Camera-Based 3D Motion Tracker for Interactive Computer Graphics Animation

Alejandro Cornejo and Maria Elena Algorri

ITAM, Sistemas Digitales, Rió Hondo No. 1, Col Tizapán San Ángel,
México D.F., CP. 01000
acornejo@gmail.com, algorri@itam.mx

Abstract. We built a system for interactive computer graphics animation based on real time 3D motion capture. The system uses a high-precision, 3D motion sensor developed using infrared vision. The sensor consists of an array of low-cost, high-performance webcams modified to detect only infrared emitter LEDs. The webcams are queried simultaneously via a FireWire communication channel by a Linux-based PC using custom developed drivers. The webcam images are processed in real time to detect the position of the infrared LEDs. The 3D position of the infrared leds is triangulated by aligning and registering images from all webcams. The 3D infrared LED positions are then processed by an OpenGL program to interactively animate characters using inverse kinematics. The technology has been used successfully to develop a museum exhibit where computer characters mimic real movements, and we are currently aiming at using it for high-precision (under 1mm), real time 3D localization applications.

Keywords: Camera-based sensors, 3D movement tracking, Open GL character animation, triangulation, simultaneous camera capture.

1 Introduction

Animation is a deliberately interpreted "illusion of life"[1]. We are all aware of the increasingly realistic computer-generated human motion that abounds in movies, advertisements and especially games. This natural-looking appearance in many cases is attributed, as it has been for years, to the fine skills of professional animators. But more and more these realistic motions also involve the use of motion capture [2]. Much of the drive behind the progress of computer graphics in the past decades has been the possibility of creating ever more life-like animations. Although complex geometry and high-end rendering can produce impressive computer models, nothing beats the allure of a physically based animated model.

The use of 3D animated computer graphics has quickly gone from being reserved to an elite of high-end computers (mainly SGI and super workstations in the mid-90's) to a must-have requirement for current desktop computers. Graphic cards are also the first component in a computer to become obsolete.

The development of 3D animated computer graphics has followed a similar trend. Nowadays, any motivated student can create realistic 3D interactive applications with APIs such as OpenGL. But the development of high-end graphics such as human-like computer animation, real-time physically based animation, and real-time motion

based animation (also known as computer puppetry [3]) is still reserved for big production studios in the film and game industries. This restriction is due, in part, to the fact that the creation of life-like animations requires much more than just a top-of-the-line graphics card and the use of graphic libraries. Character animation at the professional level starts in sophisticated motion capture laboratories [4] where motion markers can be attached to a human actor. The movement of the actor is then stored in movement databases that can be edited and mapped to computer characters using techniques such as Inverse Kinematics (IK). Though published research on the field of human-like animation is abundant, the cost of precision motion capture sensors is usually unaffordable for smaller research laboratories.

Some years ago motion capture was only used for face animations, to create faces that could mimic the facial expressions of actors, these days motion capture is used to control an entire animated character, from facial expressions to all his skeleton joints.

Recently, computer games have introduced motion capture devices as interfaces to increase the realism of their 3D games. The infrared sensor-based "mocap" games Mocap Boxing and Police 911 by Konami are an example [5].

In the area of interface techniques for controlling avatars, vision-based motion-capture interfaces are appealing because they allow the user to move unencumbered by sensors. [6] and [7] have used motion capture data to develop mappings for reconstruction of 3D pose and motion from video.

In our laboratory we have developed a complete, high precision, low cost system for realistic animation of human-like computer characters in real time. The system consists of a motion capture sensor based on infrared vision that we built with off the shelf components. We have also developed the vision system, from interface drivers to the motion capture sensor to real time image processing routines for motion tracking. Last, we have developed an OpenGL environment to map the coordinates of the motion trackers to a fixed-length articulated skeleton using IK. We present the results of using the system to develop a museum exhibit where a computer character mimics the movements of the visitors.

2 Technology Evaluation for Motion Capture Sensors

There are several companies devoted to the manufacturing of high performance motion tracking equipment [8][9][10]. When it comes to the technology that supports these systems, there are a great number of options. Most systems use either optical[11] or electromagnetic[12] devices, each technology has its advantages and downfalls.

