ICAT '99
THE VIRTUAL REALITY SOCIETY OF JAPAN

Traces: Wireless full body tracking in the CAVE

Simon Penny, Jeffrey Smith and Andre Bernhardt

Carnegie Mellon University
Robotics Institute and School of Art
and Karlsruhe University
penny+@andrew,cmu,edu, jeffrey@cs.cmu.edu, ab@imdb.com

Abstract

"Traces" is an immersive art project which uses the CAVE stereo-immersive environment for an unorthodox purpose. While most virtual worlds are based on a paradigm of virtual navigation through texture mapped worlds, Traces has no "world" and no navigation. The aesthetic/theoretical goal of Traces is to focus the attention of the user onto their sense their own of embodiment through time. The bodily behavior of the user generates real-time graphics and sound. The technical goal of Traces is wireless full body interaction without the use of standard trackers, joysticks and wands, and without icons, menus or graphical pointers of any kind. To achieve these goals, we have built an infra-red multi-camera machine-vision system which constructs a volumetric body model of the user in real time. We use this body model data to perform all sensing functions in the CAVE. It controls all graphical and sound events, it replaces the wand or joystick for control, it replaces trackers and supplies tracking data. The vision system developed for the Traces project is notable both for its speed and accuracy, and for the fact that it uses inexpensive, off-the-shelf hardware.

Traces was developed off-line at CMU, ZKM and elsewhere in the fall of 1998 and the spring of 1999[1], and was integrated in the CAVE at Institute for Media Communication, Sankt Augustin, Germany, during the summer of '99. Traces won first prize in the Cyberstar98 competition, sponsored by WDR and GMD, Germany. Traces was premiered at Ars Electronica in Linz, Austria September 4–9, 1999.

Key words: Machine vision, CAVE, infrared video, wireless tracking, full-body interaction

[1]Participating researchers: Simon Penny (artist/project director); Andre Bernhardt (vision); Jeffrey Smith (graphics and body model); Phoebe Sengers (agent behavior language); and Jamie Schulte (spatial sound).

Traces has received financial and/or technical support from: Institute for Media Communication, GMD, Sankt Augustin, Germany; Carnegie Mellon Robotics Institute, Pittsburgh, PA USA; The Studio for Creative Inquiry, Carnegie Mellon University; VRCo (distributors of EVL CAVElib); and Ars Electronica

1. Artistic and Theoretical Overview

Traces is the most recent in a series of projects by Simon Penny[2] which focus on the question of embodiment with respect to computational systems. The CAVE is an inherently more embodying technology that head-mounted-display VR: in a CAVE you share your physical space with virtual objects. But in both traditional CAVE and HMD VR applications, the body is reduced to a single data-point in the computational system: the coordinates of the tracker. To address this shortcoming, we have built an infra-red multi-camera machine-vision system which constructs a volumetric model of the whole of the users body in real time. The user is able to interact in a direct bodily way with the computational system and is not encumbered by wires or by pointing devices (which often require learned multi-button procedures).

The standard interactive paradigm for most immersive projects is constrained to virtual navigation through texture-mapped worlds. Within this paradigm are odd kinesthetic inconsistencies, for instance: the user can physically turn to face objects of interest, but cannot physically walk toward them. That is to say, the so called "immersion" is not intuitive. The paradigm of bodily behavior in Traces does not suffer from this confusion. Traces is an unorthodox use of the CAVE, in that there is there is no "world" and no virtual navigation. In Traces, the bodily behavior of the (single) user generates real-time graphics and sound in a limited and static virtual space: a virtual room about double the volume of the physical CAVE. The goal of Traces is precisely *not* to present a panoptic spectacle for the user, but to turn the attention of the user back onto their own sense of embodiment through time. The movement of the user through the space leaves "traces": volumetric and spatial-acoustic residues of user movement; which slowly decay. As time progresses, the traces become more active, and in the last stage of the

[2]See, for example, `http://www.art.cfa.cmu.edu/Penny/works/fugitive/fugitive.html` or `http://www.art.cfa.cmu.edu/Penny/works/petitmal/petitcode.html`

user experience, autonomous entities are spawned by the user, which have complex behaviors of their own.

