Implementation of a Low Budget, Raster Based, 3D Motion Capture System Using Custom Software and Modern Video Tracking Technology

W. Scott Meador, Carlos Morales
Purdue University

Abstract

This paper details the implementation of a system developed to generate 3D motion capture data through the analysis of raster based motion video. The system's general procedure includes acquiring video, processing the raster data to raw motion data through motion tracking technology, formatting the raw data into various useable forms using custom software, importing it to 3D animation software via custom scripts and then applying it to 3D geometry. The purpose of the project is to use the realism and efficiencies that motion capture provides, but without the high cost of traditional motion capture equipment. Though this system may not always provide the resolution or possibility for real time applications that traditional motion capture can, it does allow users to apply real-world motion to virtual objects in an efficient manner.

I. Introduction

While motion-capture techniques have been accepted by larger production companies as a cost-effective means of achieving extremely realistic movements, the technology has not gained industry wide acceptance among smaller cost-conscience firms due to the high entry-level cost associated with the process, which can start out in the tens of thousands of dollars. The cost can impact the user both in terms of the equipment required and the expertise needed to engage the motion-capture process from planning to the actual application of the data to three-dimensional geometry. This paper details a method for engaging in motion-capture in a cost-effective manner through the use of low-cost raster tools.

Production animation firms and academic institutions that can cope with the entry-level costs associated with this process benefit in numerous ways. First, they are able to produce animations with more realism than production companies that do not have access to this technology. In producing scenes for The Mummy, ILM used an optical system from Oxford Metrics to capture Arnold Vosloo's movements and map the motion to the main character. According to ILM's Jeff Light, motion capture allowed them to capture the essence of the actor's movement [1]. In video gaming scenarios, where the rendered graphics need to react to the player's movement, motion capture makes it possible to generate realistic animation. In Parasite Eve animators from Square Soft used motion capture for situations where "there is a lot of physical dynamics to the motion and you really see the gravity of the character…because that sort of movement can be really hard to achieve in keyframe animation."[2]

Second, these firms gain the ability to build reusable motion libraries that can be amortized over time. "Once we have a motion captured, we can use it as many time as we want…," says Richard Fiore [3]. High Voltage Software (HVS) used motion files captured for their NCAA Final Four video game and in the production of their proposed Pacers animated opening. After the proper skeleton has been set-up in the animation package, motion libraries can be reapplied with just a few mouse clicks. Motion capture data can also be captured at an astonishing rate. HVS captured all of the motion for their NBA Inside Drive 2000 in two days [4]. The ability to build a library of motions quickly is a tremendous financial advantage.

Third, companies gain the ability to produce live real-time animation. Donna Coco tells us that real-time animation would be impossible without the use of motion capture [5]. On the show Canille Peluche, Mat the Ghost is captured, composited with live footage, and broadcast in real-time [6]. Without the ability to capture the character's movement and render the results in real-time, it would be impossible to have the character react to actors in a live broadcast setting.

The net result is that companies who have access to motion-capture methods can produce more realistic animations faster and less costly than their counterparts. Emmanuel Javal from Medialab tells us that keyframe animation will give way to motion-capture as the prevalent mode for animation production, because it is a more cost-effective, production worthy alternative [7]. Dominique Pouliquen, marketing manager at Medialab tells use that motion capture can lower the cost of producing a 3D animated show to almost the level of doing a 2D show [8].

II. Development of System

By integrating the two dimensional motion capture capabilities of Adobe After Effects, the math functions in Microsoft Excel, and a few custom scripts, the authors were able to produce a motion capture system that could be used in cost-conscience situations and provide many of the same capabilities as some of the more conventional and costly alternatives. The central part of the system was based on using the two-dimensional tracking capabilities of Adobe After Effects to track points on video shot from stationary cameras. Because of the time needed for tracking points, then formatting and importing data, the system would not be suitable for a real-time motion capture application. Also, because the system would be limited to the resolution offered by standard NTSC video it would reduce the accuracy of this system compared to conventional motion capture systems.

