(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
12 September 2008 (12.09.2008)

PCT

(10) International Publication Number
WO 2008/109567 A2

(51) International Patent Classification:
G06T 7/20 (2006.01)

(21) International Application Number:
PCT/US2008/055728

(22) International Filing Date: 3 March 2008 (03.03.2008)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/892,738 2 March 2007 (02.03.2007) US

(71) Applicant *(for all designated States except US)*: **ORGANIC MOTION** [—/US]; 336 West 37th Street, Studio 530, New York, NY 10018 (US).

(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: **TSCHESNOK, Andrew** [US/US]; New York, NY (US).

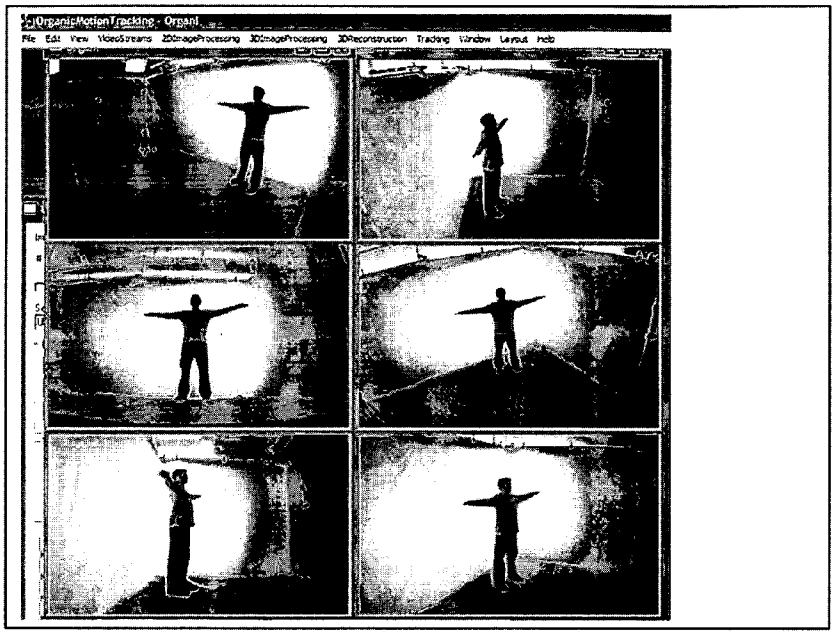
(74) Agents: **DOBROW, James, F.** et al.; Lowenstein Sandler P.C, 65 Livingston Avenue, Roseland, NJ 07068 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL, NO, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

(54) **Title:** SYSTEM AND METHOD FOR TRACKING THREE DIMENSIONAL OBJECTS



(57) **Abstract:** Embodiments of the invention are directed to improved systems and methods for three dimensional (3D) image reconstruction. The systems and methods are directed to extracting, digitally reconstructing and tracking 3D objects from multiple two dimensional (2D) video camera sources. The systems and methods are directed to reconstructing a 3D scene via 2D cameras and then re-projecting this data back onto 2D surfaces. This systems and methods can greatly simplify the image processing required to analyze the 3D model by moving the analysis techniques back into the 2D domain.

WO 2008/109567 A2

SYSTEM AND METHOD FOR TRACKING THREE DIMENSIONAL OBJECTS

CROSS-REFERENCE TO RELATED APPLICATION

[0001] This application claims the benefit of U.S. Provisional Patent Application No. 60/892,738, filed March 2, 2007, the contents of which are hereby incorporated by reference herein.

BACKGROUND OF THE INVENTION

Field of the Invention

[0002] Embodiments of the invention relate to computer vision and three dimensional (3D) image reconstruction.

Description of the Related Art

[0003] Over the past twenty years many well documented techniques have been developed in the field of computer vision to try to reconstruct, digitize and track objects in 3D. One such technique is the reconstruction of objects from multiple two dimensional (2D) outline images taken of this object. This technique typically makes use of a 3D volume of voxels which are carved using the 2D outline images from varying viewpoints. This technique has been effectively used to generate voxel volumes that represent the general 3D shape of the object being reconstructed.

[0004] Several algorithms have been published for the computation of the visual hull, for example: W. Martin and J. K. Aggarwal, "Volumetric descriptions of objects from multiple views," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 5, no. 2, pp. 150-158, March 1983; M. Potmesil, "Generating octree models of 3D objects from their silhouettes in a sequence of images," Computer Vision, Graphics and Image Processing, vol. 40, pp. 1-29, 1987; Richard Szeliski, "Rapid octree construction from image sequences," CVGIP: Image Understanding, vol. 58, no. 1, pp. 23-32, July 1993; and W. Niem, "Robust and Fast Modeling of 3D Natural

Objects from Multiple Views", Proceedings "Image and Video Processing II", Vol. 2182, pp. 388-397 (1994). The contents of these documents are hereby incorporated by reference herein.

[0005] These approaches attempt to solve the problem in a volumetric space representation. The most common of these representations is to subdivide a 3D box into a set of voxels of discrete size. The size of the box is predetermined so that the object can be contained by it. To improve performance these may be represented as "octrees" or are run-length encoded.

[0006] Further related information can be found in: C. H.Chien and J. K. Aggarwal, "Identification of 3D Objects from Multiple Silhouettes Using Quadtrees / Octrees", Computer Vision Graphics And Image Processing 36, pp.256-273 (1986); and A.W. Fitzgibbon, G. Cross and A. Zissermann, "Automatic 3D Model Construction for Turntable Sequences", Proceedings of the European Workshop on 3D Structure from Multiple Images of Large-scale Environments (SMILE '98), LNCS 1506, pp. 155-170 (1998). The contents of these documents are hereby incorporated by reference herein.

[0007] The problem with traditional space carving techniques is one of performance and flexibility. By predefining a grid of voxels in X,Y and Z dimensions computer resources are quickly depleted. Real-time performance has only been possible with greatly reduced resolutions involving as few as 1 million voxels representing a 100x100x100 low resolution grid. Many techniques have been developed to try to optimize this approach using space subdivisions and other optimizations. These techniques have helped but have not made a voxel based approach a real time practicality. US Patent No. 7,327,362 describes one such approach. The contents of this patent is hereby incorporated by reference herein.

[0008] After image reconstruction using the above techniques, researchers have then often used various techniques of analyzing and tracking the objects. Analyzing voxel grids has proven to be very time intensive. One technique of tracking a human figure represented by a voxel grid has been described in US Patent No. 7,257,237. The contents of this patent is hereby incorporated by reference herein.

[0009] Accordingly, there exists a need in the art for improved systems and methods for three dimensional (3D) image reconstruction.

SUMMARY OF THE INVENTION

[0010] Embodiments of the invention address these and other needs by providing improved systems and methods for three dimensional (3D) image reconstruction.

[0011] Embodiments of the invention are directed to new systems and methods of image reconstruction which avoid the use of voxels. These embodiments address both the performance and resolution problems inherent in voxel based reconstruction approaches as well as the performance problems related to analyzing and tracking a voxel shape once it has been reconstructed.

[0012] A new approach ahs been developed for extracting, digitally reconstructing and tracking 3D objects from multiple 2D video camera sources. This method particularly lends itself to high speed detection and tracking of complex organic shapes such as animals as well as the human body. Embodiments are directed to a method that successfully achieves real-time marker-less motion capture system which no longer requires subjects to use markers or attached tracking devices.

[0013] Embodiments are directed to a novel approach to reconstructing a 3D scene via 2D cameras and then re-projecting this data back onto 2D surfaces. This method greatly simplifies the image processing required to analyze the 3D model by moving the analysis techniques back into the 2D domain.

[0014] In addition to providing a novel new approach of digitally reconstruction 3D scenes, embodiments of the invention provide a unique approach of leveraging and re-purposing graphics card hardware (Graphics Processing Units - GPU) developed for the Video Game industry. Repurposing this hardware using this novel approach can accelerate 3D scene reconstruction at multiples of 100 folds over conventional CPU driven approaches.

BRIEF DESCRIPTION OF THE DRAWINGS

[0015] Objects and advantages of the invention will become apparent upon consideration of the following detailed description, taken in conjunction with the

accompanying drawings, in which like reference characters refer to like parts throughout, and in which:

[0016] FIG. 1 depicts 6 views of video from 6 cameras surrounding the subject;

[0017] FIG. 2 depicts video views showing background subtraction information;

[0018] FIG. 3 depicts a video view of cameras looking at large calibration board;

[0019] FIG. 4 depicts a 3D view on right showing a 2D polygon after all camera data has been projected on it (in this case the location of the Polygon is offering a cross section of the floor along the length of the subjects feet);

[0020] FIG. 5 shows the recursive use of 2D projected polygons to "examine" the space and find the full figure of a human subject;

[0021] FIG. 6 depicts initial matching of a skeletal figure to the data generated by evaluating the 2D polygons;

[0022] FIG. 7 shows 2D polygons after texture data has been projected onto them;

[0023] FIG. 8: depicts a close up of an arm being tracked using 2D polygons in local coordinate space;

[0024] FIG. 9 shows how a texture can be projected onto a 2D polygon;

[0025] FIG. 10 depicts another viewpoint of the projected texture from FIG. 9;

[0026] FIG. 11 shows a layout depicting how the real world relates to a digital domain and how cameras relate to virtual projectors;

[0027] FIG. 12 shows how camera/space calibration is setup;

[0028] FIG. 13 shows a virtual projection cloud of intersecting camera projections;

[0029] FIG. 14 shows how data from camera projections are "Drawn" onto a 2D Polygon;

[0030] FIG. 15 shows how slices positioned local to the bones of a character being tracked can offer position information for the next time-step;

[0031] FIG. 16 shows how 2D slice analysis can be used to eliminate confusion in complex scenes;

[0032] FIG. 17 shows how "end caps" can help properly position a skeletal figure being tracked;

[0033] FIG. 18 depicts the flow of data in an exemplary system;

[0034] FIG. 19 shows a typical video camera layout, in accordance with embodiments of the invention; and

[0035] FIG. 20 depicts a close up view of projections onto a 2D polygon.

[0036] It is to be understood that the above-mentioned drawing figures are provided solely to assist in describing the concepts of the present invention and may not be to scale, and are certainly not intended to be limiting in terms of the range of possible shapes and proportions well within the scope of embodiments of the invention.

DETAILED DESCRIPTION

[0037] The steps used, in accordance with embodiments of the invention, to extract and track a human subject in 3D who is being watched by multiple video cameras, are described below:

[0038] Step 1. Image acquisition.

[0039] Certain embodiments use the video feeds of 10 or more digital video cameras which are oriented around the area of interest in a way to give the maximum number of perspectives.

[0040] The Video acquisition subsystem is described on FIG. 18. Multiple cameras [1801] feed into a host computer either via Firewire Cable [1802], Ethernet IP, via Analog cable or any other common used method to transfer video. This video feeds into camera acquisition boards [1803]. In the case of our demonstration we use custom made cards that function using the IEEE1394a firewire standard. In general any type of image interface board can be used as long as it has the capability to transmit video data via Direct Memory Access (DMA) Transfer [1804]. Using DMA we then transfer the video data to system memory on the host computer. From there the data is moved via

DMA transfer directly to the Graphics Processing Unit Memory (GPU Memory) [1805]. The GPU in our case is a standard DirectX capable consumer grade 3D graphics card capable of running vertex and pixel shaders. Optionally we simultaneously make a copy of the video data and DMA transfer it to a Hard Disk Array [1806] for later retrieval. The purpose of using 100% DMA transfers is to free up the CPU of the host computer for other tasks. This method of image transfer allows us to move all video data into place using only about 1% of CPU cycles.

[0041] FIG. 11 shows a diagram of how to position cameras around a subject or scan area. FIG. 1 shows a snapshot of our software showing the 6 of the typical video feeds that feed into the host computer when a subjects stands in a setup described in FIG. 11.

[0042] Step 2. Background Subtraction.

[0043] i) Using black and white cameras: When we use black and white cameras we use an active background to aid us in rapidly subtracting the subject from the background. In our case the active background consists of a backlit sheet or a reflective surface as can be achieved with reflective products sold by 3M called ScotchLite. This material is commonly found on street signs, reflective stripes on clothing or even in paint on streets. If shining a light on this material it reflects the light back to the location the light came from. When placing lights around the camera lens the effect of the light returning to the camera makes the background appear to glow. The basic idea behind these glowing backgrounds is that even the lightest texture of image of the subject will be darker than the background. Two methods can then be used to subtract the background. We can use either of these techniques interchangeable depending on the lighting conditions of the setup. In one case we simply clip the image at a specific brightness level. In this case we say that every pixel that is brighter than a specific threshold belongs to the background. In the FIG. 2 this background is replaced with black to help illustrate. Another way to replace an active background for black and white cameras is to take a reference shot of the background and then do a pixel per pixel subtraction of the background and foreground image. We then take the absolute value of the subtracted image and clip on a specific threshold. The idea here is that pixels that differ more than a specific amount between foreground and background are probably part of the foreground. The resulting subtracting looks similar to clipping the

background at a high intensity value (as in option one) By using active backgrounds we eliminate complexities in image segmentation that are introduced by changes in ambient lighting or shadows as the subject moves in the scene. The backlighting eliminates or washes out shadows. The disadvantage to this approach is that we need to use a dedicated studio setup with active or special reflective surfaces placed around the subject.