Just how different Traces is from a "regular" CAVE application became clear as we worked on it. Because the system is effectively "building the world" every time step based on user movement, the computational processes required for Traces are both demanding and unusual. Many of the standard tools, or major aspects of them, were irrelevant to us. Aside from the CAVE operating system (EVL CAVElib), all of the code in Traces is entirely custom. Future iterations of Traces will include the networking of two CAVEs.

2. Hardware installation and environmental constraints

We set out to build a multi-camera machine vision system specifically for the CAVE. This specificity offers both advantages and constraints. Using a technique previously developed by Penny[3], the highly active visual environment of the CAVE was simplified by utilizing only the near IR range of the CCD video cameras, filtering out visible light from the cameras, and lighting the space with IR. Thus two separate optical "channels" were established, and video projection for the consumption of the user did not overlap with illumination of the space for the vision system. The physical structure of the CAVE and the projection path of the video present significant constraints on the placement of both cameras and IR lights. Indeed as every CAVE is different "backstage," each implementation of the camera and light system is configured differently.

The goal for the lighting is to disambiguate the users body from the background. Without active lighting approaches, and assuming monochromatic video, this implies two possible solutions: to light the body against a dark background, or to silhouette the body against a bright background. After numerous experiments we chose the latter. The solution used in Linz was to flood the backs of the projection screens, each with a single 500watt quartz halogen floodlight, each lamp fitted with a custom visible light filter. In addition the "back," or open, side of the CAVE had to be fitted with a screen or curtain, which was also back-lit. In addition, two smaller IR lamps were mounted overhead in the CAVE in order to light the floor. Each lamp was individually dimable to allow tuning of the illumination.

This lighting solution was certainly inexpensive and it performed adequately, although not perfectly due to environmental inconsistencies such as reflectivity of the floor. In addition, we found that the color of clothing in the visible range did not predict its "shade" in the IR range. For example, black clothing would regularly appear white under IR lighting. This is very undesirable, because if the body is white against a white background, no body data can be collected. Given the public nature of Traces (over 200 members of the public interacted with it in three days) a pragmatic solution was adopted: we supplied a reliably IR-black shirt which users were asked to wear .

After numerous experiments with both placement and number of cameras, we found that four cameras was adequate for model building. The cameras were mounted 210cm from the floor in each corner of the CAVE. (This height was determined in part by the optics available for the cameras). Two dimensional calibration using a 1m grid on the floor of the CAVE proved adequate for our needs. The four inexpensive monochrome "board" video cameras were fitted with standard Kodak Wratten IR-pass filter material which eliminated visible light while passing the infrared. The combination of controlled IR lighting and blocking the visible light to the cameras filters away the projected images allowing the system to derive unambiguous data about the user. The four cameras were connected to a quadsplitter (a commercial video device which fits four images in one frame) reducing the resolution of each frame to 320x240, which did not effect the performance of the system adversely. The presence of an (undocumented) frame-buffer in the quad splitter proved to be a cost advantage, since we did not need to use external sync capable cameras, however it does introduce a latency to the system. The use of the "quad" allowed us to digitize all the video data via one single channel monochrome framegrabber, and to process the vision data on a 366 MHz Pentium II PC running Linux.

3. The Vision System

This section describes an algorithm suitable for real time construction of a three dimensional voxel model of a person in a CAVE using inexpensive standard PC hardware. Existing algorithms are discussed with their advantages and disadvantages and finally some speed benchmarks are given.

3.1 Motivation

In order to allow wireless full body interaction for Traces, a vision system had to be developed that is capable of building a three dimensional body model of a person in real time. We had access only to inexpensive standard PC systems, limiting the mathematical complexity of the algorithm. For stability reasons, Linux was chosen as an operating system and the Matrox Meteor board selected for framegrabbing as it is supported by Linux. At the outset it was not clear how many cameras would be needed

[3] http://www.art.cfa.cmu.edu/Penny/works/fugitive/fugitive.html

for a good reconstruction, so the system was designed to work with 3 to 8 cameras.

Based on estimations of acceptable latency in kinesthetic/graphical interaction, we set a minimum acceptable frame rate of 15fps. It became obvious that in order to achieve a good frame rate, there would be a necessary trade-off between temporal and spatial resolution. Our position was that for the project, temporal resolution was more important that spatial. It was not clear that our project was possible at all, especially after researching several existing algorithms for this task which take minutes on powerful machines to compute a single voxel model (which do, however, produce high resolution).