With motion capture systems based on optical technology, it is possible to achieve a very high time and space resolution, but occlusion of the emitters is very common because a line of sight between the emitters and the sensors is required. Electromagnetic systems rely on magnetic fields which go through all non-metallic objects and therefore occlusions do not occur (unless a metallic object is between the sensor and the emitter). However, electromagnetic systems require rather large emitters depending on the volume that the magnetic field must cover, and both the emitter and the sensors are active devices. When using optical technology light

emitters can be very small, and some systems based on color or form detection support the use of passive "emitters".

Despite the increased availability of motion capture systems, their price restricts their use. The cost of some of these systems exceeds 100,000 USD, depending of their sensing range, speed and precision.

In our laboratory we had the chance to use and evaluate a low end Fastrak electromagnetic motion capture sensor from Polhemus priced at roughly 3000 USD. The Fastrak has a spherical sensing range of about one meter in diameter and millimetric resolution. After careful consideration of the pricing, performance, maintenance and durability of the Fastrak we decided to build an optical sensor that would have comparable accuracy and speed but at a fraction of the price.

3 Methodology

We built a complete system for character animation form motion capture. We next describe all the stages involved in the system: The optical motion capture sensor, the interface drives, the camera and emitter characteristics, system calibration, registration, emitter tracking and IK for character animation.

3.1 Motion Capture Sensor Architecture

We use a scalable architecture for the motion capture system. The system consists of n CCD sensors (webcams or videocameras) that "sense" or see m infrared (IR) emitters. Depending on the required spatial accuracy, speed or robustness to emitter occlusion, n will be established, where the minimum n (number of cameras in the system) is 2 so that 3D information can be obtained from triangulation. Speed and spatial accuracy tend to be inversely proportional, so we had to optimize every stage in the process to support more than 2 cameras maintaining real time (under 0.1 s delay) performance.

Each CCD sensor communicates with a PC via a FireWire interface. FireWire was chosen due to its wide bandwidth and isochronous transfer scheme. Hi-Speed USB ports were also considered, but one of the key advantages that FireWire offers is the availability of several standard protocols for video transfer (defined in the IEEE 1394 standard), what makes it device independent.

There are two different standards for transmitting video across FireWire, DV (Digital Video) [13] and DC (Digital Camera) [14]. The two standards are different, mainly because DV is a standard to transmit compressed video (and defines the compression and decompression codec) and DC is a standard for transmitting uncompressed video. DV is used in devices which store the video locally and DC for live video feeds. Currently the system supports DV and DC compliant devices, but DC devices are preferred.

3.2 Driver Architecture

Because we needed to be able to communicate simultaneously with *n* webcams, we had to develop custom drivers to support both DV and DC FireWire devices. Both drivers are thread safe and capable of capturing live video feeds at 30 FPS of two and

up to 63 webcams simultaneously (FireWire supports up to 63 simultaneous communication channels).

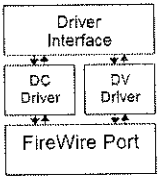

Fig. 1. Driver Architecture

The driver queries the FireWire ports to find out what video capable devices are connected to each port. When it finds webcams or video cameras that comply with the DC or DV standards, it sends special packets to ask each device to transmit on a different channel (with commercial FireWire drivers all devices default to broadcast channel 63). After each device has been set to transmit on a different channel, the driver continues to poll the FireWire device, decodes received packets, decompresses the information (only from DV devices) and reconstructs the video images. Driver architecture is schematically shown in Fig. 1.

3.3 Webcams vs Video cameras

One choice of CCD sensors was to use video cameras. We tried 2 top-of-the-line Sony Handycams connected via FireWire with the DV driver. Although we where able to capture video feeds from the two cameras simultaneously in real time, the video decompression proved to be very taxing on the processor. Since all video cameras use the DV standard and their price is moderately high (over 500 USD), we did not pursue them as CCD sensors.

We then used a pair of cheap (under 100 USD) ADS FireWire webcams that were capable of capturing uncompressed video at 640x480@30fps with 24 bit color depth. Simultaneous capture of the two DC cameras was successful.