[0044] ii) Using color cameras: When we use color cameras we have more sophisticated options to subtract background. In the case of color cameras we take a snapshot of the background (i.e. without the subject in the scan area) and convert this image from RGB into HLS color-space (Hue, Luminosity, Saturation) We do the same for the foreground and apply the following rules:

[0045] a. If the foreground pixel is lighter in Saturation than the background then it is a foreground pixel.

[0046] b. If the foreground pixel is darker than the background pixel it may be a foreground pixel or it could be a background pixel that has a shadow on it. Shadows tend to have the same hue value as the color they are affecting. In this case we do a second check to see if the hue value of the foreground is similar (within a certain range) to the background. If they are and the difference in saturation and luminance is not too great then we can assume that this is a background pixel which is being effected by a shadow of the subject.

[0047] c. If the foreground differs from the background in hue value even if saturation and luminance are the same then we clip and make the pixel a foreground.

[0048] d. If foreground and background differ in saturation beyond a specific range then we also clip and call the background the foreground.

[0049] By combining all these rules we have developed a strong per-pixel background subtraction which holds up well to lighting changes in a scene. It allows us to do solid background subtraction in both indoor and outdoor settings without the need for a dedicated studio setting. In addition to the per pixel subtraction technique mentioned we also may use region based techniques that look at changes in gradients in background verses foreground. In some cases a pixel in the foreground may have been

falsely set to background because it has similar color values to the background. But since adjacent pixels differ in gradient (such as going from light to dark in opposite direction between foreground and background) we can make the assumption that the pixel is in fact not a background pixel.

[0050] Combining these techniques allows us to build solid background subtraction in complicated lighting conditions such as daylight. The downside is that there is a significant performance hit to the system with all these additional calculations when compared with the black and white technique described above. Both techniques can run in real-time however by the projection method we describe later in this document.

[0051] A special note on Background Subtraction:

[0052] We describe background subtraction in concept here. In our approach we don't subtract backgrounds at the 2D level or in a pre-process step. We project backgrounds into the scene in the same way we project the video (as described below). This gives us complete freedom to work with the data after it has been projected onto the 2D Polygon (or slice) (as described below).

[0053] Step 3. Lens distortion and camera calibration:

[0054] The lenses on video cameras distort images. This is seen most clearly when looking through a fisheye or wide angle lens. Straight lines have a tendency to curve around the edges. To successfully reconstruct objects that are seen by the cameras we need to un-distort these images and compensate for the image shifts introduced by the optical properties of the lens. Only if we can compensate for these distortions can projected lines converge properly in the digital domain as we reconstruct the real scene. To do this successfully we must first derive the exact lens parameters. The parameters include the following: Exact, or near exact, focal point in both X and Y directions (this is the number on the lens such as 6mm focal point but we need to figure out the real number which may be around 6.0231 mm for example. Exact lens center-point in both X and Y (this is the measure of how the lens is attached to the image sensor. Manufacturers place the lens in the center of the image sensor but we need to know this number down to less than one pixel or 0.002mm). Lens Skew in both X and Y. This is the value for which the camera lens is not mounted straight on the camera. We use two

more values to accommodate for how the focal length changes at different points from center of the lens to the edges.

[0055] Step one: We show the lens a checkerboard pattern and take about 30 images from different angles. The system auto-detects these checkerboards and figures out the intersection point of all squares on the board. This leaves the systems with 30 sets of point. From these points it uses well published algorithms to calculate all lens parameters.

[0056] Step two. Once we have figured out lens parameters we need to calibrate the location of the cameras. Here we take a larger checkerboard and place it in the center of the scene. Every camera that can view the checkerboard takes a picture and detects the checkers and the points. Although we use published methods to derive camera positions relative to the board, we have devised a technique that allows us to improve on accuracy here. We continuously take pictures by all cameras (100 or more) and then build a map of positional relationships between all cameras based on each cameras relative position to the board. (See FIG. 12).

[0057] We then place the board on the floor and have one camera get the position relative to the board on the floor. The top left checker on the checkerboard on the floor becomes the X and Y origin point of the scan space and the up vector (coming out of the board) becomes the Z vector. Since every camera knows the relative position between it and every other camera the camera that can best see the board will orient the grid of cameras relative to the origin point in both position and orientation. Now every camera's position is known relative to the boards origin point. The accuracy of the camera grid to the origin point is not as important and we do this using just one sample. But the relative position of cameras to each other use over one hundred samples. We further improve this by taking the standard deviation and removing outliers from the calculation. We have found that we achieve a sub-millimeter XYZ positional accuracy for every camera relative to the origin point. FIG. 3 shows a view from various cameras of the larger checkerboard.

[0058] Step 4. Image reconstruction via projections onto 2D surfaces.

[0059] One we have subtracted backgrounds and have gained the knowledge of every cameras exact position and lens properties we are ready to implement one of our

breakthrough techniques: We can project these images onto 2D surfaces. To project the video onto surfaces we employ a technique commonly used in 3D graphics to project shadows in 3D scenes. There are many well published techniques used to project shadows, or light sources or even video onto 2D polygon textures using DirectX or OpenGL. These techniques involved distorting the UV texture coordinates of the 2D slice being rendered to read the texture being projected. In our case the texture being projected is the 2D video stream. These techniques are well understood in the filed of 3D computer graphics. The following publication gives a good background to the art:

[0060] Mark Segal and Carl Korobkin and Rolf van Widenfelt and Jim Foran and Paul Haeberli, "Fast shadows and lighting effects using texture mapping," SIGGRAPH Comput. Graph., vol. 26, no. 2, 1992, issn. 0097-8930}, pp. 249-252. The contents of this document is hereby incorporated by reference herein..

[0061] By way of analogy, the computer is basically simulating video projectors. In a 3D simulation we turn every camera into a video projector. We project the video feed of the background subtracted image, or the image together with the background image into the 3D space for each of the cameras. The idea here is that every camera projects a video feed into the exact direction and from the exact position of the actual camera with the exact lens properties of the actual camera. In effect we are reversing what is happening in the real world. In the real world light from all objects is emanating from the subject into the many cameras of our system. In the 3D model in the digital domain these images are doing the reverse – the system is projecting the light from the cameras into the simulated room with the same exact lens properties. When all cameras are properly calibrated then these rays of light meet inside the 3D digital domain in the same way they had emanated from the subject in the real world. (See FIG. 13).

[0062] FIG. 19 offers a side by side comparison between the real world and the digital domain. The "data cloud" represents a theoretical space in the digital domain. All data is available and ready to represent the real world, but only by placing a polygon into this space do we actually project the video data onto a surface. By placing a 2D polygon into this projection space we can catch these simulated rays. We do this in the following way: if a foreground texture (i.e. non-black) touches a pixel on the surface of the slice we increment a counter by one or by some amount by which it differs from the background. If at the end of projecting all cameras onto this slice we

count the number in increments. If they equal the number of cameras we have projected (or some slightly lesser number) then we know that there is actually a solid object in the space of this pixel. This solid pixel then represents a solid object in the real world at that exact location. We use many 2D polygon slices to deconstruct a 3D scene. It looks similar to an MRI imaging system when we place many slices from top to bottom on the scene. By getting all the cross sections this way we can visually see. the person standing in the real word reappear on the slices in the digital domain. The more slices we use the more precise the image becomes.

[0063] FIG. 14 illustrates how two points in the video data may line up on a polygon. FIG. 20 shows this same intersection in a different way by taking a closer look of slice number 3 from FIG. 19. In this case one can see how a ray that is part of. the right arm projects out from every projector to meet at the right spot on the slice. Also note the doted slices on the 2D pictures [2001]. They give an idea of what the slice would look like if it was projected back into the cameras.

[0064] In this way the polygon (i.e., 2D slice) effectively slices through space and offers a "view" into the real world that the cameras are filming. We can now generate a. slice cross section of the real world by placing such a polygon into the virtual space. An analogy to the slice may be data from an MRI where you can get a slice through the cross section of the human body (note: MRI uses very different techniques) We get this same cross section but of course without being able to see inside the body. We simply get the outside contours of the slice along with the outside texture (Actual video image of the surface area of the cross-section)

[0065] Why is this so special? It differs greatly from other published techniques of volumetric reconstruction using multiple cameras in the following important ways:

[0066] 1. It is resolution independent. The clarity of the slice is determined by the resolution of the surface of the slice texture we are rendering. This means we can chose. the properties of the slice dynamically. We can chose where and at what orientation the slice is placed. We can chose what "real world size" it represents (i.e. is it one meter across or 5 cm). We can chose the resolution (i.e. is this 2D slice made up of 10x10 pixels or is it 1000x1000). This dynamic ability to "look" and "examine" the real world inside our 3D virtual projection system (the digital domain) allows us to do something

very important: We can now decide on the fly what we need to look at carefully and what parts of objects are not as important. In the case of tracking human motion for instance we can now look at the chest at a less fine resolution then we may look at the fingers of the subject. By avoiding a voxel based approach to reconstruction we completely circumvent the lookup table and resolution dependant models that are part of most research in this field.

[0067] 2. We can repurpose 3D video hardware (Graphical Processing Units – GPU) designed for the latest retail video gaming market. This technique allows us to directly use hard wired features developed as part of billion dollar research in 3D graphics and video gaming and repurpose them for our needs. Projecting textures is a big part of making lighting in video games more realistic. Using our slice technique we piggyback on this ability and use it in a completely new way. This way we achieve hardware accelerated projections speeding up rendering time to 100 or even 1000s of times faster than CPU based approaches.

[0068] 3. It is very fast. The techniques of motion capture described below run up to one million times faster then many published techniques in this domain.

[0069] 4. The dynamic resolution technique allows us to achieve a much higher resolution than is possible with a voxel based approach. We typically run at 200 times the resolution of published systems running on voxel models.

[0070] 5. In addition to general 3D model reconstruction we are able to use this slice approach to solve many other problems in computer vision that have been previously solved in the 2D domain but not in the 3rd dimension. Using slices we can for instance effectively track textures and cross section (described later) in the 2D domain and then extrapolate their 3D position in a second step. We therefore effectively get 3D image processing at the cost of 2D image processing techniques. We get to repurpose many 2D image techniques that never worked effectively in the 3D environment.

[0071] 6. Scalability. Building a vision architecture based on slices is very scalable. A system could be setup using many servers and hundreds of cameras and every individual slice could be rendered by picking the right cameras that are relevant

to it based on its position in the space. This would allow for the ability to outfit a large facility with cameras and track objects over vast amounts of spaces.

[0072] Step 5. 2D Image Analysis on Slices

[0073] As mentioned above, one of the great benefits of the projected slice technique is the ability to work with conventional image processing techniques and then project them into the 3D domain. The first step we use is to sort objects that we find on the 2D slice. An object is defined as a continuous cross-section on the 2D plane. We call them blobs. Once we have isolated individual blobs we apply various published 2D curve fitting algorithms. A common technique we use is to "fit" an ellipse to the blob. This gives us an X and Y position value of the center point of the blob, as well as an X and Y radius and a rotation angle. In this way we have moved from raw 3D data to raw 2D data using our slice system and then we moved from raw 2D data to usable vector data by simply fitting a 2D ellipse to our raw blob points.

[0074] In the next step we then take the 2D vector data and derive its 3D properties by applying the properties of the slice to it. We take into account the position and orientation of the slice, the resolution of the slice as well as its real world size. From this we can convert the pixel XY positions of the ellipse to XY coordinates in real world space (i.e. in millimeters relative to the origin point of the floor calibration board)

[0075] In this way we have generated 3D vector data of a real world object in just a few simple steps.

[0076] FIG. 4 shows our system with a single slice placed at the feet of the subject at floor level. Through the projection techniques described above we found the "foot prints" of the subject. We then applied the above mentioned ellipse fitting technique. The red crosses mark the center points of the objects on the slices. The two 2 dimensional windows of the slice on the screen show the actual textures of the slice that is in the 3D view.

[0077] Step 6. Model Detection.

[0078] This is how we detect a human shape that walks into the space:

[0079] a. We place a Slice on the floor and look for footsteps. We do this by analyzing the slice and by looking for "blobs" with the properties that may be the size of feet. If we find them we:

[0080] b. Place another slice a few inches higher. These are the ankles. If we find them we determine what direction the person if facing. The tips of the feet tend to face away from the ankles.