3.2 Existing Algorithms

We isolated two general methods for automatic reconstruction of 3D models. So-called "active" methods use laser scanner or structured light (grids) projected on the object. Passive methods use the plain camera image either from different cameras or a series of images from a single camera (e.g. object on a turntable or camera on a track). While they require less equipment, passive methods are subject to certain limitations. They are, for example, restricted to convex objects. Given the limitations of context (the CAVE) and questions of cost, we chose to pursue the passive approach.

Silhouettes are calculated with the method of difference imaging. A so-called reference image is taken of the empty space lit by the infrared lights. Each new frame is subtracted from this reference image. A person within the space appears dark in front of the bright screens so there is generally a good contrast, resulting in good silhouette images.

There are several algorithms known for the construction of 3D models from multiple views. Some basic work was done by Martin and Aggarwal [4], who introduced the method of "occluding contours" which uses the real object silhouettes. Busch [5] used a polygon approximation of the silhouettes to generate voxel models. Octree representations are generated by the algorithms of Jackins and Tanimoto [6], Meagher [7], Chien and Aggarwal [8], Potmesil [9] and Szeliski [10]. These algorithms largely rely on intersecting voxel- or octree-cubes with the object silhouettes.

These intersection tests are very time consuming. There are essentially two types of intersection tests: either in the silhouette-image plane (2 dimensional test) or the voxel volume (3D test). Niem [3] describes a method of constructing a model based on volume pillars with a fast and simple intersection test routine. For the transformation between the image plane and the volume, a camera model is needed that describes the physical behavior of a camera as exactly as possible (or necessary). A camera model that takes into account almost all of the camera's parameters, including lens distortion, and that is easy to handle and calibrate is the one used by Tsai [1, 2]. The package is available on the Internet and provides all the necessary functions for camera calibration and transformation from world- to image- and camera-coordinates.

3.3 A Different Approach

These reconstruction algorithms were not suitable for our purposes because they aim for a highest possible spatial resolution and don't care about computation time (usually a couple of seconds to minutes even on very fast Sun or SGI workstations). What we needed was an algorithm that is very fast on standard machines with less emphasis on the spatial resolution. Therefore a completely new approach had to be made.

3.3.1 The New Algorithm

The algorithm presented here uses the voxel-space representation rather than the octree representation for the 3D model. Every voxel has the same (physical) size. Instead of transforming a complete voxel from world- to camera coordinates and then perform the intersection tests with the silhouettes, this algorithm reduces every voxel to a single point that is transformed into camera-coordinates and then tested against the silhouette-images. This saves the substantial effort of transforming and intersecting lines. Also, for a very dense voxel-space $\frac{N}{x} \rightarrow \infty$ (where NxNxN is the dimension of the voxel-space and x is the physical size of the space) the result would be identical to existing algorithms, as all the edges of a given voxel in the voxel-space would be transformed into a single point in the image-plane.

To speed things up further, these transformation calculations (which are very time consuming matrix operations) are calculated in advance for every voxel and for every camera and are stored in (very) large lookup-tables. These lookup tables contain pointers to the corresponding pixel in the silhouette picture memory. The storage order is optimized for sequential access by the CPU.

Using four cameras, the CPU only has to get four pointers and look at the four values where the pointers point to in order to determine whether or not a voxel belongs to an object within the space. This is determined by the following two criteria:

(1) a voxel has to be visible in at least three out of four camera images to avoid ambiguity.

(2) if a voxel transforms outside a silhouette in one silhouette image, the voxel does not belong to a person.

For every voxel in the voxel-space this is repeated. The silhouette from each camera is, in effect, projected from the cameras position in a virtual model of the

CAVE space, producing four frustri projecting into the space. Any voxel which is located in three of the four frustri is kept and becomes part of the body model point cloud. This procedure carves away all voxels except those occupied by the body.

Another advantage of this algorithm is that the process of reconstruction always takes the same amount of time no matter if the space is empty, or occupied by one or more persons or objects[4]. The voxel-space is run-length encoded on the fly and stored in a memory efficient way (about 1 kilobyte per frame for a single person in the space).