In addition to the inherent limitations of NTSC video, the authors would also have to keep in mind that their method would not allow the cameras to report their settings to the capturing computer in real-time. As a result the authors would need to only employ the system in cases where small scale facial motion-capture or in full body cases where the camera would not have to pan, tilt, or zoom to track the user. This severely restricted the type of motion capture that could be done with this system in comparison to multi-camera commercial set ups.

Understanding the limitation that could not be overcome, the authors focused on setting up samples for both facial motion capture using a single camera and for full body motion capture using two cameras. For facial-motion capture a single camera was pointed onto a subject's face. The subject was marked up using a series of colored dots at strategic points (see figure 1) Two important markers were the dot between the eyes, and the one on the nose. These features do not move on a person with regard to their facial expression. Therefore these could be used for calibration during shoots on different days.

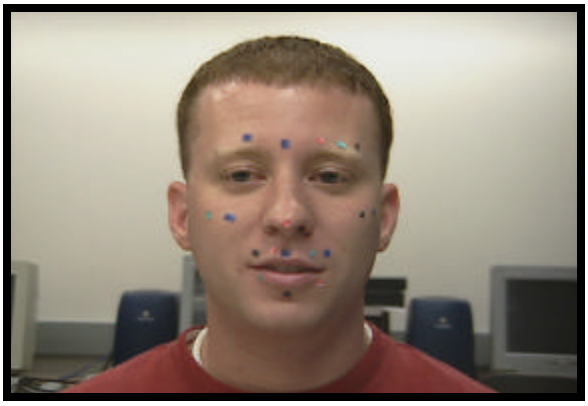

Figure 1. Subject with Markers

For full body motion capture, two cameras were set up at ninety degrees to each other facing the subject. Both cameras were leveled on a tripod and zoomed to the same degree and were the same distance from the center of the subject. This would insure that image captured through both cameras would be in the same scale. This would be critical during the post-processing of the frames.

Additional considerations at this point were the size of the dots used in relation to the image or purpose of the shoot. For optimal processing the authors found that the dots should be at least10 pixels squared. Thus, for facial motion capture, when the camera would be relatively close to the face, the dots would have to be physically smaller than when used for full body motion capture. Thus, the intended use for the motion-capture data dictated the size of the markers used.

In shooting the video, the authors were concerned primarily with the effects of the quality of image provided by the camera and the digitizing process on the end motion-capture data, and the lack of synchronization between the cameras. Because the cameras used did not have a data feed to the computer reporting their position or timing information, the authors used a physical placard to manually synchronize the events. Before shooting the actual subject, the camera would be locked down and the authors would insert a placard in the middle of the scene that would be captured by both cameras. This would allow the authors to locate the same point in time later by examining the video.

The quality of the video would also affect the quality of data. The authors found the resolution and image fidelity provided by the MiniDV (720 pixel by 480 pixels) format to

be more than sufficient for the task. The five to one compression imposed by the MiniDV format did not affect the ability of the computer to find the location of the markers during the post-processing session. The authors used a Canon XL-1 and a Canon GL-1 for the video shoot. The video was then digitized using an OHCI IEEE 1394 port on a laptop computer.

Processing the video required using the two dimensional tracking features of Adobe After Effects 4.1 Production Bundle. One of the features of this software allows the user to track a feature on a series of consecutive frames and then apply those tracking data to a second layer. The authors found this could be exploited to perform the bulk of the work for the motion capture method. First, the frames were imported as an animated sequence and resized from 720 by 480 to 720 by 540. This was necessary because the NTSC video provided by the DV capture process has a pixel aspect ratio of .9. Using the image at that resolution would produce data that was compressed in the x-axis by ten percent. By resizing the image to 720 by 540 the pixel aspect ratio is forced to 1 and the produced data has the same scale in the x-axis and the y-axis. This re-sizing process was repeated for the video from both cameras for the full body scene. Next, the video from both cameras was synchronized by trimming out all of the frames preceding the point of contact on videotaped placard.