[0081] c. Next we place a slice at knee, hip, chest and head height. We place a few slices around the head to try to pinpoint the height of the person.

[0082] d. We find the orientation of the body by looking at the angle of the torso.

[0083] e. We look for the arms by placing slices on either side of the body.

[0084] When all these test check out we can determine with a high degree of certainty that we are looking at the human shape in a specific pose. A graphic sample of this can be seen in FIG. 5.

[0085] Step 7. Model Matching:

[0086] Once we used the above "smart sampling" technique and have determined with a rough estimate that we are looking at a human subject we move into the next phase. We employ a rag doll system based on physics, joints, mass and inverse kinematics. We apply forces to this Rag Doll and move it into place where we think the arms, hands, head and feet are. We initially assume that the "bones" of the rag doll fit into the center sections of the ellipses of the slices we have placed into the space. With step 7 we have roughly matched a model to the 3D data of the real world. FIG. 6 shows our system superimposing a rag doll into the 2D slice framework to illustrate this technique.

[0087] Step 8. Model Tracking:

[0088] In step 8 we hand off the mechanism of placing slices over to the Rag Doll. From the point of view of the Rag Doll (and no longer from the global reference point) we place slices for every body part. The slices are "attached" to the bones in fixed orientations. We calculate all center points and figure out the discrepancy between the

ellipse centers and the bone center. We then apply forces to the model to allow the model to further approximate the position of the 3D data. FIG. 15. In this way we can animate the Rag Doll and make if follow the real world human where the Human becomes the puppet master and the Rag Doll a mime which follows the masters projected data in the 3D computer simulation. Many times per second we repeat this process: The subject moves, the cameras record and update the projections they project into the 3D simulation. Each Rag Doll bone places slices around its bones and compares its position to the position of the ellipses that are generated. Forces are then applied to the Rag Doll and each Rag Doll bone is moved incrementally closer to the center of the cross section of the slice that is representing the real subject.

[0089] Things we watch out for when tracking bones this way:

[0090] Interference between objects: Example: when tracking an arm there is a problem when this arm moves close to the body. At some point two separate blobs, one belonging to the cross section of the arm and one to the cross section of the body are so close that they form a single blob. It then becomes hard to determine which object belongs where. We solve this problem by using the Rag Doll and checking it for such situations. Since the Rag Doll is moving over time in the same way the real person is moving, we can make the assumption that the Rag Doll closely resembles the real data. In the case the two bones are close enough to be "touching" then we employ a mechanism that calculates the highest likelihood of a small blob that has merged with a large one as may become the case with an arm that is too close the body. FIG. 16 for details. Since we know the diameter of the arm and the diameter and relative position of the body we can make an assessment that the small blob of the arm resides within the larger blob of the body. This technique has proven very effective at keeping body parts from sticking together and creating a breakdown of the tracking system.

[0091] End Caps. We call slices that search for the end of body parts end caps. End of body parts include the hands, feet and head. End caps make sure that we are stretching the Rag Doll into the extremities. By forcing the hands, feet and head into position we have achieved great model matching. FIG. 17. We apply the same model aware technique to end-caps as in the previous section. If one hand moves to close to another hand we adjust our forces to prevent one hand from "merging" into the other.

[0092] Step 9. Using textures:

[0093] As mentioned before, one of the great advantage of projecting onto slices is the ability to use 2D image processing techniques on 3D models.

[0094] Once we have matched and are tracking our Rag Doll using the cross sections of our slices we can run a second pass to refine the position of our model. By analyzing the surface textures of the model we can further position the Rag Doll to match the actor in the real world.

[0095] Here is how it works:

[0096] When the Rag Doll is in very close proximity and position of the actual person we can place "slices" not as cross sections but as surface sections. For example we can place a slice on the chest of the subject. The outline of this slice is determined by the same method the cross sections, showing us the outline of the chest it is intersecting. But since we know the position of this slice relative to all cameras we can determine and render the actual projected texture from a blend of the best positioned camera projectors. Now the chest has not only the outline of the body but also the "look" and texture of the body from the video. We can place these slices all around the body and give the model the same "look" of the real subject. More importantly we can now analyze each 2D slice and track the movement of the textures themselves to generate a high degree of accuracy.

[0097] 1st pass – fit the Rag Doll to the contours of the slice (i.e. ellipse positioning)

[0098] 2nd pass – once the Rag Doll is moved into very close proximity of the position of the real person we can place slices on the surface of the subject. These will offer us the 3D textures of the real person. We can then track these textures to further improve the positioning of the Rag Doll. See FIG. 7 and FIG. 8 for examples of projection textures onto slices.

[0099] Step 10. Texture Tracking.

[00100] We employ published techniques of texture tracking on 2D surfaces. For each slice we compare the "image" or texture to a reference image that we took from

the same slice at the start of the animation sequence. We then determine by what X and Y value the two textures differ in position and by what angle they have rotated. The unique part of this process is that we can then use this 2D X and Y and angle data and calculate its 3D position in much the way we did with the ellipses. We take the position and orientation of the slice itself and extrapolate how the change in 3D is actually a change in 3D. We use this offset to add forces to the Rag Doll to further refine its position in a second pass after its initial fitting with the ellipses. It is in this second pass that we achieve a very high degree of accuracy that exceeds that of even marker based motion capture systems which employ markers and body suits today.

[00101] FIG. 9 and FIG. 10 show how we can place a slice onto the chest of the model and project a texture from the video image onto the model. The projected texture is shown in the 2D window in the lower left corner of the images.

[00102] In summary:

[00103] By projecting onto 2D slices (2D polygons) we have achieved a major breakthrough in computer vision. It allows us to approximate organic objects (in our chase the human subject) by applying and repurposing many published techniques from the 3D world and make them work efficiently in the 3D domain. It has allowed us to build the world first real time marker-less motion capture system which can perform at the speed and resolutions of systems that require the subject to wear specialized markers and body suits to aid the vision computer in their tracking task.

[00104] Notes on GPU hardware:

[00105] Embodiments make use of the Redeon 9700 Graphics Processor. The Redeon processor is the first GPU capable of handling hardware based projection algorithms described in this paper. The first slice system was implemented using shader model 1.4, the most advanced shader system at the time.

[00106] With the introduction of GPU shader version 3.0 we have been able to implement the entire projection technique into a single pass. This means we calculate everything from lens distortion to background subtraction and projection all in a single pass and via instancing we are able to project all slices in a system with a single GPU call.

[00107] Embodiments of the invention can include a new method and system of reconstructing real world objects within the digital domain of the computer in 3d by projecting multiple camera video streams within the digital domain onto 2D polygonal surfaces, comprising of 2 or more video cameras, a video camera acquisition or input port, a host computer with a DMA controller, Central Processing Unit and or a Graphics Processing Unit.

[00108] Embodiments of the system of can use a DirectX or OpenGL accelerated graphics card to accelerate the computations of the projections onto polygon surfaces.. Embodiments of the system can use IEEE1394 standards to transfer image data. Embodiments of the system can use Ethernet to transfer image data.

[00109] Embodiments of the system can use Wireless networking protocols to transfer image data. Embodiments of the system can project both video data and background data for each camera onto polygons. Embodiments of the system can convert texture pixel on the surfaces of the 2D projected polygon back to 3d world coordinates.

[00110] Embodiments of the system can be used for tracking human subjects. Embodiments of the system can be used to track animals. Embodiments of the system can be use to animate a 3d character Embodiments of the system can be used as a visual aid in a karaoke system

[00111] Embodiments of the system can use 2D image processing techniques on data projected from the cameras onto the 2D polygons. Embodiments of the system can use 2D shape fitting techniques to evaluate the 2D shapes generated on the projected polygons. Embodiments of the system can perform all background subtraction after all data has been projected onto the 2D polygon.

[00112] Embodiments of the system can use black and white cameras and 3M ScotchLite retro-reflective material to accelerate and simplify background subtraction stage. Embodiments of the system can use color cameras. Embodiments of the system can use vertex and fragment shaders to move slice projection calculation from the CPU to the GPU.

[00113] Embodiments of the system can recursively use projected and analyzed polygons to figure out where to place the next polygon to increasingly investigate the shape of a reconstructed object. Embodiments of the system can project video textures onto the 2D polygon based on a blend between cameras based on their inclination angle relative to the polygon normal. Embodiments of the system can use 2D optical flow techniques to track contrast points in the texture which has been projected onto the 2D polygon.

[00114] Embodiments of the system can be used as part of a motion capture system to track and digitize the movement of humans without the use of markers or attached tracking devices. Embodiments of the system can be used to measure the size of a human subject and to calculate precise joint rotation points based on analyzing the subjects movement. Embodiments of the system can be used to analyze a tennis player: analyzing the swing and body movement of a player to improve performance and to prevent injuries like shoulder, elbow, and ankle problems.

[00115] Embodiments of the system can be used in research and development to studying the movements of subjects while using particular products. Embodiments of the system can be used to analyze the movements of a skier/snowboarder to improve performance and to prevent injuries. Embodiments of the system can be used to animate and broadcast a virtual character of the actual subject over TV broadcast networks.

[00116] Embodiments of the system can be used as part of human computer user interfaces. Embodiments of the system can be used as part of a security system. Embodiments of the system can be used to fit human subjects to sporting goods equipment such as bikes, golf clubs, clothing and or shoes.

[00117] Embodiments of the system can be used as part of a visual simulation or training system. Embodiments of the system can be used to evaluate human body shape, size or vital signs such as breathing mechanics. Embodiments of the system can be used as part of a public entertainment experience. Embodiments of the system can be used as a fitness or exercise evaluation or recommendation system

[00118] Embodiments of the system can be used as part of a physical rehabilitation system. Embodiments of the system can be used as part of a clinical motion analysis or

evaluation tool. Embodiments of the system can be used as an aid for the handicapped. Embodiments of the system can be used as part of a training tool.

[00119] Embodiments of the system can be used as part of a Virtual Reality visualization system. In certain embodiments of the invention, all of the steps of the method can be performed by a computer, or computerized system, as described above. In alternative embodiments, one or more of the steps can be performed manually, by a person.

[00120] In alternate embodiments of the methods described herein, additional steps may be added, certain steps may be excluded, certain steps may be performed multiple times, and/or the steps may be performed in a different order and/or simultaneously.

[00121] One skilled in the art will appreciate that the invention can be practiced by other than the described embodiments, which are presented for purposes of illustration and not by way of limitation. Various arrangements of the described embodiments can be made by those skilled in the art without departing from the spirit and scope of the present invention, which is limited only by the claims that follow.

What is claimed is:

1. A method of tracking a three dimensional object comprising:

acquiring an image including the object from a plurality of cameras;

subtracting a background of the image;

calibrating at least one of the cameras:

reconstructing the image via projections onto two dimensional surfaces;

analyzing the image;

detecting one or more models;

matching the one or more models;

tracking the one or more models;

analyzing the surface textures of the one or more models; and

tracking the surface textures.

2. A system for tracking a three dimensional object comprising:

a plurality of video cameras for capturing an image including the object;

a video camera acquisition module coupled to the plurality of video cameras; and

a host computer including at least a DMA controller and a graphics processing module; the system being configured to perform the steps of:

acquiring the image including the object from a plurality of cameras;

subtracting a background of the image;

calibrating at least one of the cameras:

reconstructing the image via projections onto two dimensional surfaces;

analyzing the image;

detecting one or more models;

matching the one or more models;

tracking the one or more models;

analyzing the surface textures of the one or more models; and

tracking the surface textures.