3.3.2 Simplifications

Obviously in the real world, the condition $\frac{N}{x} \rightarrow \infty$ cannot be realized. The fewer voxels per meter, the larger the error. The maximum size of N is restricted by the amount of memory in the computer (the number of voxels increases as the cube of N, see tables 1 and 2), and the speed at which the CPU can address this memory. The lookup tables for the 1cm resolution for example did not fit into the physical memory of the PC (128MB)! Fortunately the error made by this simplification (i.e. a voxel is reduced to a single point) becomes less significant as the resolution of the silhouette images decreases. That means that with a smaller voxel-space the results improve (within limits) with lower resolution images.

Another important issue for the error is the distance of a person to the camera. The closer a person is to a camera the larger the potential error. First, the camera no longer captures the whole person. Second, and more importantly, the transformation of a voxel with a finite size that is close to a camera covers an area of more than one pixel in the silhouette image. But the algorithm reduces it to one pixel no matter how far away the object is from the camera. One possible solution could be to define a minimum distance to the camera (which could be reflected in the lookup table). In our case it proved sufficient that the cameras were mounted diagonally opposite and perpendicular to each other to disambiguate the information and make up for this error.

3.3.3 Restrictions and Limitations

The simplifications discussed above limit the maximum resolution of the reconstruction. In a large room like a CAVE it does not make sense to aim for a better resolution than 2-3cm using only four cameras with low resolution. Additionally, a large voxel-space requires a lot of memory for the lookup-tables. Even if the frame rate and the number of cameras was not an issue, it is hardly possible to account for the memory usage of very large voxel-spaces. The obvious solution to this problem - get rid of the lookup tables and do the transformation calculation every time - significantly slows down the algorithm (see below). In any case, a high spatial resolution is not desirable because as the resolution increases, the errors increase as well. Multiprocessor and distributed computational solutions also do not seem feasible: it is very difficult to distribute the code that works on the reconstruction of a single frame in real time.

As with all passive construction algorithms it is not possible to detect concave surfaces. For our purposes this is seldom a limitation: our tests have shown that there is practically no posture that a person can assume within the space that is not resolved properly as long as the person is fully visible in all four camera images.

3.3.4 Benchmark Tests

The following tables presents the results of some speed tests. The 3D modeling was done with a spatial resolution ranging from 10cm cubes down to 1cm cubes. As the total size of the voxel-space and the lookup-tables for the transformation increases, the frame-rate drops. For the 1cm resolution, the voxel-space and the lookup-tables did not fit into the memory at all, so a special version had to be compiled that took 8 minutes to compute a single model! These figures are based on the computation of the 3D model using 4 images out of 4 cameras on a standard Pentium II-366MHz PC running Linux.

Resolution	10cm	9cm	8cm	7cm
frames per second	39.2	35.5	30.4	24.5
Number of Voxels	22.5K	30.5K	43.8K	66K
NB_VOXELS_X	30	33	37	43
NB_VOXELS_Y	30	33	37	43
NB_VOXELS_Z	25	28	32	36
SCALE_VOXEL	100	90	80	70

Table 1: A comparison of resolutions and speeds

Resolution	6cm	5cm	4cm	3cm
frames per second	19.4	13.3	8.5	4.1
Number of Voxels	105K	180K	337.5K	800K
NB_VOXELS_X	50	60	75	100
NB_VOXELS_Y	50	60	75	100
NB_VOXELS_Z	42	50	60	80
SCALE_VOXEL	60	50	40	30

Table 2: A comparison of resolutions and speeds (continued)

Figure 3 shows the actual reconstruction. In order

[4]Our tests have shown that in most cases four cameras are sufficient to completely separate at least two persons in the room

to get the required 15 fps, a voxel-space with the size of 60x60x45 voxels was chosen for Traces. With the fixed physical dimension of 3x3x2.5 meters for the CAVE, every voxel therefore represents a 5cm cube. On faster machines the resolution can be better than 15 fps, as long as the lookup tables fit into the memory. However, it is important to note that the memory access speed is more important than the CPU speed. A slower CPU with very fast memory access provides better results than a fast CPU with slow memory access.

3.4 Future Improvements

A perfect silhouette is the crucial part of this algorithm. Using difference images, a tiny spot with a bad contrast to the background creates a "hole" in the silhouette and thus a gap in the 3D reconstruction. This happens even if this hole appears only in one silhouette (On the other hand, using this technique we get noise filtration free: pixel errors due to noise in the silhouette images seldom make it into the reconstructed model).

