Multi-User Natural Interaction with Sensor on Activity

Shamsuddin N. Ladha
IITB-Monash Research Academy
Indian Institute of Technology Bombay
sladha@cse.iitb.ac.in

Kate Smith-Miles
School of Mathematical Sciences
Monash University
kate.smith-miles@monash.edu

Sharat Chandran
Department of Computer Science and Engineering
Indian Institute of Technology Bombay
http://www.cse.iitb.ac.in/~sharat

Abstract

The quality of user interactions in virtual environments influence the richness of experiences derived. In many virtual environments, interactions are captured by external visual sensor(s) that observe the environment. This requires end user tracking, and extraction of interaction – a task neither scalable, nor easy to accomplish.

We suggest associating a sensor with the interaction so that the human computer interaction information extraction becomes less difficult. We term this paradigm as sensor on activity*, and we exhibit this paradigm in a scalable multi-user virtual shooting environment built using off-the-shelf components. Here, a sensor (camera) is attached to the activity (shooting). Tracking and inferring position and orientation of weapon, as in traditional setups, is not required any more. Our system is able to support firing at video frame rates. As the sensor need not be pre-configured, the virtual environment does not require instrumentation. Hence it can be easily ported to different locations and is accessible to masses.*

1. Introduction

Virtual environments are spreading virally and the human brain can fail to distinguish between real and virtual environments [3]. This can happen when the vividness of the experiences in virtual environments is at par with those in real environments. The quality of the virtual environment and interactions supported are two key factors that influence the realism of virtual environments. Many of these virtual environments are immersive or semi-immersive and are created using projectors. Projectors, or banks of projectors, are well known for their ability to create large, bright, and crisp displays that mimic real environments.

Interaction techniques have also greatly evolved from the traditional keyboard and mouse based input systems to the myriad of modern day natural interaction techniques based on gestures [8] [14] [20], audio [11] [10], touch [5] [4], body [16] [22] [9], and so on. Typically the environment has one or more sensors external to the human to capture interactions. For example, one or more **visual** sensors are deployed to observe the environment and the interactions. These interactions are generally performed using movement of hands, body, markers, *etc*.

While natural from an engineering perspective, this approach requires appropriate instrumentation and configuration of the environment thereby restricting virtual environments to specific locations. Further, the sensors capture a lot more data in addition to interactions. This requires extraction and possibly tracking of the interaction data. With multi-user environments, such shared sensors capture interaction data for more than one individual thereby further adding to the complexity of interaction extraction. Now the individuals may need to be tracked along with mapping of interactions to the individual. These tasks are neither easy to accomplish nor are scalable.

With the availability of inexpensive off-the-shelf sensors the above disadvantages can be minimized or eliminated by associating *sensor with the interaction, i.e*., each individual has a sensor and the sensor follows the activity performed by the individual. This model is sort of the opposite of the conventional model of external sensors. In this paper we present an application of this paradigm in the context of a shooting range training arena simulator (Fig. 1).

1.1. Related Work

Natural interaction techniques like gestures, voice, touch, *etc*. have, naturally, became prominent [8] [11] in recent times. A widely used scheme to capture natural in-

Figure 1. A camera sitting on top of a weapon depicting *sensor on activity* paradigm for a shooting simulator.

teraction has been to deploy one or more common sensors in the environment, like camera or microphone [19] [8]. Unfortunately, the sensors not only capture the interactions but also the environment. As mentioned above, this complicates the understanding of the human computer interaction.

A novel example of what we term as the sensor on activity paradigm has been successfully demonstrated, albeit with different terminologies. For example, in [7] electromagnetic noise at different locations at home is sensed by a conducive pad, placed on the human body. By connecting this to an analog to digital converter interesting applications have been demonstrated.

[18] is an example of shooting (capturing the face of an opponent) with a camera at real targets (people) in a non-intensive virtual gaming environment. However, other tasks pertaining to shooting are not automated. For example, the photo taken by a shooter is sent to the target (person) for approval. In [6], a single user uses a camera as an alternative pointing device for computers. As the camera moves it observes the monitor, and the pointer position on the monitor is updated using homography. However, for homography computation it is assumed that the entire monitor screen is always visible in the field of view of the camera and the content displayed on the monitor is known and well structured so that the corresponding feature points can be reliably extracted. Further, the minimum distance from which the system can be used is 40cm. These limitations restrict the practical utility of this system.