Fig #1

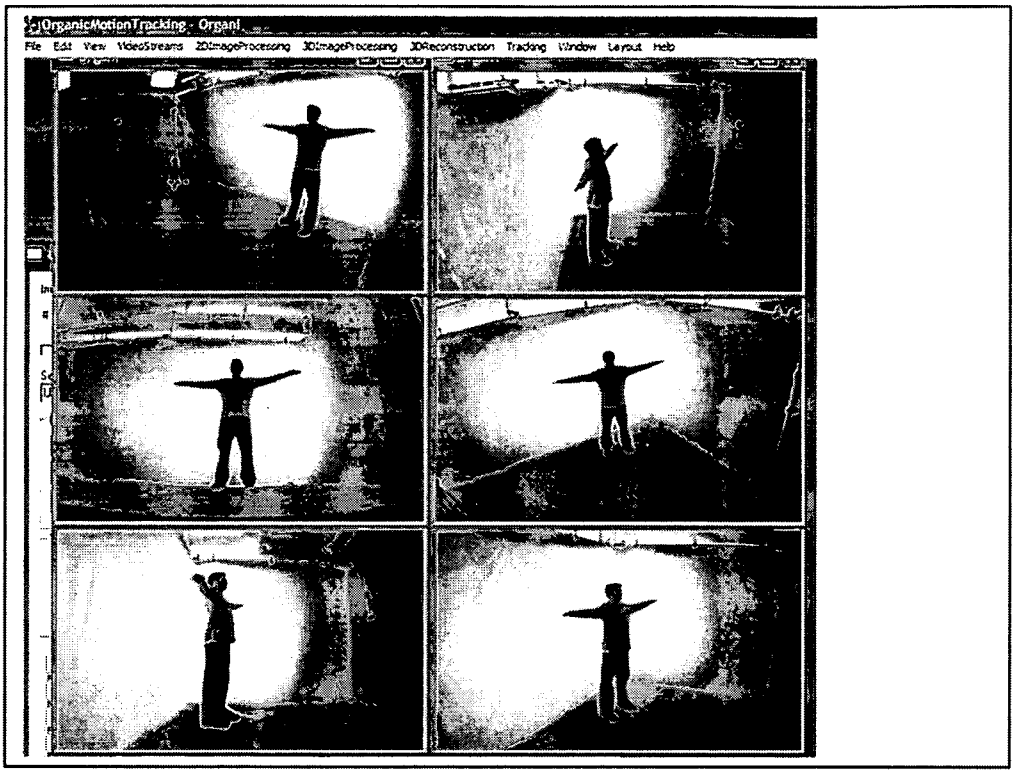

OrganicMotionTracking - Organ
File Edit View VideoStreams 2DImageProcessing 3DImageProcessing 3DReconstruction Tracking Window Layout Help

Fig #2

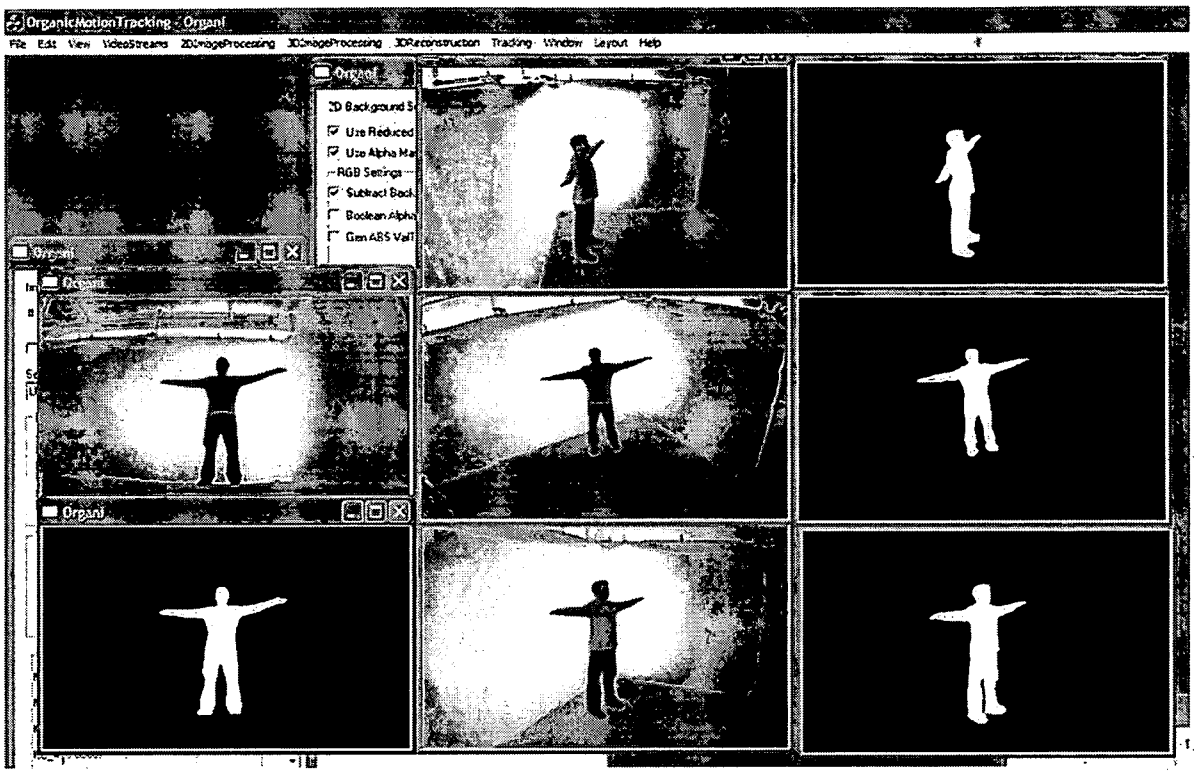

OrganicMotionTracking - Organi
File Edit View VideoStreams 2DImageProcessing 3DImageProcessing 3DReconstruction Tracking Window Layout Help
Organi
3D Background S
Use Reduced
Use Alpha Ma
RGB Settings
Subtract Back
Boolean Alpha
Gen ABS Val
Organi
Organi
Organi

Fig #3

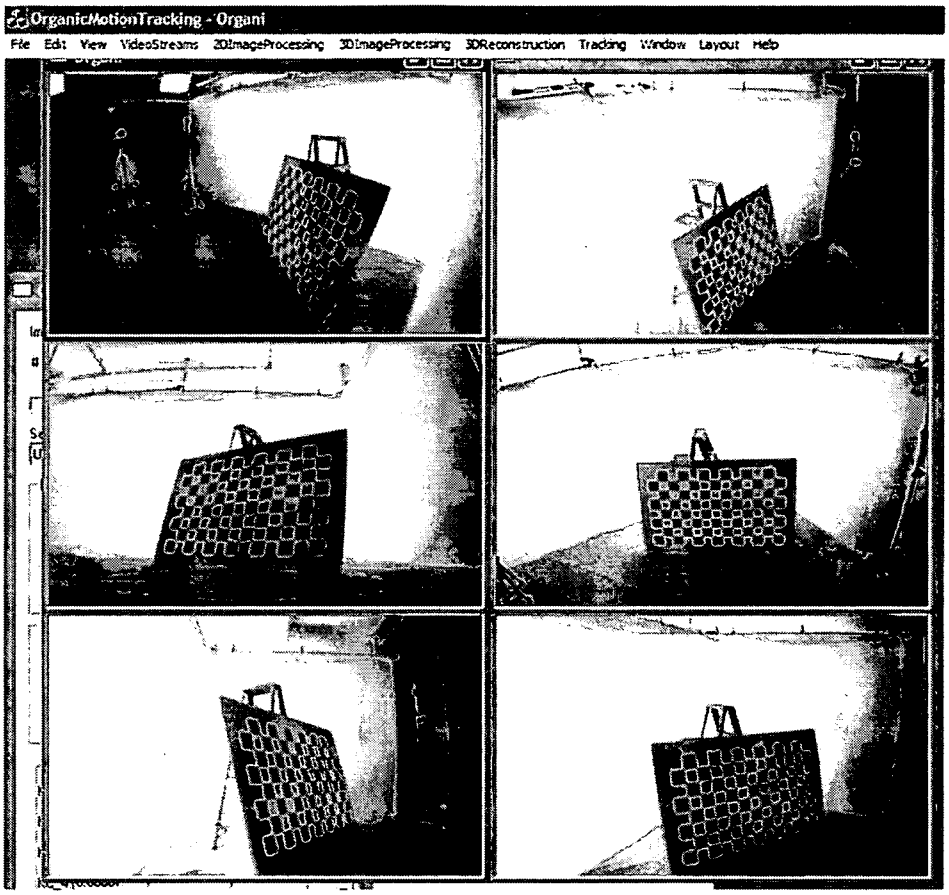

OrganicMotionTracking - Organi
File Edit View VideoStreams 2DImageProcessing 3DImageProcessing 3DReconstruction Tracking Window Layout Help

Fig #4

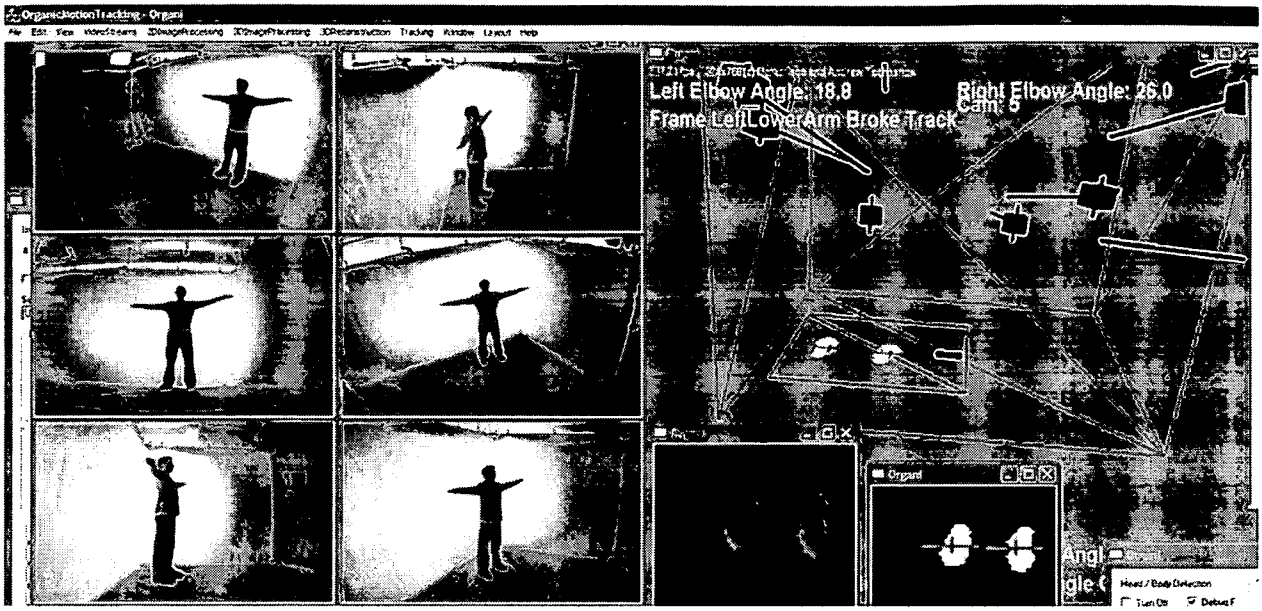

Left Elbow Angle: 18.8
Right Elbow Angle: 26.0
Cam: 5
Frame LeftLowerArm Broke Track
Organ1

Fig #5

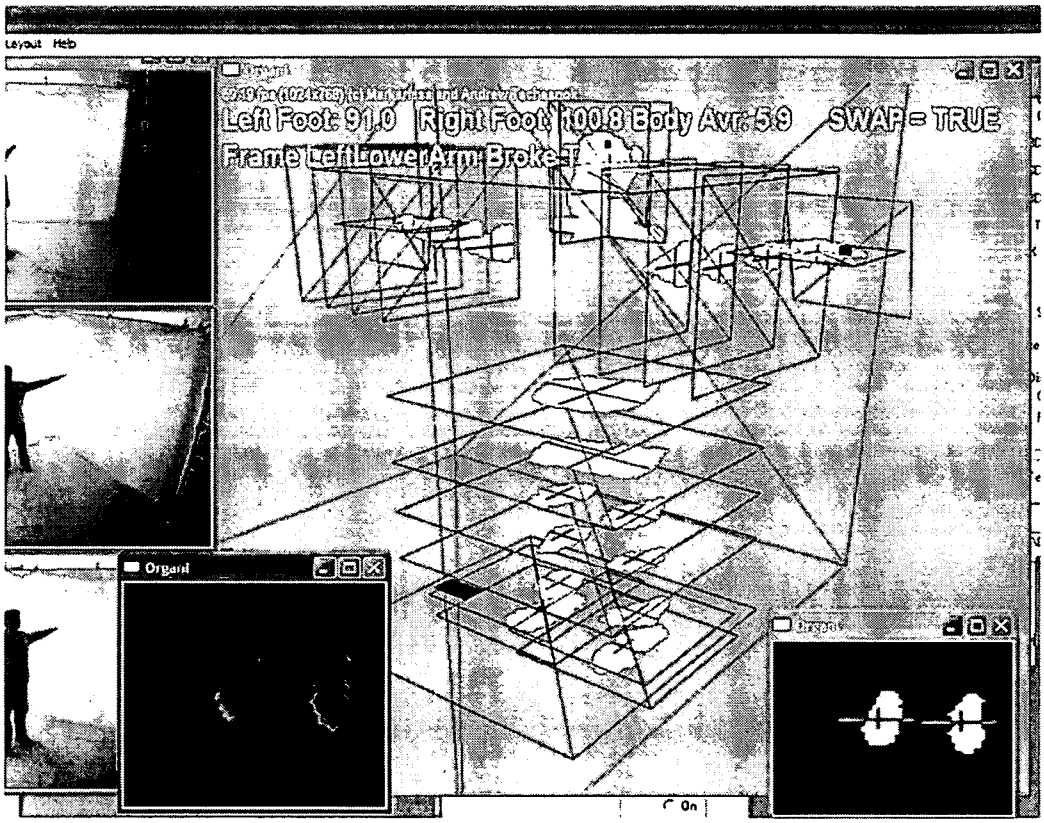

Layout Help
Left Foot: 91.0
Right Foot
00.8 Body Avr: 59
SWAP = TRUE
Frame Left Lower Arm Broke
Organ
On