4. Machine Vision Driven Interaction

The three-dimensional model of the user's body that we construct with this vision system is the data from which we drive all the events which happen inside the CAVE. However, our constructed body model is initially just a cloud of unsorted points which describe the convex volume of the user. Thus, our first task after constructing this cloud is to extract meaningful data about the user's position and configuration.

Ideally, we would like to be able to reconstruct all the relevant information about the user from this cloud: the global position and orientation or the user's whole body, the angles of his or her elbows, knees and hips, the direction in which he or she is looking, and so forth. And, given sufficient time, most if not all of this information could be calculated. However, it is a truism of human-computer interaction that latency is as important as frame rate for forming an impression of interactivity for a user and every millisecond we spend analyzing data is another millisecond of latency in our system. Thus, we have a real motivation for reducing the amount of time we spend reconstructing data about the user from our point cloud and we are forced to be selective about the types of data we do construct.

Since we are using the model of the user's body to drive an interactive art experience, we are interested in certain types of information about his or her pose. Specifically, we thought it most valuable to know the following properties of the user at any moment:

- Position within the CAVE
- Size ("mass")
- The location of the user's head, feet and hands
- Approximations to the velocities and accelerations of these features

Luckily, the position and mass of the user are provided for free during the body model construction and no extra time has to be spend during processing to extract this information. However, this still leaves us with the problem of identifying the user's head, hands and feet and tracking them through time.

We accomplish this identification and tracking by using a few simple heuristics. First and most generally, we can count on a certain amount of frame-to-frame consistency of the body model. That is, during the fifteenth of a second between consecutive frames, the user will not have moved much, or changed the position of his or her hands and feet significantly. Secondly, for each body part to be tracked we can make an a priori guess at where it is likely to be. Specifically, we use the following assumptions:

- the head is usually highest point on the user's body
- the feet are usually the two lowest points
- the hands are those points furthest from the center of mass which are not the head or feet

Clearly, these heuristics are not foolproof and errors in tracking could occur. However, it is somewhat misleading to label the points we track "feet" and "hands" since we are not interested in the hands or feet per se. Rather, we are interested in those points that project from the torso and which are moving. Thus, whether we end up tracking for a few frames an elbow instead of a hand not particularly significant. Furthermore, the user is not being told which points of his or her body are being tracked, and there is no glowing "this is your hand" sign with which the user can find fault. The user simply sees graphical entities spawned by various parts of their body when in motion.

The one instance in which our inability to guarantee perfect feature tracking was a disadvantage was for head tracking. The head in particular is important to track because the stereo illusion is highly dependent on knowing the exact position and orientation of the user's eyes. Since, as noted earlier, we wished to avoid wrapping the user in wires and hardware, we had no information on the location of the user's head apart from that which we could extract from the body model.

Luckily, the EVL CAVE libraries provide hooks for an application to specify the location and orientation of the user's head. Specifically, we wrote our own tracker daemon, supplanting the one in use which read data from the Polhemus sensor. The X-Y-Z location of the user's head (and thus his or her eyes) was generated by using the heuristics we describe above, which were consistently

accurate. In fact, the only time our heuristic would fail would be when the user was so contorted — e.g. head between their legs, looking backwards — that the incorrect identification of the head would go unnoticed. However, the cartesian location of the head is not the only important factor when projecting and drawing stereo images: the angular orientation of the head is equally important. Unfortunately, we were not able to produce as reliable a method for determining the orientation of the user's head as we were location. Our heuristic was that, in general, the user would be looking in the direction perpendicular to the plane formed by the vertical axis of their body and the broadest horizontal axis of their body.

4.1 Using Velocity and Acceleration Data

During the first two phases of Traces — the passive and active trace — the user's body model was used as a three dimensional "brush" to fill in voxels as the user moved through space. The speed and acceleration of the user and his or her limbs affected the simulation only through what volumes they traveled through. The third phase, however, was critically dependent on these computed quantities.

