Proceedings of the Second IASTED International Conference
HUMAN-COMPUTER INTERACTION
March 14–16, 2007, Chamonix, France
ISBN Hardcopy: 978-0-88986-654-6 / CD: 978-0-88986-655-3

A CAMERA-BASED POINTER WITH VISUAL FEEDBACK

Makio Ishihara
Faculty of Information Engineering
Fukuoka Institute of Technology
3-30-1 Wajiro-Higashi, Higashi-ku
Fukuoka, Japan
email: m-ishihara@fit.ac.jp

Yukio Ishihara
Faculty of Engineering
Yamaguchi University
2-16-1 Tokiwadai
Ube, Yamaguchi, Japan
email: i.yukio@yamaguchi-u.ac.jp

ABSTRACT
In this study, we introduce a camera-based pointer. A camera-based pointer is a pointer system where a user uses a camera to point at a spot on a screen instead of a laser pointer. In the system, the spot on the screen is located by visual feedback. That is, a cone image is projected on the screen and then it is taken by the camera to send back to the system. Using the visual feedback, the system moves the cone image towards the spot, so that the system will find the center of the cone image right in front of the camera in several iterations. Thereby this system does not require any homographies between the screen and the camera coordinates systems to be made in advance. In addition, a cone image is scaled in a liner gradient so that it is robust for blurring caused by an out-of-focused camera. This robustness enables a user to move around in front of the screen while pointing the camera on it. Finally we demonstrate Ping pong that works with two camera-based pointers in order to show that a camera-based pointer technique is practical.

KEY WORDS
Visual feedback, laser pointer interaction, projector-camera system, augmented reality, human computer interaction.

1. Introduction

Human computer interaction (HCI)[1] is becoming more important in helping people interact with computers more intuitively and implicitly. Augmented reality or AR[1] has recently been studied in order to enhance HCI. AR enables users to see the real world with an overlay of additional information. Users wear see-through head-worn-displays, and their position and orientation are monitored by tracking techniques. As a result they see the overlaid information through the displays as if it was attached to the real world.

There have been various AR systems reported [2, 3, 4]. For example, Höllerer, T. et al.[3] built an experimental mobile AR system that helps a user orient him/herself in an unfamiliar environment. He/she sees the real world in miniature and route arrows overlaid on it, which points to the way to a location that he/she has requested. Rekimoto, J.[4] introduced an idea of augmented interaction. It assists and enhances interactions between a user and the real world. In the system, 2D barcodes are attached to the real objects such as doors, books, shelves, name cards etc. Then he/she acquires additional information about the real objects by looking directly at them.

Stereoscopic displays such as see-through head-worn displays are useful for 3D representations. They cause, however, perceptual errors due to conflict between accommodation and vergence in the visual system[5]. Compared with stereoscopic displays, monoscopic displays such as projector-based displays do not cause this problem and are also useful for 2D representations. Fjeld, M. et al.[6] built a planning tool that enables users to cooperate in a virtual environment for planning real objects such as rooms, schools, factories etc. Beardsley, P. et al.[7] built a handheld-projector that enables a user to create an opportunistic display on any nearby surfaces.

For monoscopic displays, projector-camera systems have been studied most [8, 9, 10, 11, 12, 13]. Bimber, O. et al.[8], for example, built a system for view-dependent stereoscopic visualization on geometrically complex ordinary surface using gray-code. Wallance, G. et al.[9] built a system that enables arbitrary tiled displays to be aligned automatically using structured patterns. Thus projector-camera systems are usually used for calibration.

Along with those calibration techniques, interaction techniques such as pointing, selecting, clicking using a laser pointer have been dominant as well in the field. Laser pointer interaction is an interface that enables a user to point at a spot on the screen by a laser pointer. Olsen, D.R. et al.[14] built a prototype that enables a user using a laser pointer to interact with the information on a projected screen. Matveyev, S. et al.[15] introduced multiple pointer interaction. In their system, a user can split the laser beam into three, so that the correlation between the three laser spots on the screen can be employed for interaction. Oh, J. et al.[16] built a collaborative environment where users can use laser pointers to interact with the information on a screen simultaneously.