Fig #6

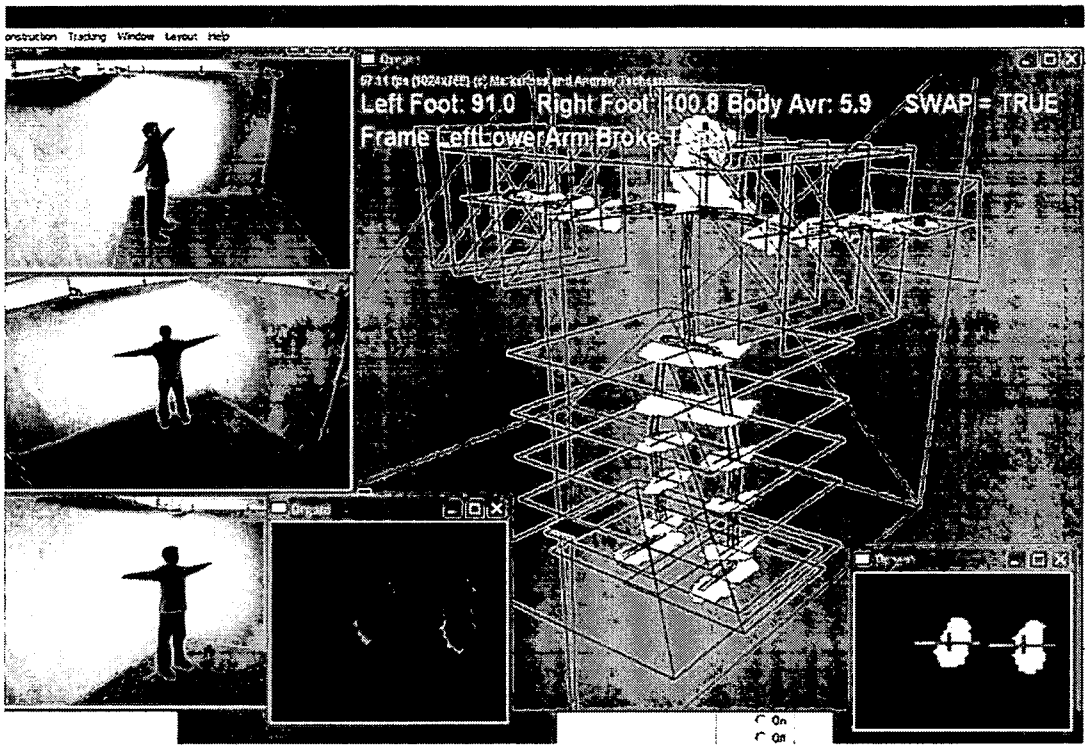


Fig #7

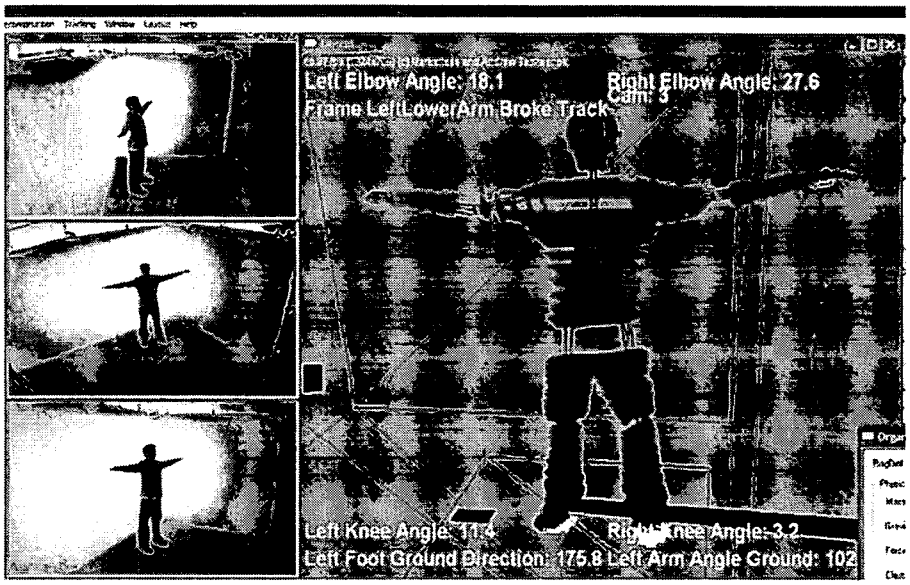

Left Elbow Angle: 18.1
Right Elbow Angle: 27.6
Cam: 3
Frame Left LowerArm Broke Track
Left Knee Angle: 11.4
Right Knee Angle: 3.2
Left Foot Ground Direction: 175.8 Left Arm Angle Ground: 102

Fig #8

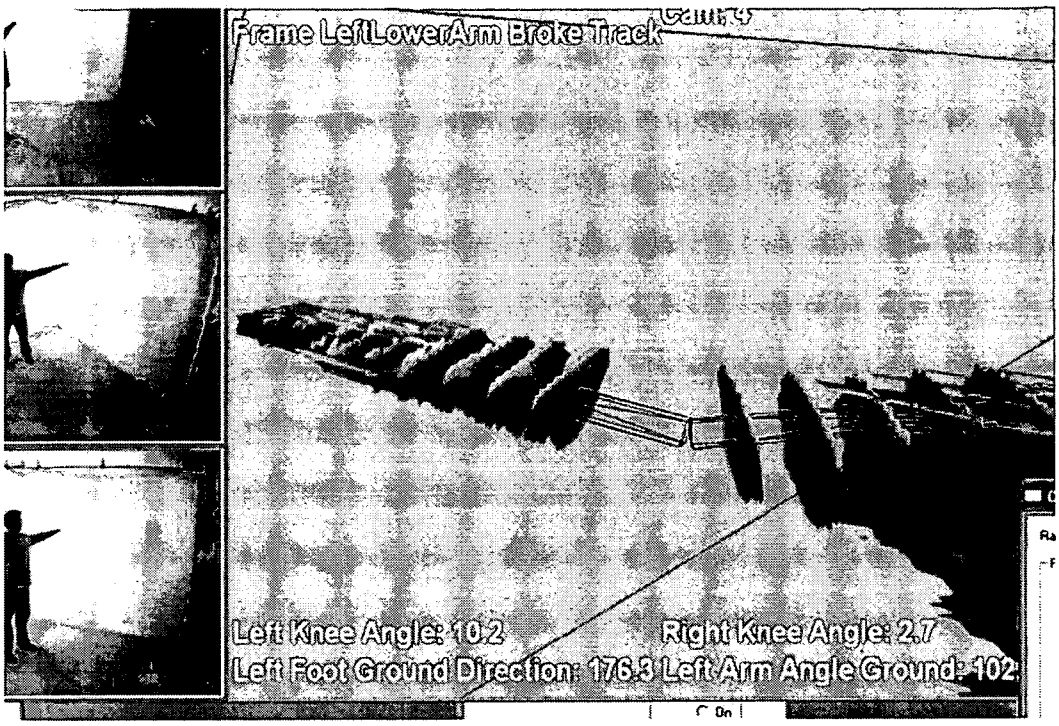

Frame LeftLowerArm Broke Track
Left Knee Angle: 10.2
Right Knee Angle: 2.7
Left Foot Ground Direction: 176.3 Left Arm Angle Ground: 102

Fig #9

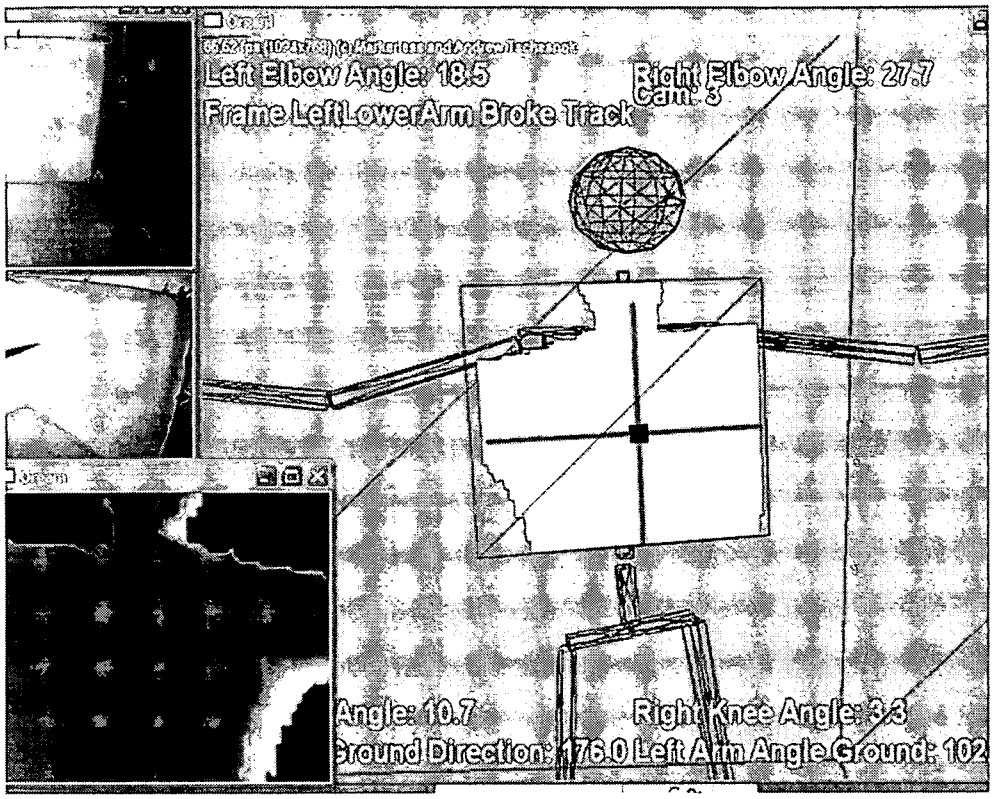

Left Elbow Angle: 18.5
Right Elbow Angle: 27.7
Cam: 3
Frame LeftLowerArm Broke Track
Angle: 10.7
Right Knee Angle: 3.3
Ground Direction: 176.0 Left Arm Angle Ground: 102