With the video synchronized, the authors then used the tracking functions of After Effects to produce numerical data to represent the movements of the markers. A four pixel square composition was produced for each point that needed to be tracked. This was necessary because After Effects must have a target to apply the data after it has tracked a feature. The authors tracked only the position of each point on each image and then applied it to one of the "dummy" four-pixel square compositions. This generated positional keyframes for each four-pixel composition at each frame. The authors then manually selected all of the keyframes and pasted them into a text document (see figure 2). On the full body shoot the authors performed the same operation on the frames coming from camera one and from camera two. Camera one provided the x and y data for each point. The authors discarded the y data from the tracking information generated from camera two's images and used the x data as the z coordinates for final 3D motion capture data.

Adobe After Effects 4.0 Keyframe Data

	Position	
Frame	X pixels	Y pixels
0	349	260
1	349	260
2	349	260
3	349	260
4	349	260
5	349	260
6	347	262
7	347	262
8	347	262
9	347	262
10	347	262
11	347	262
12	347	262
13	347	262
14	347	262

Figure 2. Raw After Effect Motion Data

At this point, the authors had all of the raw data required for applying motion capture within a three dimensional animation program, but it would have to be formatted in manner that could be read into some of the industry standard animation packages. The authors decided to standardize on the biovision segmented data format also known as BVA [9]. This format was readily readable into SoftImage 3D, Pixel 3D, Electric Image Animation System, and numerous other packages. With relatively few modifications it could also be read into 3D Studio Max, Maya and even LightWave 3D. The format carries x,y,z translation, x,y,z rotation, and x,y,z scale for each point. The file always carries a header identifying these properties and each piece of information is separated by a TAB character (See Figure 3). LightWave 3D's format was very similar to BVA. The only changes that had to be made to the LightWave format were (1) the headers had to be removed, and (2) the z-axis translation would have to be multiplied by negative 1 [10]. For Max and Maya, custom scripts would have to be written to read the BVA format.

Segment: Chest
Frames: 3
Frame Time: 0.033333

XTRAN	YTRAN	ZTRAN	XROT	YROT	ZROT	XSCALE	YSCALE	ZSCALE
INCHES	INCHES	INCHES	DEGREES			DEGREES		DEGREES
0.0225	0.57	10.0837	0	0	0	1	1	1
0.0332557	0.517897	10.0526	0	0	0	1	1	1
0.034022	0.515758	10.02118	0	0	0	1	1	1
0.0347978	0.513586	1.989462	0	0	0	1	1	1

Figure 3. Biovision BVA Motion File

Initial testing was done in SoftImage because it imported BVA files directly. Upon importing the data, the authors found the animation playing in quadrant one of the coordinate axis, but all of the animation was upside down. This occurred because the origin in After Effects is in the upper left corner. The x-axis increases to right and the y-axis increases down. In most three dimensional animation packages, the origin point is the middle of the world. The x-axis increases to the right, but the y-axis increases in the up direction. The authors found two ways to fix this. One, they could group the data and mirror along the x-axis to reverse the direction of the y-axis. A second solution was to import the data into Microsoft Excel before formatting to the BVA standard and (1) subtract 540 from all of the y translation values, and (2) multiply the resultant y points by negative 1. The first step would place the animation below the x-axis into quadrant four and the image would still be reversed. The second step would mirror the motion around the x-axis back into quadrant one, fixing the orientation of the motion relative to the three dimensional world. This second solution was the preferred method because it made sure that all data was properly oriented before going into the BVA format. Option one would have to be performed every time the motion was imported into the software.

After testing the tracking system with SoftImage, the authors' next goal was to be able to use the motion in 3D Studio MAX, which is their primary 3D teaching tool. 3D Studio MAX has a built-in scripting language called MAXScript. MAXScript provides the ability to script most aspects of the program's use, such as modeling, animation, materials, rendering, and so on. It also has the ability to build custom import/export tools

using the built-in file input and output (I/O) [11]. With this functionality, the authors were able to quickly write a script to read the standardized BVA file format.