There is a tradeoff between the resolution of the CCD sensors and the time required to capture and process the images. When using DV devices we can have a higher image resolution, but at the cost of more processing power.

3.4 Camera Arrays

To be able to recover 3D motion from the 2D CCD sensors we need at least two cameras. Increasing the number of cameras increases the precision and tolerance to occlusion because, with each camera that is added to the system, the viewing volume increases, and therefore the probability of occlusion decreases. All cameras must be align, and their optical axis must be parallel.

Each pair of cameras constitutes a stereo pair for which a system of simultaneous equations can be derived to triangulate the 3D position of an emitter that is seen by both cameras. If an emitter is seen by more than two cameras there are more stereo pairs to calculate its 3D position and thus reduce the localization error. For n cameras there are

$$C_n^2 = \frac{n}{2}(n-1) \tag{1}$$

different stereo pairs. However, there is a practical limit to the number of cameras that the system can support. If the cameras are positioned in a linear array, the viewing volume where an emitter is visible by all cameras decreases and moves further away from the sensors (reducing the accuracy) with each camera added[15]. The effective viewing volume consists of the intersection of the viewing volumes of every camera in the system; only emitters within this volume can be tracked by all stereo systems.

3.5 Calibration

In order to triangulate the 3D position of the emitters using only the data from the CCD sensors, it is necessary to know the camera's intrinsic and extrinsic parameters. The intrinsic parameters include the focal distance, pixel size and pixel aspect ratio. Extrinsic parameters describe the geometry of the camera array (linear vs orthogonal), the distance between the cameras and the angle and orientation of each camera.

Without camera calibration it would be impossible to achieve reliable sensing results. However, if the cameras are placed in a fixed array, the intrinsic and extrinsic parameters do not change over time, and calibration needs only be done once.

There are several approaches for camera calibration; they are all based on the same principle: With a camera array one can reconstruct the 3D position of a physical point from two or more different projections of the point (obtained from two or more different images of the point taken by the cameras in the array). The inverse process occurs with camera calibration: from the known 3D position of M points in space, and their 2D projection onto two or more camera images, it is possible to calculate the camera parameters that minimize the error between the real 3D position and the triangulated position. (see [16] and [17] for more details). Camera parameters are important because they are used for triangulating the 3D position of IR markers as explained in 3.8

3.6 Infra Red Considerations

The CCD sensors used in cameras are sensitive to the electromagnetic waves in the spectra of visible and IR light. Their peak sensitivity lies somewhere in between these two frequency spectra (usually closer to the visible light spectrum). However, CCD sensors used for consumer video products are modified with IR filters that block IR waves and only allow waves within the visible spectrum to go through.

For the IR vision system, we had to remove the IR filter from the CCD sensors of the webcams and substituted it instead with a filter that blocks all visible light and only allows IR waves to go through.

On the emitter side, we had the choice of using active or passive IR emitters. Active emitters transmit using coded schemes that allows the sensor to identify each emitter distinctively. This also implies that active emitters need some kind of electronic circuitry that requires them to be powered and synchronized to transmit different codes each.

Passive emitters do not actually emit but rather reflect IR light. Therefore, they require a strong IR source to be pointed at them, and the reflected IR waves are then captured by the sensor. We will call these passive devices IR markers. IR markers are very cheap, but do not allow unique identification. In our system we use IR markers.

There is also a hybrid type of emitter, which does emit IR waves, but does not use any electronic circuits except a power source (usually a watch battery). These type of emitters do not require an external IR source pointed at them, but are not distinguishable.

3.7 Marker Registration

The data from the IR sensors (the webcams) is captured, and for every image frame, processing is required to locate the 2D projection of the IR markers. The webcam images are first filtered using a 5x5 Gaussian kernel to remove any unwanted noise. Because the images were acquired with a filter that only allowed IR information to go through, we use a simple threshold filter to separate the image regions corresponding to the IR markers from the rest of the image background. At this point, the images only contain blobs that represent the 2D projections of the markers. Fig 2. shows an example of an image from the IR markers after filtering and thresholding.