Fig 10

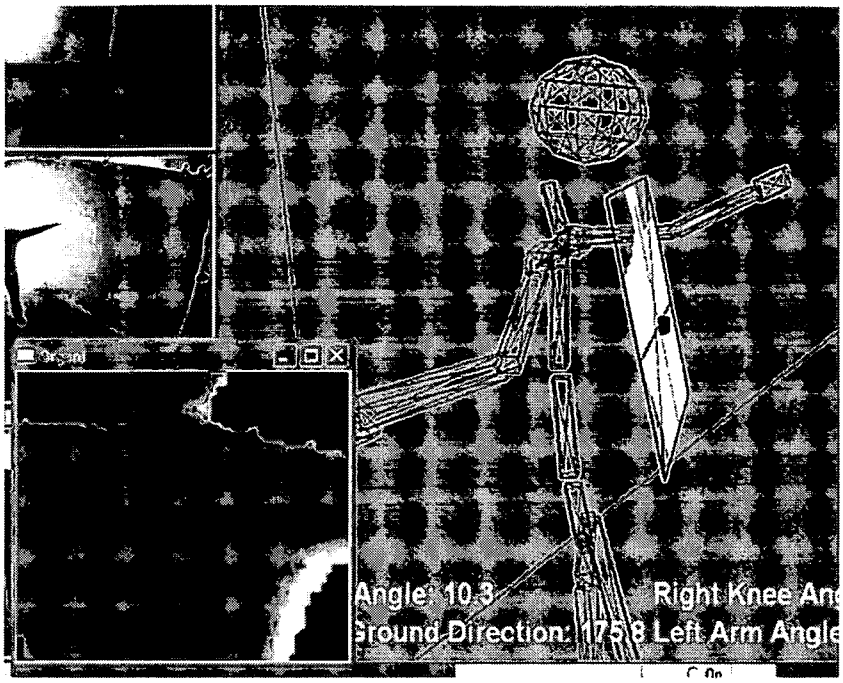


FIG. 11

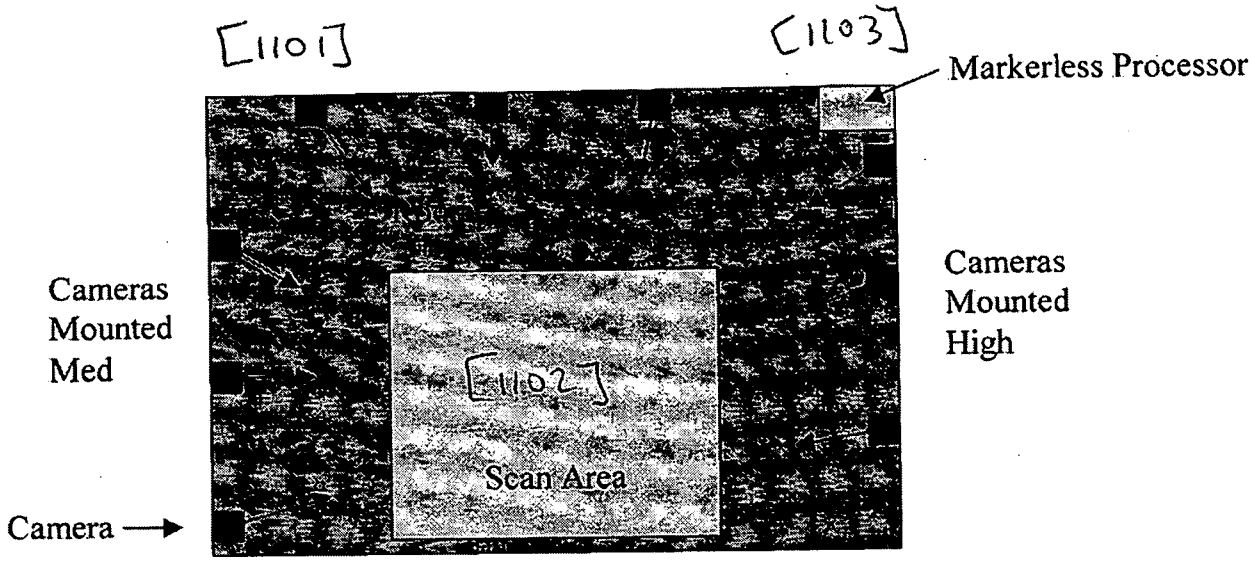

[1101]
[1103]
Markerless Processor
Cameras Mounted Med
Cameras Mounted High
[1102]
Scan Area
Camera

FIG. 12

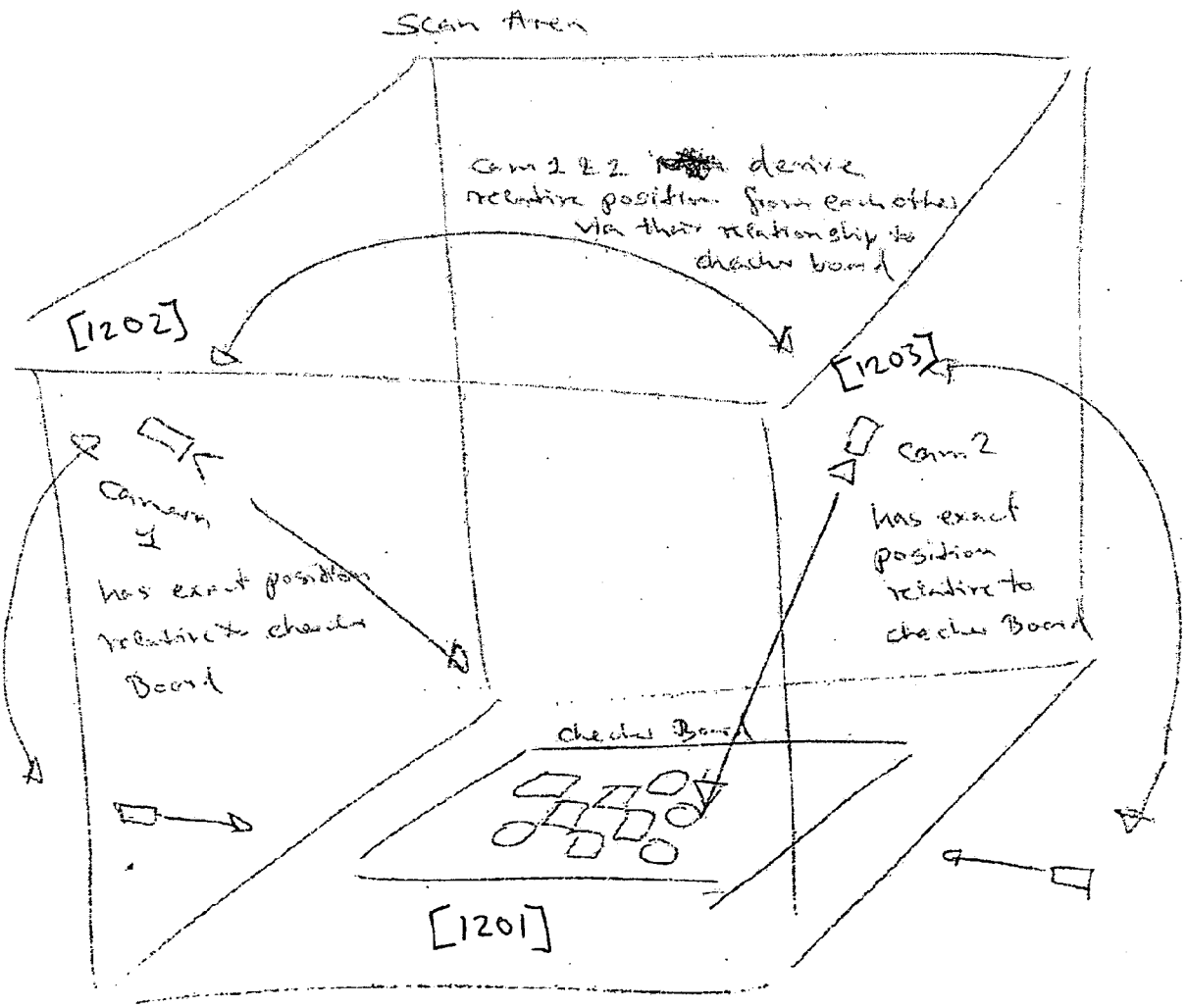

Scan Area
Cam 1 & 2 derive relative position from each other via their relationship to checker board
[1202]
[1203]
Camera 1
has exact position relative to checker Board
Cam 2
has exact position relative to checker Board
checker Board
[1201]

FIG. 13

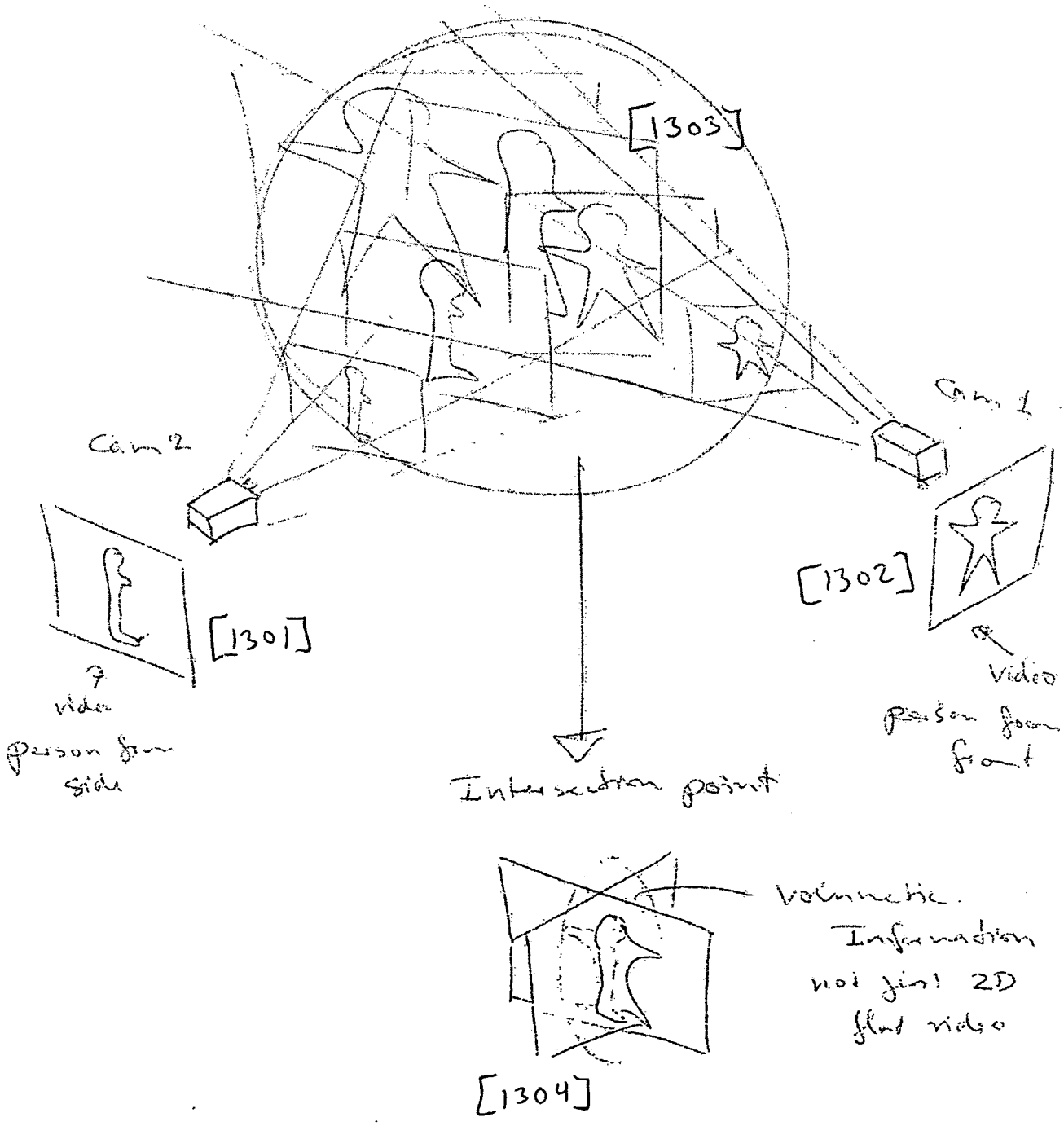

[1303]
Cam 2
Cam 1
[1301]
[1302]
Video person from side
Video person from front
Intersection point
Volumetric Information not just 2D and video
[1304]

FIG. 14

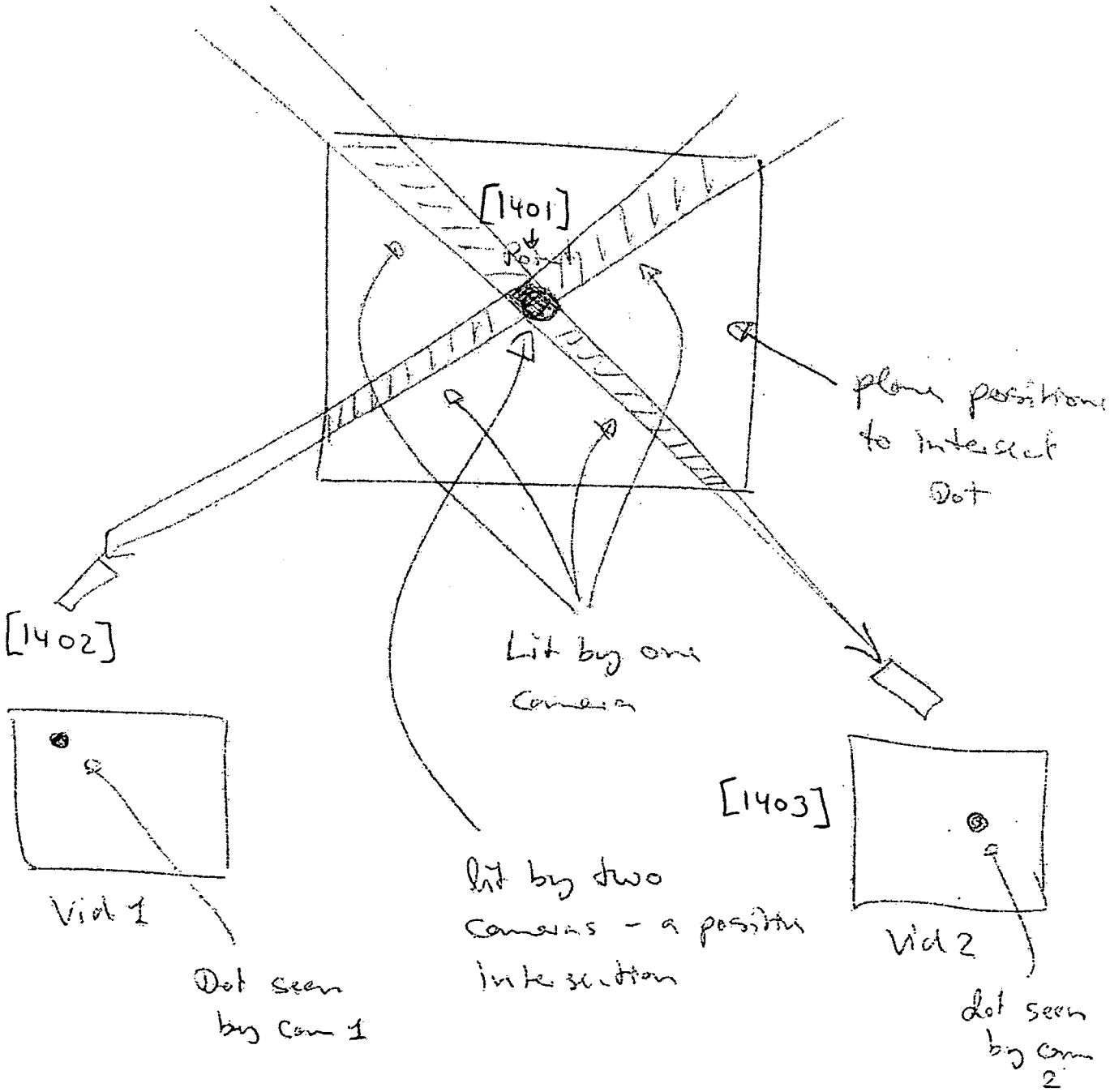

[1401]
Plane positioned to intersect Dot
[1402]
Lit by one camera
[1403]
lit by two cameras - a position intersection
Vid 1
Dot seen by cam 1
Vid 2
dot seen by cam 2