As mentioned before, the BVA file format has headers that provide the information about each marker that was tracked. Each marker is called a segment, and under each new segment, the file contains the number of frames that were tracked for that segment. After documenting the number of frames, the file lists the transformations of the marker for each frame. A TAB delimits these transforms. Transformations include; translation in the x, y, and z-axis, rotation in the x, y, and z-axis, and scale in the x, y, and z-axis. The script reads each line of the file looking for the key word "segment". When the word is found, the script creates a new null, or dummy, object in the 3D scene and then names the object based on the segment name. The next step is to read the number of frames that have been tracked. The number of frames in the time control of MAX is changed to reflect the frame number from the BVA file. Once the null and scene animation length have been set, the script skips over the text that notes what each column of numbers represent and begins to read the numbers. At the first line of numbers, the script sets MAX to record key frames. It then reads the x, y, and z-axis translations and sets the current translation of the null to those values. Because the video tracking process does not take into consideration the rotation and scale of the tracking points, the script disregards the BVA file's rotation and scale information by skipping to the next line in the file. The script increments the current time in the MAX scene and then reads the new translation values in the BVA file. It loops through this process until it reads the word "segment" again. Once a new segment is detected, a new null is created and the process continues until all of the segments are read.

To make the script more useful for many users in a production or academic environment, a user interface was added so no file names were hard-coded into the script. The user has to simply evaluate the script in MAX and then press a button to choose which BVA file they want to import. Having an interface also allows the user to set the size of the null object in the scene. During the creation process of this script the authors were reminded that different 3D applications have different default coordinate spaces. Some applications use a z-up coordinate system, while others have a y-up coordinate system. To keep with the standard set by the BVA format, we used a y-up coordinate system. Unfortunately, MAX uses the z-up system so our animations came in on their sides, or 90 degrees off axis. To correct this issue the script's user interface was given radio buttons that allow the user to choose whether the motion should be imported with a z-up or y-up orientation.

Once all of the nulls are created in the MAX scene they can be applied to any type of geometry for animating. MAX provides several options for linking objects together, but two are outlined in this paper. The first option is a strait linking of the null object to another object in the scene. An object, such as a sphere could be linked to each null to give the motion a form when rendered. If actual character control were desired then linking inverse kinematics (IK) bones from a character rig to the nulls would be the answer. The second option is using the linked xform modifier to selected vertices of the geometry. MAX has a tool called soft selection, which gives user-definable falloff to the

selection. Falloff will keep the selected vertices from all moving the same amount, which would cause the deformation of the geometry to be unnatural. As mentioned earlier, the video tracking process does not take into account the rotation of the markers. Since there is no rotational information, the user cannot simply skin geometry with the nulls. There would be no joints by using a strait skinning method. Linking to a bones system is the most logical way of overcoming this limitation. Other motion capture file formats like Biovision's BVH format store actual hierarchies and rotational information. BVH was created for conventional motion capture systems that can take 3D translation information and by using limitations set by a hierarchy, or boning system, can derive rotational information.

III. Conclusions

The authors set out to create a system that was very inexpensive to use compared to conventional motion capture systems. The system works as expected, but is not very automated. Conventional systems can offer real-time motion capture, output complex data to standard file formats, and in many cases provide a large capture area with a high amount of accuracy. The system outlined in this paper offers no real-time opportunities, works with simple data that is not as accurate, and requires a lot of seat time on software to get the data. The primary advantages to the system over conventional systems is its extremely low cost, it does not require proprietary equipment or specialized spaces to capture in, or the need for people with specialized skills in motion capture technology. Though certainly not as fast at generating data, this system would allow any production facility or academic program the ability to grow a library of useful motion.

Almost any production studio or academic program could use this system to create motion capture data. The video equipment, and software needed is inexpensive and if not already in inventory, the equipment can be purchased through many vendors. No specialized training is needed to perform the steps needed for gathering the data, although it may take some time to get used to getting the tracker in Adobe After Effects working simply because it is a function of the software that is not used as much as other parts of the software such as transformations, filters, or masks. What users will notice immediately is that the process of tracking each point is time consuming and cannot be automated. Unfortunately, this may lead some users to use fewer markers as they really need for useful motion capture data.

Efficiency becomes an issue in any production environment. The fastest stage of this process is the video capture stage. Video capture is a standard procedure that is done often at production studios and academic programs though it is usually done for video editing and compositing rather than motion capture. This system is not very efficient for large-scale projects where speed is needed because of the large amount of time needed in manual post processing. The MAX script is completely automated, but may take some time if there are several nodes because it must process each marker individually in a linear manner.