In the final stage of Traces, the user's motions spawned autonomous creatures, which would fly around the space and interact with the user and each other. We did not wish for these agents to be created randomly, or merely based upon where the user was located in space. Instead, we wanted these entities to be "thrown off" by the user as he or she moved enthusiastically through the CAVE. Our tracking of the user's hands and feet provided us with rough approximations to the velocity and acceleration of those body parts which we used to control the creation of these agents.

The flying agents were created either when the hands or feet were moving faster than some threshold (about 1 meter a second) or were moving *s*lower than some threshold, but had a high acceleration; i.e. had recently come to a stop or reversed direction. This simple set of rules gave a convincing impression that swift or violent motions of one's limbs threw off the flying agents, like droplets of water being shaken from wet hands.

4.2 Generating and Displaying Graphics

The generation and display of graphics in Traces presented some unusual challenges. Our choice of what graphics were displayed was constrained by a number of factors, some mundane and some specific to the field of immersive electronic art. The most obvious constraint on the type and number of graphical entities that we could show in the CAVE was the limited processing power available to us. Although a multi-processor Onyx can provide an enormous amount of number-crunching power, only one of our four processors was devoted to graphics display. Furthermore, any graphical object must be drawn eight times per frame: once for both the right and left eyes, for each of the four walls of the CAVE. Given that we required a frame rate of at least ten frames per second to insure a feeling of real-time interactivity, our graphics could take no longer than 12 milliseconds to display. While the problem of limited computational resources is common to all CAVE applications, some of the more common workarounds were not available to us. For example, since we had no pre-existing "world" of texture mapped scenery, and all our graphics were being generated in real time from the body model, we could not perform time-saving preprocessing such as establishing display lists.

In addition to problems of limited resources, the choice of what graphics to use was constrained by artistic decisions as well. The original conception was that a user would create, by their movements, real time sculptural forms which would persist and fade, in a manner reminiscent of a 3D time-lapse photograph, and that these "traces" would take on increasingly autonomous behavior during the user's experience. The constraints of graphics computation mentioned above, along with a desire to avoid gratuitous eye candy, led to the graphical solutions presented at Ars Electronica, in which cubic voxels with procedural transparency formed the traces. Texture maps were avoided except for the virtual room itself.

A curious paradox of stereo illusion opened up during graphics development. We found that by utilizing these abstract volumes (cubic voxels), we jettisoned a range of highly significant depth cues which are cultural, as opposed to physiological in origin. In a conventional representation, a world may contain an avenue of trees receding into the distance, or a road winding to the horizon. We contend that a substantial component of the persuasiveness of the stereo illusion is supplied by the users real-world experience: a row of trees seldom is arranged by size; a road tends not to reduce in width. Hence much of the persuasiveness of VR spatial illusion is a result of users' cultural training, not of the optical stereo created by the technology. This realization became very clear when building the traces forms from abstract volumes. A sphere of a particular radius could appear to be a smaller sphere close to the user or a large sphere further away. Various time varying graphical solutions were tried, from luminous point clouds to 3D crosses to spheres which expanded and became transparent, before we settled on the minimal cubic representation. Part of the motivation for this solution was to avoid any pretense of organic form. There was a desire to be up-front about the fact that this was a computational environment, not some cinematic or hallucinatory pastoral scene.

Although the use of "plain" cubes might seem sim-

plistic, note that during the active and passive traces, an average of 2000 voxels are filled and are being displayed. Anything more complex than an OpenGl primitive slows the display down unacceptably. During the third and final phase of the piece, the autonomous agents — which we nicknamed "Chinese dragons" due to their rough similarity to the segmented dragons used in Chinese New Year's parades — appeared. Since the number of agents drawn at any one frame was small — no more than a few dozen — we were able to experiment with more complicated graphical entities. The Chinese Dragons ended up being scaled chains of spheres which connected to each other via iterative Reynolds flocking.

An amusing aspect of the graphics development was the transition from the 3rd person view (while developing off-line) to the first person view in the CAVE. Solutions which were very satisfying in the simulation mode were useless in first person. For example, if person is moving forwards, the trace is developing behind them and they can't see it! Worse, if they move even a voxel backwards, their head is encased in a form which is the head location at the previous time step, so they can't see anything! Various workarounds were developed for these situations, the best being to establish a "clipping sphere" with a 50 cm radius, centered at the head, which would cull any entities drawn within that volume.