FIG. 15

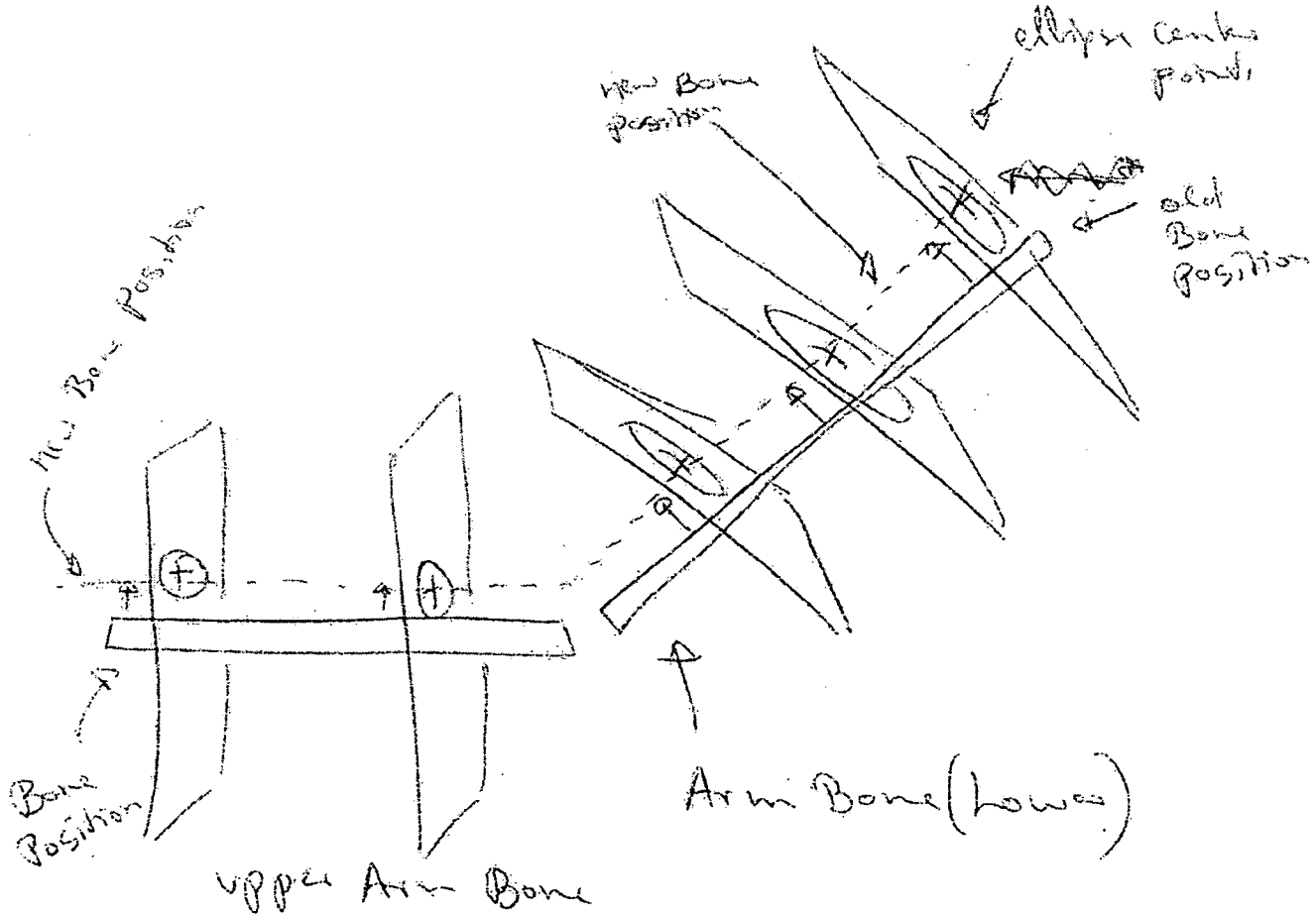
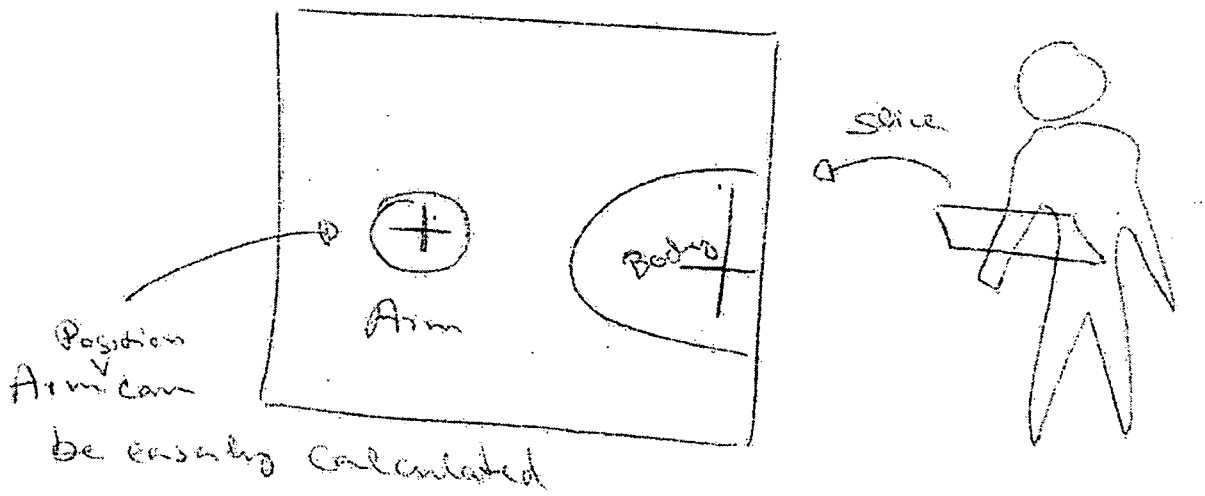

ellipse center point
new Bone position
old Bone Position
new Bone Position
Bone Position
Arm Bone (lower)
upper Arm Bone

FIG. 16


Slice
Body
Arm
Position
Arm can
be easily calculated

Vs

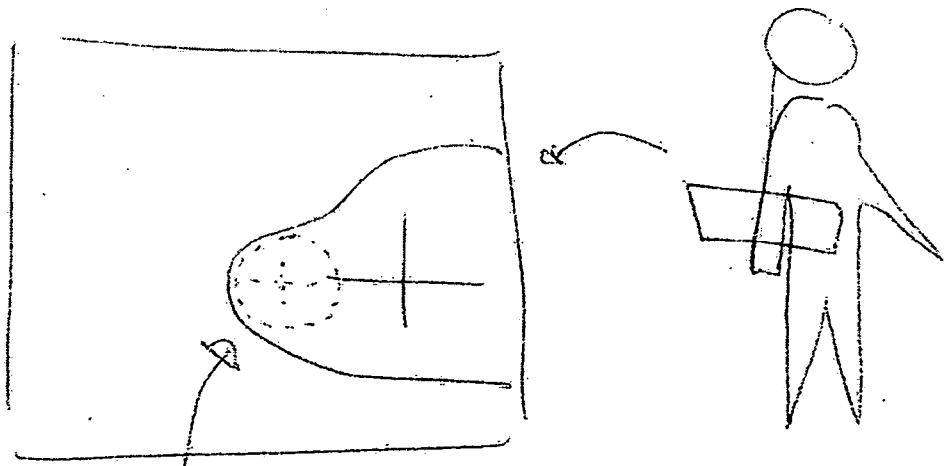
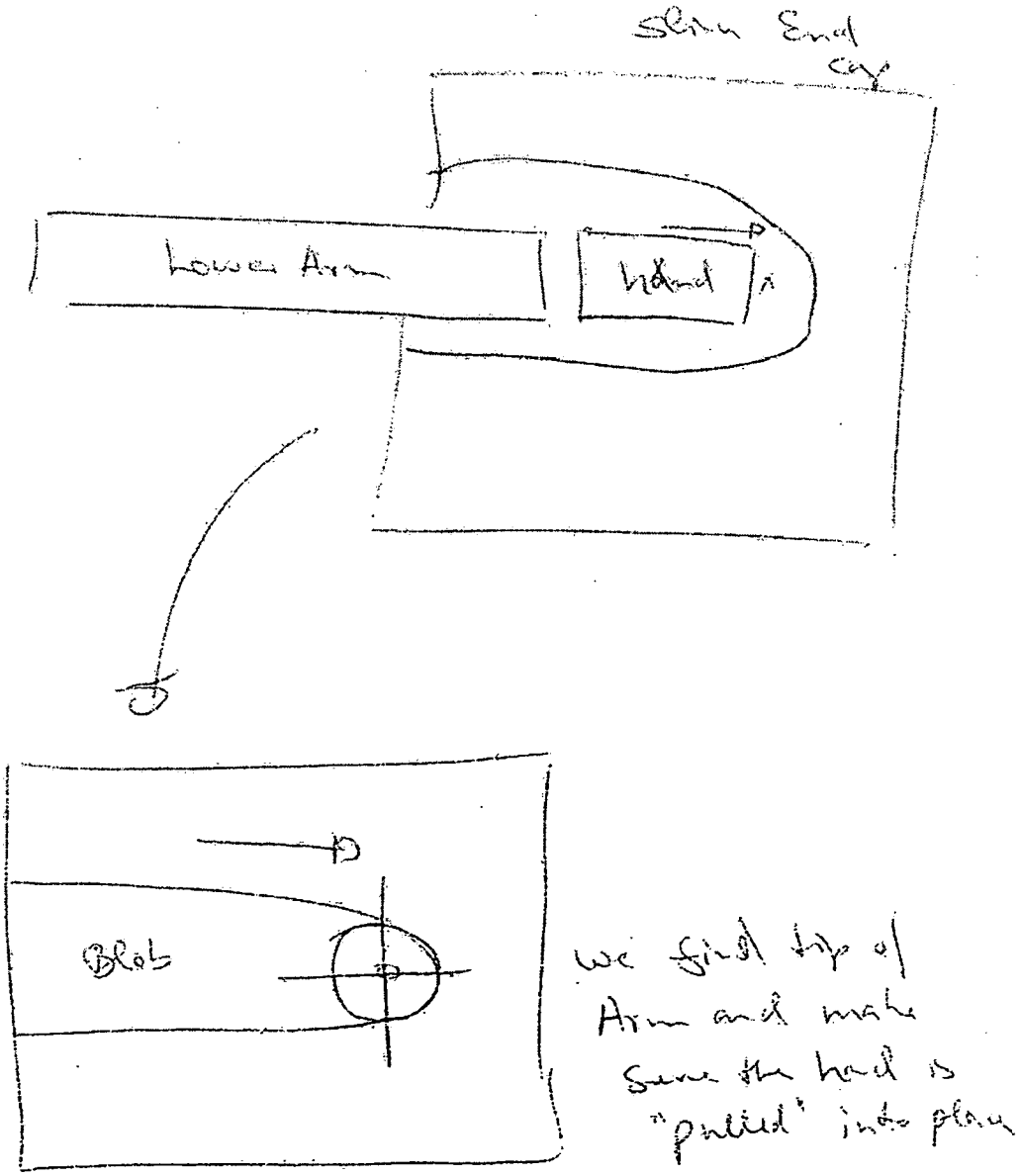
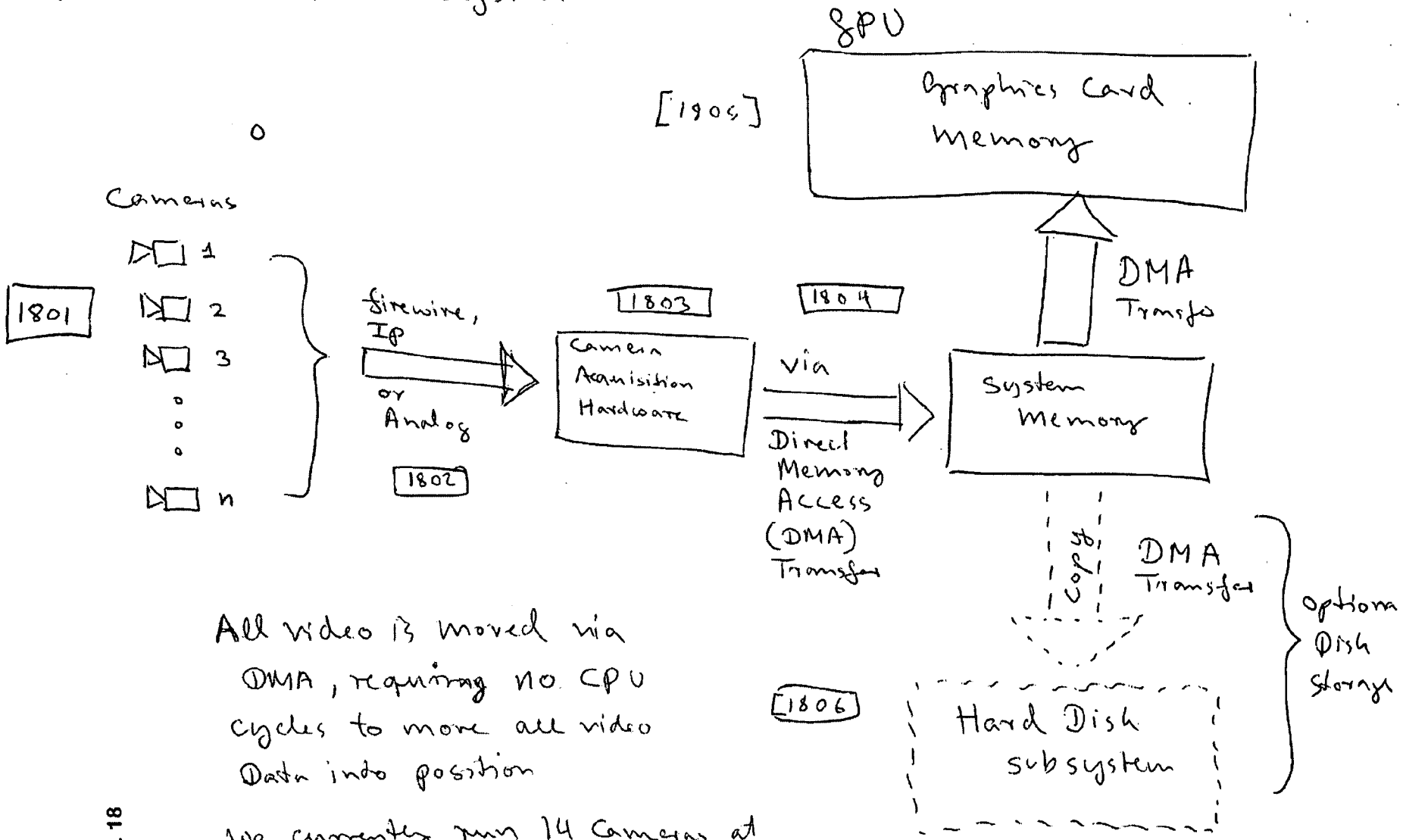