In our work of a shooting range simulator, various trainees are required to take repeated shots of targets presumably projected on a large screen. How would one figure out where the multiple shots have taken place to determine accuracy and precision? Our first attempt, inspired by the traditional approach [17] [2] [1] [15], had a camera observing the projected virtual world used in conjunction with a laser pointer. In this implementation one of the major challenges was the detection of the laser pointer "dot" in the captured image. This problem is aggravated when multiple trainees use laser pointers simultaneously. Multiple users in the past have been supported using complex hardware, with techniques like Time Division Multiplexing [2], and distinguished color laser pointers. It became quite evident that these techniques were not scalable. By inverting the approach, and placing an off-the-shelf sensor (camera) on the weapon, *i.e.*, implementing the *sensor on activity* paradigm for shooting simulator, the system became portable and scalable.

1.2. Contributions

Our primary contribution is the development of the *sensor on activity* paradigm, wherein the sensor is associated with the interaction and follows the activity.

This paradigm is showcased by the implementation of a **realtime, multi-user, natural interaction** based shooting simulator (more completely described in Sec. 2). In this application an off-the-shelf sensor, a web or a mobile camera, is associated with the interaction (in our case the shooting). This is done by mounting the sensor on the weapon so that the sensor follows the shooting activity, which can be defined as aiming followed by firing.

The portability of the sensor-on-activity begs the use of the simulator in unusual environments, such as the bedroom in a home. At home, or in office cubicles, finding a large single planar display surface for projection could be difficult. We are easily able to move our implementation to typical dual planar surfaces (Fig. 6).

2. Sensor on Weapon (SoW)

A sensor, a.k.a camera, is mounted on the weapon as shown in Fig. 1). The sensor sees a First Person View of the activity thereby eliminating the problem of tracking the weapon and the shooter. In the simulator, a computer generates a virtual shooting scene with targets. Targets can be, for instance, soldiers or tanks. The virtual scene is projected onto a planar display surface as a First Person View (FPV). Trainees then aim and shoot at targets displayed on the scene. The challenge lies in the fact that targets are dropped asynchronously and with random gaps; further, trainees are required to conserve "bullets". Bullets 'cost' a lot, so the trainee is not expected to fire in burst mode. All of this translates to the obtained camera image being blur free since there is sufficient pause before shots.

2.1. Homography Computation

The virtual scene is rendered in the FPV, and what one might call the Projector Co-ordinate System (PCS). Once shots are "fired", using image processing techniques we can identify the shot (hit point) in the captured image, *i.e.* in the camera coordinate system (CCS). Our goal is to find the hit point in PCS. Fig. 2 illustrates this concept. For training

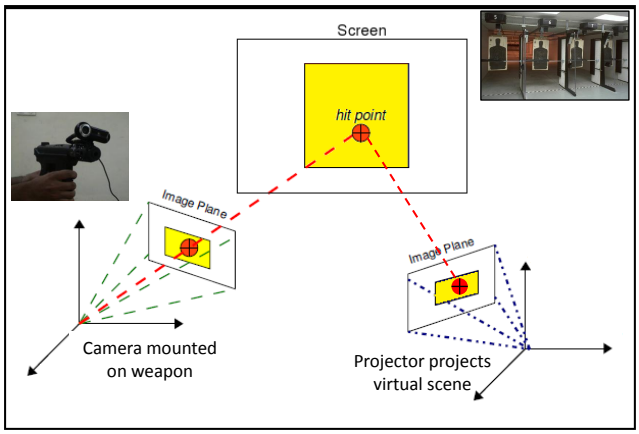

Figure 2. Hit point computation. The hit point as seen by a trainee is also observed by a camera (left). This point must be mapped to the projector coordinate system (right) to verify the accuracy.

purposes, it is often sufficient to assume that the depth in the virtual scene is limited. Therefore we can measure the (two-dimensional) accuracy by mapping hit point, C_h, from CCS to hit point, P_h, in PCS using the 3×3 homography matrix, H, as shown in Eq. 2.1.

$$P_h = HC_h \tag{2.1}$$

The two important aspects of the system are *accuracy of hit point* in PCS and *response time*. Since the virtual scene is known aprirori, the use of homography allows sufficiently accurate hit point computation with real-time response. Thus relatively computationally complex techniques like SIFT [12] are not required for registration of the virtual scene in the camera space.