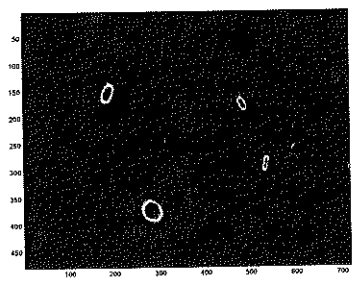

Fig. 2. Image of 4 IR markers after filtering and thresholding

The images are scanned in search of blobs, when a blob is found a modification of the scan line flood-fill algorithm is run on each blob. Linear prediction can be used to avoid unnecessary processing in subsequent frames [18]. The flood-fill algorithm can segment all the pixels that belong to a given blob without visiting a pixel more than once. We store the position and intensity of the pixels in each blob. To calculate the exact position of the IR marker in the blob with subpixel accuracy it is possible to either calculate the local maximum of each blob using cubic interpolation between adjacent elements, or to calculate the center of mass of the blob, the latter was chosen.

3.8 Triangulation

In order to set up the triangulation equations to calculate the 3D position of the IR markers from two camera sensors we need to study the geometry of the camera array. We first need to select a stereo system from the camera array, since a 3D position will

be calculated for each stereo system available [19], and the results will be then averaged to get a final position of the markers.

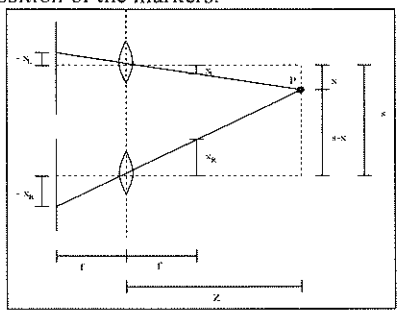

Fig. 3. Geometry of a Stereo System

In Fig. 3 we are trying to calculate the coordinates of point P, which are X, Y and Z. We are looking at the cameras (represented in Fig. 3 by a fisheye lens) from above, and therefore we can only see X and Z, however when looking at the system sideways we also get similar triangles for Y and Z. In Fig. 3 we observe two similar triangles in the camera geometry. We can exploit this relationship to derive a system of equations to solve for the 3D coordinates of the markers. The coordinates that we have are x_L, y_L, x_R and y_R, which are the projected coordinates of point P on the Left and Right camera respectively. From the similar triangles in the sideways system we solve for Y for both cameras to get:

$$Y = y_L \cdot \frac{Z}{f}, Y = y_R \cdot \frac{Z}{f}, \therefore y_L = y_R \tag{2}$$

So, at least theoretically, the projected Y coordinates in both cameras must be the same, which is logical since both cameras have the same position on the physical Y axis. This fact can be used to measure the error in the camera arrangement. We have been assuming that the camera array is setup perfectly parallel on the X axis, and that s (the horizontal separation between the cameras) is known. Similar equations can be solved for the X coordinate. f is a new parameter that we introduce; it represents the focal distance of each camera measured in pixels. This distance is assumed to be the same for all cameras, the assumption is reasonable considering that the cameras used are identical. To calculate f we go through the process of camera calibration described in 3.5. Calibration is only carried out the first time the system is setup. When more than 2 cameras are available, the triangulation process is repeated over the rest of the stereo pairs, and the solutions to the simultaneous equations are then averaged to reduce error.

3.9 Inverse Kinematics Solver

We used a 2 camera motion capture sensor to animate the arms of a huma-like character (a gorilla) in an interactive museum exhibit. This was a difficult problem, since we only had 2 motion sensors, one for every hand and we needed to articulate the whole arms of the gorilla so that its hands would match the position of the sensors. In order produce real time, realistic animation of the gorilla's arms, an IK system to solve the position of all the intermediate arm joints having only the end effectors (the movement of the hands) was needed.

For this particular application, the arms of the gorilla were modeled with 2 joints, one at the shoulder and one at the elbow, the wrist was left fix, so a two joint IK solver was sufficient. The first joint was a 2 DoF joint and the second joint only had 1 DoF. Following the natural movement restrictions of the shoulder and elbow, we set the angle restrictions of the joints as follows: For the first joint one of the axis can have an angle between 0° and 360° the second angle must be between 0° and 90°. The angle of the second joint is restricted to vary between 0° and 180°. The 2 joints described can be modeled as a hemisphere and a disc.