5. Results of work so far and future research agenda

About 200 people interacted with the system at the Ars Electronica exhibition and the response was generally positive. It was amusing to see people emerge from the CAVE sweating, panting and red faced! They really had to do physical work to interact with the system. People lay on the floor, jumped, kicked and danced. This in itself was proof that we have built a dramatically different type of CAVE experience. In order to allow as many people as possible to see the piece in a limited amount of time, we had to manage a timetable in which each user had a 4 minute experience. We storyboarded an experience through three different types of behavior which ramp up in complexity. In order to acquaint the user with the modalities of the system, the first behavior was quite passive. Movement of the user through the space produced a volume of voxels (represented as semi-transparent lilac cubes — see figure 1). These drifted away from the user as if blown by a virtual breeze. The second, more active behavior was based on the dynamics of cellular automata[5]. Voxel volumes left by the user percolated and died away. In the third phase, the user could generate autonomous

[5]Specifically, we implemented Conway's Life in three dimensions.

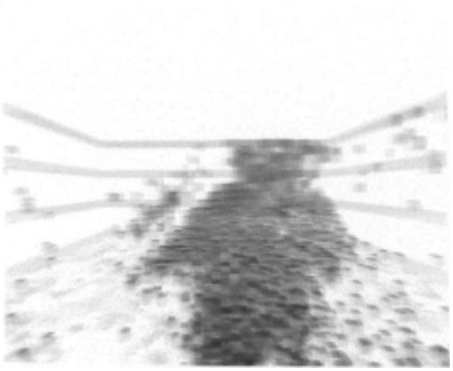

Fig. 1: Third-person view of passive trace in action

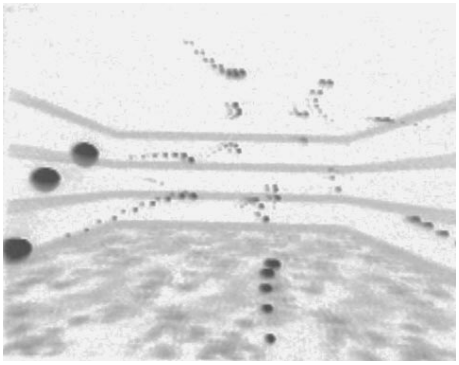

Fig. 2: First-person view of moving "Chinese Dragon" agents

agents ("Chinese Dragons") by making throwing actions (see Figure 2. The agents accelerated away as if thrown, but then interacted with each other (flocking) and with the user.

We have succeeded in developing and publicly demonstrating an entirely new sensor system for the CAVE, which captures the full extent of the users body as usable input data. Untrained users engaged in a physically involving and easily comprehended "bodily" interface to an immersive environment. Users enjoyed engaging directly with dynamic virtual entities. The vision system has proven to be an excellent complement to the CAVE environment. It is inexpensive and in some cases obviates the need for any other sensor system. There are simple but as yet unimplemented fixes for the limitations of the system for reliable stereo image construction.

We have built an entirely new interactive interface for the CAVE which functions outside the standard paradigm of CAVE interfaces and allows new forms of interaction. We have produced new modes of graphical representation (the 3D cellular automata) in the CAVE which differ from conventional CAVE imagery, in that they are entirely abstract, conceptual and non-representational. A wide range of graphical behaviors were developed and more are in process. In particular, the embodied and spatial interaction with semi-autonomous agents offers exciting prospects (Sengers- footnote Phoebes new paper). We will continue the elaboration and presentation of a wide variety of autonomous agent behaviors. An eight channel spatial sound component of the system was developed by Jamie Schulte, but it was not possible to install this at Ars Electronica due to technical idiosyncrasies of the Ars CAVE.