At this point in developing the system, there is no intelligence built in to the tracking process for interpolating points that were obscured during the capture process. This is generally not a problem for facial motion capture, but for full body motion it could leave the user with a lot of "clean up" work after applying the motion to his/her objects. Markers may get obscured when a subject moves his/her arm or leg behind the other, or twists his/her body enough to put the marker out of sight of the camera.

IV. Future work

Working on the slow stage of tracking the markers will be the first step in increasing the usability of this system. Other raster tracking technologies beyond Adobe After Effects are being researched. One technology that shows promise is using built-in functions in Macromedia Director. It allows the programmer to get pixel color and coordinate information from the stage. By polling the color of a pixel or group of pixels and tracking the coordinate change of those pixels over multiple frames, Director can be used as a motion tracking application.

Since the authors would be writing their own tracking software using an available software platform that was not originally intended for that use, the authors would also have to include code that would logically determine the coordinates for markers that might get obscured by the motion of the subject. Other low-cost off-the-shelf software such as Discreet's Combustion provides a robust tracking algorithm that takes into account markers that get obscured and then reappear, but it cannot be automated, so it cannot be used.

After the authors are satisfied that the tracking process works as well or better when automated as it did manually, the next step in development is to test it in the field. In the academic environment, we plan to have several student groups individually setup video capture sessions and then process their video with the automated tracking tools. The script will also be ported to Maya's MEL scripting language to increase the number of possible users of the system.

Another possible extension, is to automate this process through a web enable application that not only analyzed the images and generated to motion capture data, but also archived the results in a reusable library. This would accelerate the creation of motion capture libraries and would create a better return on investment.

Bibliography

1. Chronopolus, Emilio. Personal Interview. *Character Animator*. NuFX, Rolling Meadows: Illinois, 1999.
2. Baumgartner, Henry. How to Catch a Ghost. *Mechanical Engineering*, Vol. 121, p108, April 99
3. Fiore, Richard. Personal Interview. *Art Director*. High Voltage Software, Hoffman Estates: Illinois, 1999.
4. Fiore, Richard. Personal Interview. *Art Director*. High Voltage Software, Hoffman Estates: Illinois, 1999.
5. Coco, Donna. Motion Capture Advances. *Computer Graphics World*, Vol. 20, Issue 11, p37, Nov97.
6. Sturman, David, A Brief History of Motion Capture for Computer Character Animation. SIGGRAPH 94, 1994.
7. Robertson, Barbara. Bad to the Bone. *Computer Graphics World*., Vol. 22, No. 5, p34 May 99.

8. Robertson, Barbara. Bad to the Bone. *Computer Graphics World*., Vol. 22, No. 5, p34 May 99.
9. Biovision Incorporated, Biovision Motion Data Specifications. http://www.biovision.com, 1999.
10. Newtek. LightWave 3D 5.5 Users Manual. Austin, Texas: Newtek, 1998.
11. Discreet. 3D Studio 3.0 Users Manual. San Rafael, CA, 1999.

W. SCOTT MEADOR
Scott Meador is currently an assistant professor in the Department of Computer Graphics Technology at Purdue University. He received his B.S. from the University of Central Arkansas in Technical Theatre and M.S. in Technology from Purdue University. Scott specializes in lighting and rendering in 3D, but also teaches animation and scripting. He is the co-director for the newly formed Purdue International Center for Entertainment Technology and outside the university he has a computer graphics design firm that produces graphics for live events such as industrial shows, theatrical productions, and concerts.

CARLOS MORALES
Carlos R. Morales is an assistant professor of computer graphics in the Department of Computer Graphics Technology at Purdue University. He holds a BA in Telecommunications and an MS Ed. in Curriculum and Instruction. Prior to working at Purdue University, Carlos worked as a Technical Director. Some of his clients have included Microsoft, Chicago Bulls Organization, First Alert and Brach's Candies. His research interest includes distance learning, animation, and multimedia development.