In those systems, it is a common way that the spot on a screen, which has been pointed by a laser pointer, is captured by a camera and then the pixel on the camera image is mapped to the corresponding point in the screen coordinates system. This way requires the camera to be precisely focused on the screen, and the homographies between the

camera and the screen to be made up in advance. In this study, we take an approach to a system that does not rely on focused cameras and homographies. The basic idea comes from the robustness of an image scaled in a liner gradient for blurring and the deepest descent method with visual feedback. This paper shows that they compensate well for out-of-focused cameras and no homographies. In our system called a camera-based pointer, a user uses a camera instead of a laser pointer. A projector projects a gray-scaled cone image to a screen and then the camera captures the image on the screen. The system calculates the gradient at the pixel in the center of the image and then moves the cone image in the descending direction so that the top of the cone image moves gradually to the spot where the camera has been pointed. This way needs several iterations to locate the spot on the screen. At this point, we present Ping pong that works with two camera-based pointers in order to show that a camera-based pointer technique is practical.

In Section **2**, we introduce the deepest descent method with the visual feedback of a cone image and how it works for a camera-based pointer. In Section **3**, we specify the design of two camera-based pointers and then we describe the performance and the pros&cons in Section **4** and **5**. In Section **6**, we present Ping pong game that works with two camera-based pointers to show that a camera-based pointer technique is practical. Finally we give the concluding remarks in Section **7**.

2. Deepest descent with visual feedback

A projector projects the gray-scaled cone image shown in Fig. 1. The pixel value at the top is 255 and the value of the base is 0. A part of the cone image is captured by a camera and then the cone image is moved and also resized in order to meet the two objectives below.

1. The image taken by the camera (the camera image) is totally covered by the cone image. This objective is to prevent the camera from being out of the cone image even if it is away from the screen.

2. The cone image is captured in the center of the camera image. This objective is to find out where the camera looks on the screen.

The projector projects the cone image again and then this visual feedback process is iterated to complete the objectives.

There are two coordinates systems. One is a screen coordinates system and the other is a camera coordinates system. In the screen coordinates system, there is the cone image stuck on the plane that is supposed to be projected. The side length of the cone image and the position of the center of the cone image in the coordinates system are represented by $cone\_len$ and $(cone\_x, cone\_y)$, respectively. In the camera coordinates system, there is the camera image itself. The size of the camera image and the

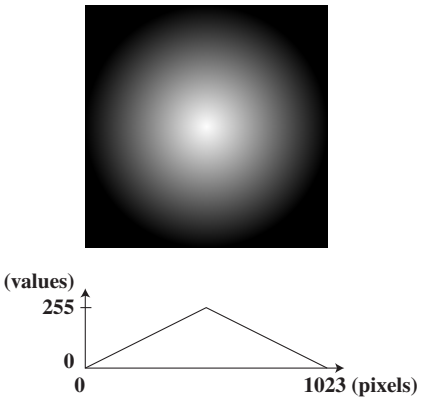

Figure 1. A gray-scaled cone image

position of the center of the camera image in the coordinates system are represented by cam_w and cam_h, and (cam_x, cam_y), respectively.

For the first objective, the magnitude of the gradient of color at the pixel in the center of the camera image is used. First, the target of the magnitude is defined in advance. This target is referred to when the cone image is resized.

$$grad\_trgt = \frac{255}{cam\_w/2.0} \tag{1}$$

Equation(1) represents the magnitude of the gradient of color between the center and the side of the camera image after the cone image is stretched over the camera image. The current gradient $(grad\_x, grad\_y)$ of color at the pixel in the center is given as follows.

$$\begin{aligned}(grad\_x, grad\_y) \;=\; & (\partial f_c(cam\_x, cam\_y)/\partial x, \\ & \partial f_c(cam\_x, cam\_y)/\partial y)\end{aligned} \tag{2}$$