The position of the hemisphere is fixed at a point, while the end effector (corresponding to the hand) is tracked by the system. The IK solver needs to calculate 3 angles in order to position the middle joint and also has to calculate the necessary rotations for the hand to be in place. The distance between the first and second joint is fixed, and it is a known variable, therefore it is known that the position of the second joint is restricted to the surface of an imaginary hemisphere with radius equal to the distance between the joints. As shown in Fig 5. the position of O (corresponding to the shoulder) and P (the hand) is known, while T (the elbow) , the rotations of the vector OT and vector TP need to be calculated.

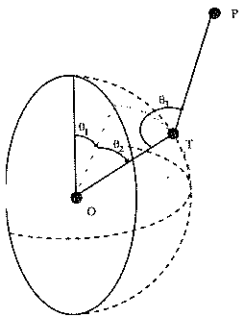

Fig. 4. IK solver diagram

The first angle θ_1 is calculated by projecting point P (which is delivered by the motion tracking system) onto the plane of the hemisphere. The plane of the hemisphere is the ZY plane. A projector plane is defined as V_{zy}, and $V_{zy}P$ (the projection of P onto plane ZY) is calculated, this result is labeled as P'. To calculate θ_1 the dot product between the x coordinate and P' is used.

After calculating angle θ_1, point T is restricted within an arc on the surface of the hemisphere. The next step is to calculate θ_2, which is divided in two components. The first component can be calculated using the dot product between P' and P.

The second component can be calculated by observing the triangle formed by OTP, the position of T is not yet known, but the length of each side of the triangle is known. Using this triangle the second component of θ_2 as well as θ_3 can be calculated.

4 Practical Implementation

For the gorilla application (3.9), the length of the skeleton to be animated is fixed, however, the human that will carry the markers has an unknown geometry. The interactive application is placed in a museum where very different people, from children to adults can interact with the computer animated character.

As explained in 3.9, the two joint IK solver is used in conjunction with the motion capture system to animate the gorilla's arms. The system has to be calibrated for each person, for this purpose, the system needs to know the position of the user's shoulders and the length of his arms. Before the user starts to play with the gorilla, he is asked to touch his shoulders and stretch his arms. The whole system works even when markers are only placed on the hands of the player.

5 Results

The system gave very accurate results for movement, and as long as the individual made natural movements the animated character mimicked the human accurately.

Better results could be achieved if extra markers were attached in key places, such as the elbows, what would also simplify the IK solvers. However, it was decided to use the least number of markers possible, since the probability of occlusion is proportional to the number of markers. The markers were placed at the individual's finger tips to further avoid occlusion.

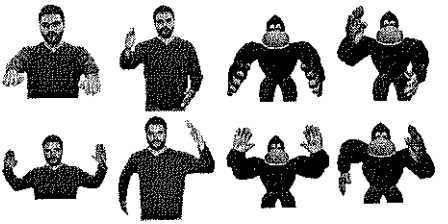

Fig. 5. Animated character mimicking the movements of the user

The system was tested at the laboratory and a coarse precision of 3 mm was achieved at a distance of 1 m from a single stereo pair. The IR markers can be observed at the tip of the test subject's fingers as small white balls, in this case, simple IR reflecting spheres were used as markers. The sensors (webcams) are placed directly in front of the visitor, looking down at an angle of 30°.

The animated character is then rendered and projected in a curved circular screen. The visitor stands in front of the screen and interacts with the animated character. The character and the subject are shown in Fig .6

The whole system, including CCD sensor capture, image processing, emitter positioning, 3D triangulation, IK solvers for two limbs and the rendering of the character run on a single Linux-based PC and real-time interactivity is achieved.

The position of the markers is refreshed at 30 Hz, and the precision proved to be more than enough for interactive applications such as the one in the museum exhibit.

