velocity from the previous frame, the velocity from the current frame, the location from the previous frame, and the projected points in the current frame are stored. In this process, for each vertex in the pair, the current vertex is the closest point on the mesh to the projected point stored in the previous frame. After the current point is found, the next point is projected. Then the current position is stored as the previous position, and the current velocity is stored. The current projected point is stored to find the closest point on the mesh in the next frame.

After the current points for the current vertex pair are set, the centroid is calculated. The three-dimensional midpoint $\vec{M}$ between the vertices of the current vertex pair $\vec{A}$ and $\vec{B}$ is calculated such that the components of each vector are averaged.

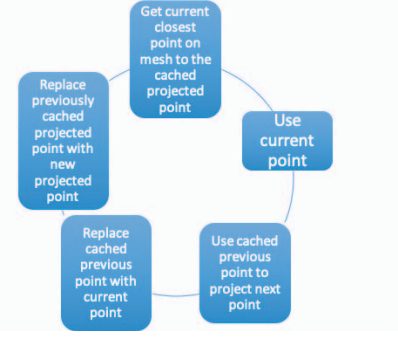


Fig. 2. The Vertex Update Cycle

Result: Joint locations are updated in each frame.

```

previousP[2]; // previous positions
currentP[2]; // current positions
projectedP[2]; // projected positions
for frame in frames do
  for joint in joints do
    for vertexPair in vertexPairs do
      currentP[i] = closest point on the
        current mesh to projectedP[i];
      projectedP[i] = currentP[i] +
        currentVelocity + (acceleration /
          2);
      previousP[i] = currentP[i];
    end
    newLocationCentroid = (currentP[0] +
      currentP[1]) / 2;
  end
end
  
```

Algorithm 1: Algorithm for updating joint locations for each frame

$$\vec{M} = \left(\frac{\vec{A}_x + \vec{B}_x}{2}, \frac{\vec{A}_y + \vec{B}_y}{2}, \frac{\vec{A}_z + \vec{B}_z}{2} \right) \quad (4)$$

The corresponding joint locator is translated to the location and this is repeated for each joint in the current frame. After each joint locator in the frame is translated to their correct position, their new locations are keyframed, thus saving their position for that frame. The pseudo-code of this algorithm is shown in Algorithm 1.

IV. DISCUSSION

We tested our method with the data set Somersault from 4DViews volumetric resources page [1]. The polygon resolution of the meshes in the animation sequence ranges from 45,306 polygons to 75,378 polygons, averaging out at 65,667 polygons per frame. There are 270 frames. It was imported into Maya 2019 as an Alembic [7] file.

Our method is still a work in progress. The accuracy of joint motion tracking slowly worsens over the course of an extended animation. For example, the position of the joint locators began to slide forward relative to their local initial position on the mesh (Fig. 3). The tracking of local positions on the mesh throughout the mesh sequence still proves to be a challenge. More sophisticated methods are needed to improve accuracy, such as skeletal fit approaches [8]. The current approach also does not take into account the biomechanics of joints. Missing are elements of rotation and range of motion. Applying these and maybe a method of volumetric bounding to restrict the drifting of the joints could prove to effectively fix current problems and improve the accuracy of the tracking [9]. In the future, we hope to implement either one or a combination of these new approaches to create a more accurate tool.

V. CONCLUSION

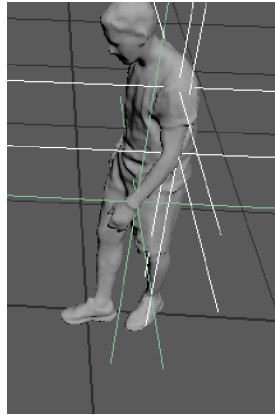
We have developed a method to track the motion of joints in volumetric motion capture datasets. The basic idea is to use the motion sequence to project the location of joints such as shoulders, elbows, and wrists at each frame. Such information can be used to conduct biomechanical analysis of human motion, such as athletic performance and rehabilitation. Our method, developed in the summer of 2019 as part of an REU Site project, is still a work in progress. We plan to integrate more advanced techniques into our method to improve its accuracy.

ACKNOWLEDGEMENT

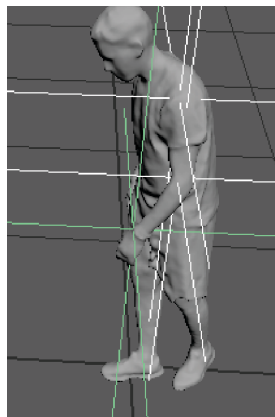
This project is supported by NSF REU Site in Immersive Media Computing (Award 1852516). Derek Roberts is an undergraduate student funded as part of the NSF REU Site program.

REFERENCES

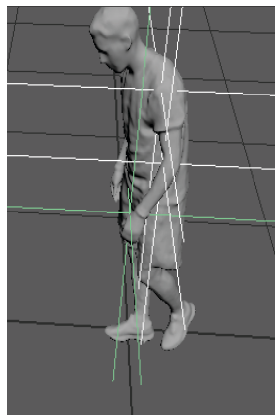
- [1] "4dviews volumetric motion capture systems." [Online]. Available: <https://www.4dviews.com/volumetric-resources>
- [2] O. van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or, "A survey on shape correspondence," *Computer Graphics Forum*, vol. 30, no. 6, pp. 1681–1707, 2011.
- [3] O. Schreer, I. Feldmann, T. Ebner, S. Renault, C. Weissig, D. Tatzelt, and P. Kauff, "Advanced volumetric capture and processing," *SMPTE Motion Imaging Journal*, vol. 128, pp. 18–24, 05 2019.
- [4] E. van der Kruk and M. M. Reijne, "Accuracy of human motion capture systems for sport applications; state-of-the-art review," *European Journal of Sport Science*, vol. 18, no. 6, pp. 806–819, 2018.
- [5] M. Zollhöfer, P. Stotko, A. Görlitz, C. Theobalt, M. Nießner, R. Klein, and A. Kolb, "State of the art on 3d reconstruction with rgb-d cameras," *Computer Graphics Forum*, vol. 37, no. 2, pp. 625–652, 2018.
- [6] S. Xia, L. Gao, Y.-K. Lai, M.-Z. Yuan, and J. Chai, "A survey on human performance capture and animation," *Journal of Computer Science and Technology*, vol. 32, no. 3, pp. 536–554, May 2017.
- [7] "Alembic file format." [Online]. Available: <https://www.alembic.io/>
- [8] J. Gall, C. Stoll, E. Aguiar, C. Theobalt, B. Rosenhahn, and H.-P. Seidel, "Motion capture using joint skeleton tracking and surface estimation," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09)*, IEEE Computer Society, 1–8 (2009), 02 2009.
- [9] S. Corazza, L. Mündermann, A. Chaudhari, T. Demattio, C. Cobelli, and T. Andriacchi, "A markerless motion capture system to study musculoskeletal biomechanics: Visual hull and simulated annealing approach," *Annals of biomedical engineering*, vol. 34, pp. 1019–29, 07 2006.



(a)



(b)



(c)

Fig. 3. From frame 1 (a) to frame 6 (b) to frame 17 (c) the locators drift forward across the mesh.