Each camera mounted on weapon captures images of the projected scene on the display surface. As long as the position and the orientation of the camera with respect to the weapon doesn't change the hit point in the camera space remains fixed and can be pre-determined. Without loss of generality, the center of each camera frame is considered as the hit point in CCS *i.e.* C_h. For example, if the camera frame resolution is 640×480 then the hit point in CCS is $(320, 240)$. Thus hit point detection in CCS is trivial and requires no additional processing. However, the camera mounted on the weapon is not stationary as in traditional setups. It moves along with the shooter and her weapon, thereby warranting re-computation of the homography, H, between CCS and PCS. As the bullets are expensive random firing is not expected and homography computation is required only when shots are fired. Once the homography is computed, hit point in PCS can be computed by forward matrix multiplication.

It is well known that only four points (no three of which are collinear) are required to compute homography. Therefore, as a proof of concept, we project the four marker pattern shown in Fig. 3 along with the virtual scene to efficiently compute homography matrix. In our implementation, we use a wall as the projection screen. In actual practice, we expect that these markers are attached to the projector screen, or four end points of the projection area are detected, or suitable markings are made on the display surface to aid homography computation.

The task now is to detect this marker pattern in the camera frame whenever a shot is fired. The process of aiming (pausing) and shooting helps prevent optical blur despite the use of commodity cameras. The patterns are well known and can be easily detected. The markers and targets are designed to lie far apart in the color space to aid marker detection. As long as the ordering of the markers is not altered in the CCS, *i.e.* the orientation of the captured image is less than $45°$, mapping markers from CCS to the corresponding known markers in PCS is straightforward. Once the markers are detected in CCS, the homography matrix is computed. Hit point is then computed using Eq. 2.1. To detect if a shot has been successful, a check is done to figure out whether the hit point in PCS lies within any target boundary. The system is scalable and supports multi-user shooting.

Figure 3. Sample four marker pattern for homography computation.

3. Experiments

Initially the experiments were conducted based on the traditional paradigm of sensor observing the environment. Accordingly, we had a laser pointer attached to each weapon and a single stationary camera observing the environment. As the camera remained stationary, H was computed only once during the pre-configuration of the environment. The laser pointer was used to fire shots. Whenever a shot is fired, a color dot is created on the display surface, which is observed by the camera. The prominent tasks in this type of setup is the detection of the laser dots in the camera captured image and tracking of multiple laser dots. Fig. 4 shows multiple laser dots captured by the camera.

To confidently extract the laser dots from the virtual scene in the camera captured images certain characteristics of the laser dot need to be used. These features include the brightness of the laser dot compared to the other elements of the virtual scene, its size and aspect ratio, gradual decrease in the intensity away from the center of the dot, and so on.

Figure 4. Multiple laser dots as seen in the camera. Notice how a red colored laser dot appears white in the image due to saturation.

Once the position of the laser dot is identified in CCS, the hit point in PCS is trivially computed. This system when implemented was seen to have several drawbacks:

- Detection of laser dot is not straightforward thereby increasing response time and reducing scalability of the system. With age, and usage, the spectral response of the laser pointers change drastically.
- The laser dot provides a visual cue to the trainee, which may not be desirable in certain shooting environments.
- With multiple trainees shooting simultaneously, the laser dots can come too close to each other or overlap thereby confusing the system. This can easily happen when more than one trainee aims at the same target.

The sensor on weapon approach doesn't suffer from these drawbacks. Fig. 5 shows the experimental setup with different components used. The system relies on off-the-shelf components, a Logitech Pro 9000 webcam as a sensor that captures video frames at a resolution of 640×480 and a Sharp XR-1X multimedia projector with native resolution of 800×600 pixels. The application runs as a client server system on two different machines each with only 512MB RAM and a Pentium IV processor.

Our application, written in C++, is based on client server model and uses OpenCV and OpenGL. The two programs (client and server) can run on the same or two different machines. The server creates the virtual environment, in this case a shooting scene, with embedded four marker pattern and multiple targets (colored rectangular blocks). This virtual scene is then projected onto a display surface. The client controls the camera that sits on top of the weapon.