6 Conclusions

In this paper we present a complete, low cost solution for computer character animation from motion capture. Our system delivers comparable performance to many low end commercial systems at only a fraction of the cost. Although the processing pipeline in the system, from IR sensing, to image processing and IK solving is long, it is optimized for real time motion capture: character animation can refreshed up to 30 fps what makes it useful for interactive applications.

When used in critical applications where precision is a key factor, the system can be improved by simply adding more cameras, without modifying the underlying architecture. The system will not slow down significantly with the addition of more cameras, and the processing can be distributed across multiple PCs if required.

It would be possible to program the emitter registration and the triangulation into a DSP, therefore having a complete external solution to the motion tracking problem. Using a DSP would eliminate the speed concerns completely.

The system would also benefit from more sophisticated IK solving. Although an arm can be modeled by the system described, there are more accurate models to mimic the movements of a human arm. A human arm involves more restrictions but also more DoF in some joints.

We are currently working on a high precision (under 1mm) version of the motion capture sensor to be used in medical applications. The new sensor uses 2 stereo pairs of cameras and was developed using the same methodology described in this paper.

References

[1] G. Cameron, A. Bustanoby, K. COPE, S. Greenberg, C. Hayes, O. Ozoux, "Motion Capture and CG carácter animation", SIGGRAPH, Intl. Conf. on Computer Graphics and Interactive Techniques, pp. 442—445, 1997

[2] M. S. Geroch, "Motion Capture for the rest of us", The Journal of Computing in Small Colleges, Vol. 19, No. 3, pp. 157—164, 2004

[3] H. J. Shin, J. Lee, S. Y. Shin, "Computer Puppetry: An Importance-Based Approach", ACM Transactions on Graphics, Vol. 20, No. 2, pp. 67—94, April 2001

[4] V. B. Zordan, J. K. Hodgins, "Motion capture-driven simulations that hit and react", ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 89—96, 2002

[5] J. Lee, J. Chai, P. Reitsma, J. Hodkins, N. Pollard, "Interactive Control of Avatars Animated with Human Motion Data", SIGGRAPH, Intl. Conf. on Computer Graphics and Interactive Techniques, pp. 491—500, 2002

[6] R. Rosales, V. Athitsos, L. Sigal, S. Sclaroff, "3D hand pose reconstruction using specialized mappings", IEEE Conf. on Computer Vision, pp. 378—385, 2001
[7] M. Brand, "Shadow Puppetry", IEEE Intl. Conf. on Computer Vision, pp. 1237—1244, 1999
[8] Polhemus
http://www.polhemus.com
[9] Northern Digital Inc. (NDI)
http://www.ndigital.com
[10] Ascension
http://www.ascension-tech.com
[11] Optotrak from NDI, http://www.ndigital.com/optotrak.html
[12] MotionStar from Ascension,
http://www.ascension-tech.com/products/motionstarwireless.php
[13] International Electrotechnical Comission, http://www.iec.ch/, IEC 61834 and IEC 61833
[14] 1394 Trade Association, http://www.1394ta.org/, IIDC 1394-based Digital Camera Specification Version 1.30
[15] E. Aitenbichler, M. Muhlhauser, "An IR Local Positioning System for Smart Items and Devices", International Conference on Distributed Computing Systems ,Workshop, May 19-22 2003
[16] F. Devernay and O. Faugeras. "Automatic Calibration and Removal of Distortion from Scenes of Structured Environments", In Proc. of SPIE'95, 1995.
[17] A. M. Kushal, V. Bansal, S. Banerjee, "A simple method for interactive 3D reconstruction and camera calibration from a single view", Proc. Indian Conference on Computer Vision, Graphics and Image Processing, 2002
[18] M. Ribo, A. Pinz, A. L. Fuhrman, "A new Optical Tracking System for Virtual and Augmented Reality Applications", IEEE Instrumentation and Measurments Conference, May 21-23 2001
[19] Norikazu Ikoma, Wataru Ito, Toru Irie, and Hiroshi Maeda, "3D Reconstruction from Stereo Camera Dynamic Image based on Particle Filter", Proc. of Fourth International Conference on Intelligent Technologies (Intech'03), December 17-19, 2003, Chiang Mai, Thailand, pp.394-403 (2003).