We use the size of the arm and its
Previous location to keep track of its
position even as part of larger objects

FIG. 17


Skin Emd
Lower Arm
Hand
Blob
We find tip of Arm and make sure the hand is "pulled" into plan

Video Stream Subsystem



All video is moved via DMA, requiring no CPU cycles to move all video Data into position

We currently run 14 Cameras at 120 fps on one computer using 6% CPU by moving all transfers to the motherboard North Bridge (Intel) or Memory Controller on chip (AMD)

FIG. 18

FIG. 19

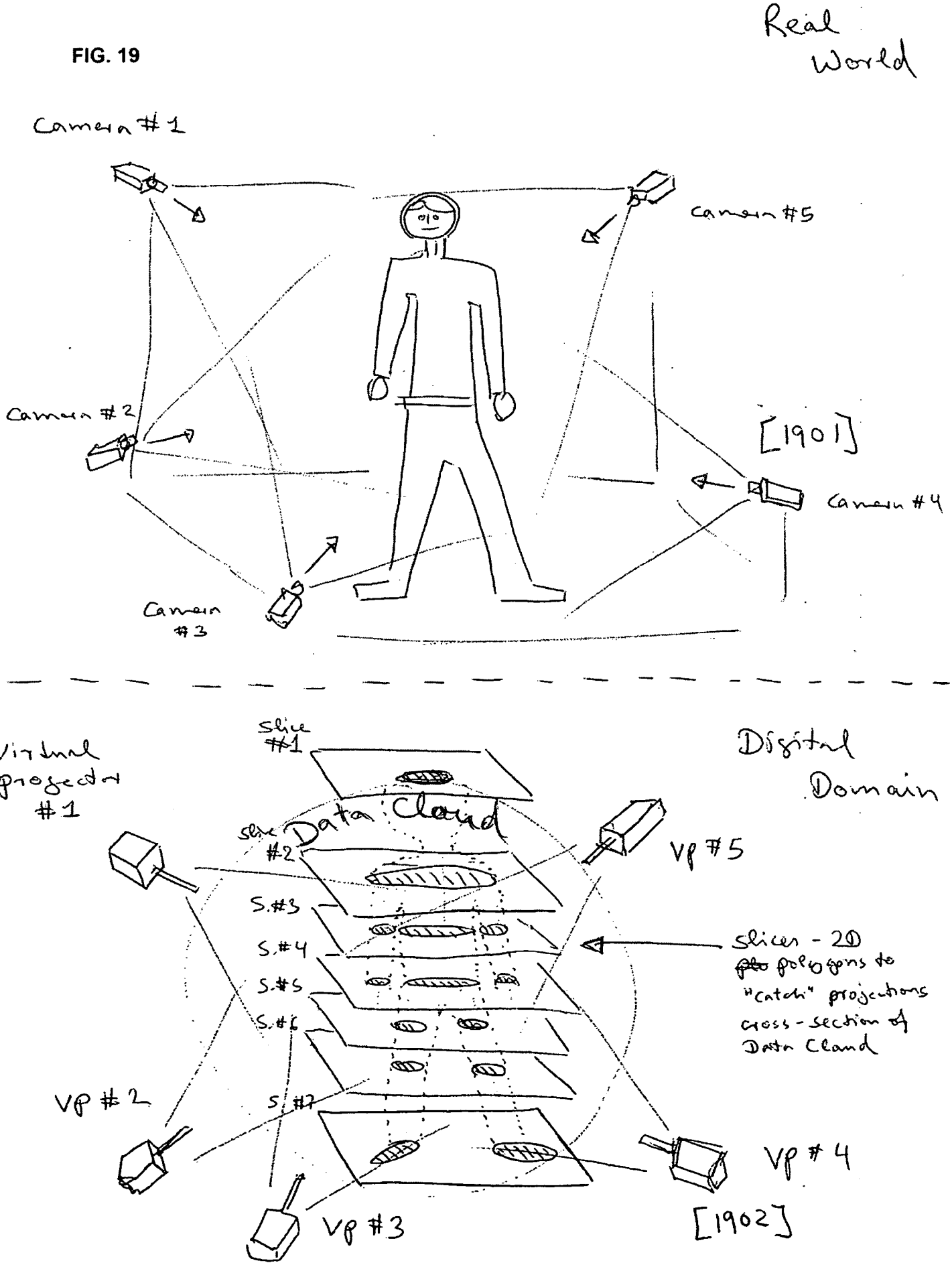
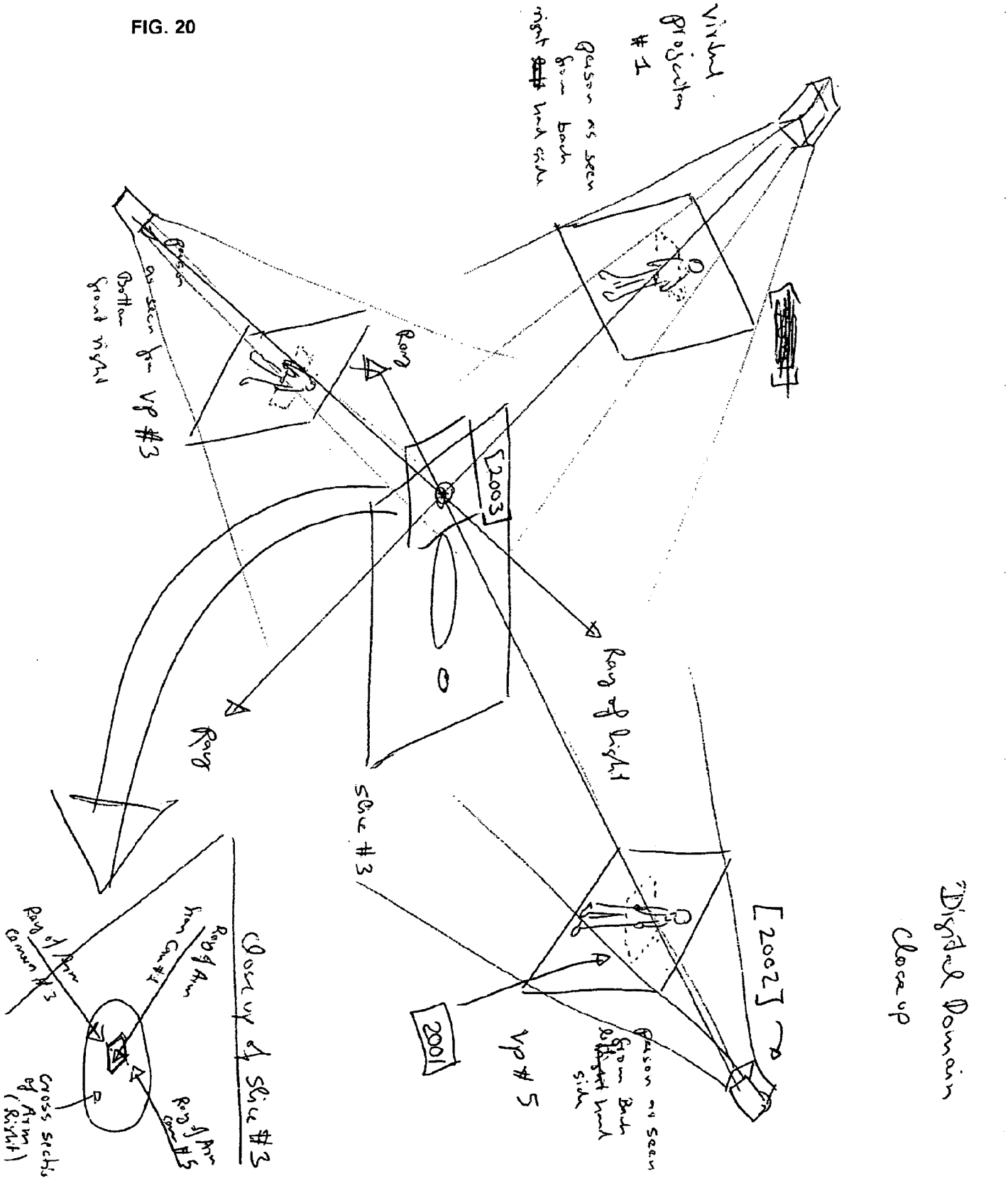

Real World
Camera #1
Camera #5
Camera #2
[1901]
Camera #4
Camera #3
Virtual projector #1
Slice #1
Digital Domain
Data Cloud
Slice #2
VP #5
S.#3
S.#4
Slices - 2D polygons to "catch" projections cross-section of Data Cloud
S.#5
S.#6
VP #2
S. #7
VP #4
VP #3
[1902]

FIG. 20


Virtual Projection #1
Person as seen from back right hand side
Person as seen from Back Right hand side
VP #5
2001
[2002]
Digital Domain
Close up
Ray of light
Ray
Ray
Ray
Person as seen from VP #3
Person as seen from Bottom Front right
2003
Shot #3
Close up of Shot #3
Robot Arm
Robot Arm Camera #3
Robot Arm Camera #5
Cross section of Arm (Right)