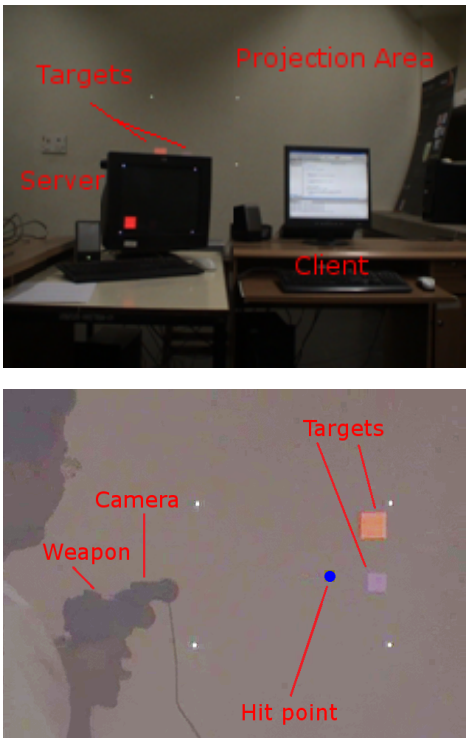

Figure 5. Top: Experimental setup. The virtual scene, with red and purple rectangular blocks as targets, displayed on the monitor labeled server is rendered on the display surface by a projector sitting behind the server; Bottom: A snapshot of the shooting simulator. Note that the blue dot (hit point) is for experimental purpose and it is not displayed in the real system.

Each shooter's camera is controlled by a separate instance of the client. On initialization, the client detects the camera attached to the system and registers with the server to obtain its unique ID useful to track the client on server. The shooter moves the weapon to aim at a target and then shoots the target ensuring that all the four markers are in the camera's field of view. On every shot P_h is computed by the client and sent to the server. If the server determines that any target has been hit then that target is moved to a different location in the virtual scene and the game continues.

As mentioned earlier, the portability of the sensor-on-weapon enables usage in casual environments. The implementation was easily shown on a dual-planar surfaces with a couple of changes. A single homography matrix can no longer map hit point from CCS to PCS, since each planar side of the dual planar surface will have its own homography matrix. This requires modification to the projected marker pattern, with at least four markers on each planar side. To position these markers on either side of the corner, the corner is detected in PCS using the method described in [13]. During the simulation, based on the location of the hit point in CCS, appropriate homography matrix is used to compute correct hit point in PCS. Projection on dual-planar

surface results in distortion of the geometry of the virtual scene. This distortion is also compensated by projecting a pre-corrected image A sample input and the corresponding output on a room corner in our laboratory is shown in Fig. 6.

4. Observations

Our system uses off-the-shelf components, runs on low end hardware with unoptimized code. Despite all this, the approximate marker detection and homography computations times are only 22 milliseconds and 0.15 millisecond respectively. Thus the system is capable of supporting about 45 shots per second. Further, since each trainee shoots independently of others, the response of the system doesn't degrade with increasing number of trainees.

Figure 7. A schematic of the sixteen marker pattern that allows a shooter to fire even if the entire virtual world projection is not in the field of view of the camera.

To compute homography at runtime the system requires all four markers be in the camera's field of view, which can be a limiting factor in certain environments. To circumvent this problem a sixteen marker scheme, shown in Fig. 7, is proposed. With the new marker scheme if four markers, in any one corner of the display screen, are visible to the camera then the homography matrix can be computed reliably. Although the area of the quadrangle occupied by the markers is smaller than that in the four marker pattern, the accuracy of the hit point in PCS does not degrade because the homography matrix is computed accurately. The four markers in a corner are designed such that one marker is larger than the remaining three and horizontal and vertical distances between these markers are different. These extra features aid in correct identification of the screen corner that is visible in the camera frame irrespective of the location, and field of view of the camera.