References

1. Roger Y. Tsai, *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1986, pp 364-374.
2. Roger Y. Tsai, *A Versatile Camera Calibration Technique for High Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses*, Journal of Robotics and Automation, Vol. RA-3. No.4, August 1987, pp. 323-344
3. Wolfgang Niem, *Robust and Fast Modeling of 3D Natural Objects from Multiple Views*, SPIE Proceedings "Image and Video Processing II", Vol.2182, 1994, pp. 388-397
4. W.N. Martin, J.K. Aggarwal, *Volumetric Descriptions of Objects from multiple views*, Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-5, 1983, pp. 15 0-159
5. H. Busch, *Ein Verfahren zur Oberfldchenmodellierung dreidimensionaler Modelle aus Kamerabildfolgen*, Ph.D.thesis, University of Hannover, 1991
6. C.L. Jackins and S.L. Tanimoto, *Oc-Trees and their use in representing three dimensional objects*, Comp. Graphics Image Processing, Vol. 14, 1980, pp. 249-270
7. D. Meagher, *Geometric Modeling using Octree Encoding*, Comp. Graphics Image Processing, Vol. 19, 1982, pp. 129-147
8. C.H. Chien and J.K. Aggarwal, *Identification of 3D Objects from multiple silhouettes using quadtrees/octrees*, Comp. Graphics Image Processing, Vol. 36, 1986, pp. 256-273
9. M. Potmesil, *Generating Octree Models of 3D Objects from their Silhouettes in a Sequence of Images*, Comp. Vision Graphics Image Processing, Vol. 40, 1987, pp. 1-29
10. R. Szeliski, *Rapid Octree Construction from Image Sequences*, Comp. Vision Graphics Image Processing: Image Understanding, Vol. 58, No.1, July 1993, pp. 23-32

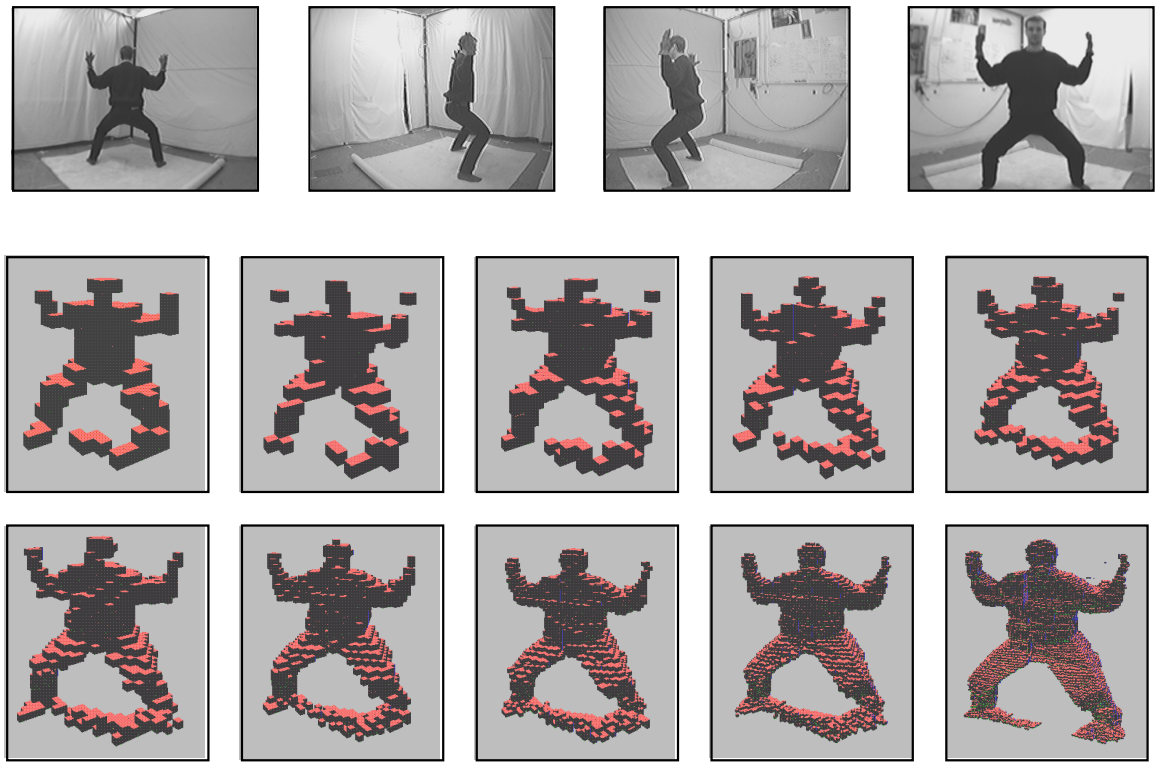

Fig. 3: Different voxel resolutions: the resolution ranges from 10cm cubes (upper left) to 1 cm cubes (lower right)