5. Conclusion

This paper has demonstrated the effectiveness of the proposed paradigm of sensor on activity. This has been done by developing a virtual shooting simulator using off-the-shelf components. This paradigm is competitive to the custom hardware based solutions such as the Nintendo Wii [21], considering its cost effectiveness, portability and accessibility. Mobile phones with inbuilt camera and increasing computational power can be potentially used as sensors thereby further increasing the accessibility to this type of applications. The simulation shown here is a prototype and can be enhanced to achieve better realism. Specifically, 3D shooting can be supported. Although our application is based on point and shoot activity, the paradigm can be effectively applied to other activities as well. This demonstrated extension of application to dual planar surfaces, like room corners, allows a deeper penetration of virtual environments, to masses who may not have access to a large single planar real-estate required for projection.

Acknowledgments
The authors are grateful to Aman Parnami and Rohit Jhunjhunwala for their contributions to the earlier version, and to Sushil Meena for his contributions to the implementation on dual-planar surfaces

References

[1] Ahlborn, Benjamin and Thompson, David and Kreylos, Oliver and Hamann, Bernd and Staadt, Oliver. A practical system for laser pointer interaction on large displays. In *Symposium on Virtual Reality Software and Technology*, pages 106–109, 2005. 2

[2] Andriy Pavlovych and Wolfgang Stuerzlinger. Laser pointers as interaction devices for collaborative pervasive computing. In *Advances in Pervasive Computing*, 2004. 2

[3] Blascovich, Jim and Bailenson, Jeremy. *Infinite Reality: Avatars, Eternal Life, New Worlds, and the Dawn of the Virtual Revolution*. William Morrow, 2011. 1

[4] Bravo, J. and Hervas, R. and Sanchez, C. and Chavira, G. and Nava, S. Touch-based interaction: An approach through NFC. In *International Conference on Intelligent Environments*, pages 440–446, 2007. 1

[5] Broll, Gregor and Graebsch, Roman and Scherr, Maximilian and Boring, Sebastian and Holleis, Paul and Wagner, Matthias. Touch to play – exploring touch-based mobile interaction with public displays. In *International Workshop on Near Field Communication*, pages 15–20, 2011. 1

[6] Cantzler, H. and Hoile, Cefn. A novel form of a pointing device. In *Vision Video and Graphics*, pages 57–62, 2003. 2

[7] Cohn, Gabe and Morris, Daniel and Patel, Shwetak and Tan, Desney . Your noise is my command: Sensing gestures using the body as an antenna. In *Conference on Human Factors in Computing Systems*, pages 791–800, 2011. 2

[8] Cowan, Lisa and Li, Kevin. Shadowpuppets: Supporting collocated interaction with mobile projector phones using hand shadows. In *Conference on Human Factors in Computing Systems*, pages 4111–4116, 2011. 1, 2

[9] Knies, Rob. Kinect body tracking reaps renown. http://research.microsoft.com/en-us/news/features/kinectskeletal-092711.aspx. 1

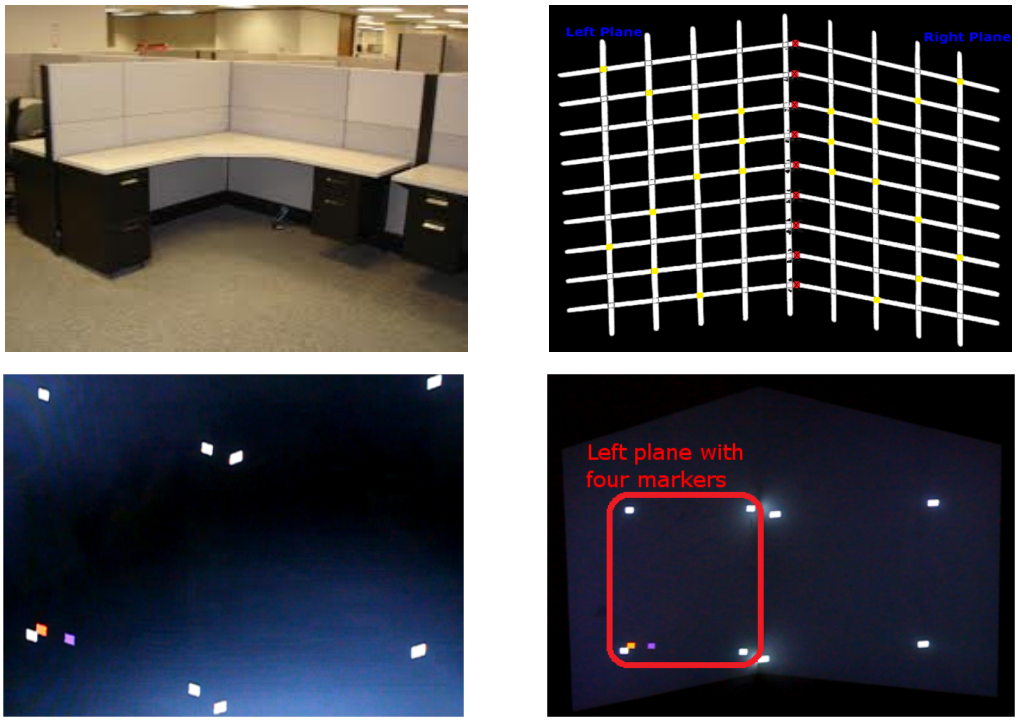

Figure 6. The shooting simulator on a dual planar surface. Top left: A typical office cubicle as dual planar surface; Top right: Sample calibration patterns used to detect the edge of the dual planar surface in the camera space; Bottom left: Geometry corrected input to the projector, Bottom right: Geometry corrected output with four markers on each side of the corner for homography computation

[10] Kumar, Anuj and Paek, Tim and Lee, Bongshin. Voice typing: A new speech interaction model for dictation on touchscreen devices. In *Conference on Human Factors in Computing Systems*, pages 2277–2286, 2012. 1

[11] Lazar, Jonathan and Feng, Jinjuan and Brooks, Tim and Melamed, Genna and Wentz, Brian and Holman, Jon and Olalere, Abiodun and Ekedebe, Nnanna. The soundsright captcha: An improved approach to audio human interaction proofs for blind users. In *Conference on Human Factors in Computing Systems*, pages 2267–2276, 2012. 1

[12] Lowe, David. Object recognition from local scale-invariant features. In *International Conference on Computer Vision*, pages 1150 –1157, 1999. 3

[13] Paidimarri, Kashyap and Sharat, Chandran. *Computer Vision, Graphics and Image Processing*, volume 4338 of *Lecture Notes in Computer Science*, chapter Ad-Hoc Multiplanar Projector Displays, pages 289–298. Springer Berlin Heidelberg, 2006. 4

[14] Pranav, Mistry and Pattie, Maes and Liyan, Chang. Wuw - wear ur world: A wearable gestural interface. In *Extended Abstracts, Conference on Human Factors in Computing Systems*, 2009. 1

[15] Si-Jung, Kim and Moon-Sung, Jang and Kwang-Chul, Jung and Hong-Sick, Kim. An interactive user interface for computer-based education:the laser shot system. In *World Conference on Educational Multimedia, Hypermedia and Telecommunications*, pages 4174–4178, 2004. 2

[16] Silva, Mara and Bowman, Doug. Body-based interaction for desktop games. In *Conference on Human Factors in Computing Systems*, pages 4249–4254, 2009. 1

[17] Soetedjo, Aryuanto and Nurcahyo, Eko and Nakhoda, Yusuf Ismail. Development of a cost-effective shooting simulator using laser pointer. In *International Conference on Electrical Engineering and Informatics*, pages 1–5, 2011. 2

[18] Suomela, Riku and Koivisto, Ari. My photos are my bullets - using camera as the primary means of player-to-player interaction in a mobile multiplayer game. In *International Conference on Entertainment Computing*, pages 250–261, 2006. 2

[19] Tang, Matthew. Recognizing hand gestures with microsoft's kinect, June 2011. http://www.stanford.edu/class/ee368/Project_11/Reports/Tang_Hand_Gesture_Recognition.pdf. 2

[20] Wickeroth, Daniel and Benolken, Paul and Lang, Ulrich. Markerless gesture based interaction for design review scenarios. In *International Conference on Applications of Digital Information and Web Technologies*, pages 682 –687, 2009. 1

[21] Wikipedia. Wii remote — Wikipedia, the free encyclopedia, August 2012. http://en.wikipedia.org/wiki/Wii_Remote. 5

[22] Yuelin Liu and Xiaobo Lu. Embodied interaction – an analysis based on chinese philosophy of body. In *International Conference on Computer-Aided Industrial Design Conceptual Design*, pages 984–987, 2010. 1