, where function $f_c(x, y)$ represents the value at the given pixel in the camera image. In this case, the center of the camera image could come to the top of the cone. If so, the gradient will not be de□ned. This is described later. Then the side length of the cone image is rescaled up/down as follows.

$$cone\_len \ast= \sqrt[n]{\frac{||(grad\_x, grad\_y)||}{grad\_trgt}} \tag{3}$$

, where symbol $||(x, y)||$ represents the magnitude of (x, y). In Equation(3), $\sqrt[n]{}$ is applied in order to prevent the size of the cone image from jittering. In our system, n is set to 4. When the current magnitude is less than the target, that is the side length of the cone image appears longer than the width of the camera image, the side length of the cone image will be shorten, while it will be stretched when the current magnitude is greater than the target, that is the side length of the cone image appears less than the width of the camera image. Finally the side length of the cone image will appear almost same as the width of the camera image when the current magnitude becomes close to the target.

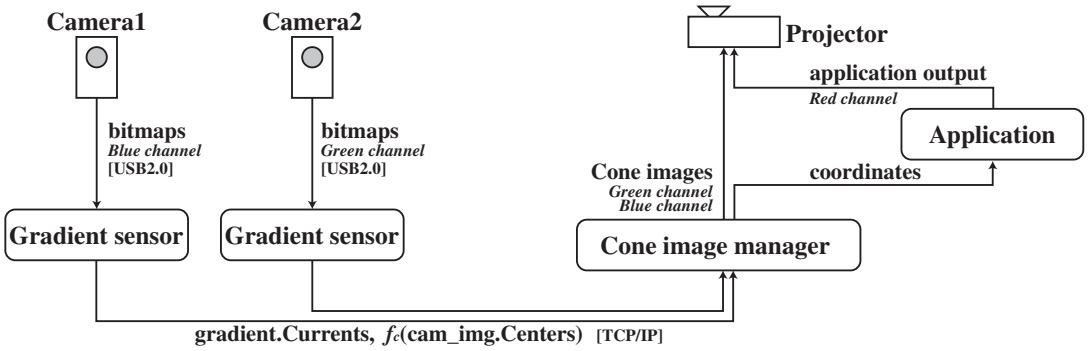

Figure 2. A schematic diagram of two camera-based pointers.

For the second objective, the direction of the gradient of color at the pixel in the center of the camera image is used. The cone image is moved as follows.

$$(cone\_x, cone\_y) \; -= a * \frac{(grad\_x, grad\_y)}{||(grad\_x, grad\_y)||} \tag{4}$$

, where variable a is a parameter of how much the cone image moves once and the variable also needs to depend on the size of the cone image to prevent it from jumping out of the camera image.

$$a = \frac{cone\_len}{b} \tag{5}$$

For the parameter a, a condition to stop moving the cone image is necessarily required in order to prevent it from jittering.

$$\begin{aligned} &if(f_c(cam\_x, cam\_y) \leq 255 * (b-1)/b) \\ &\quad allow\ moving; \\ &else\ \ stop\ moving; \end{aligned} \tag{6}$$

In our system, b is set to 4.

As mentioned above, the center of the camera image could come to the top of the cone. In that case, the gradient is invalid. To avoid using invalid values, Equation(6) is extended to

$$\begin{aligned} &if(f_c(cam\_x, cam\_y) \leq 255 * (b-1)/b) \\ &\quad do\ \ iterations; \\ &else\ \ skip\ the\ iteration;. \end{aligned} \tag{7}$$

3. System design

Our system is of two camera-based pointers. The system includes a projector, two web cameras, and three computers. The projector is a Hewlett-Packard Digital Projector HP mp2210 (1024x768 in resolution) and the web cameras are Logicool Qcam QVX-13Ns (320x240 in resolution, 30fps in max). The computers have 3GHz processors and they are also equipped with NVIDIA GeForce 7900 GSs. All the computers are networked by 100Mbps Ethernet with TCP/IP protocol. The cameras are hooked up to two of the computers with USB2.0 and the projector is connected to the third computer.

Figure 2 shows the schematic diagram of the system. It includes three software modules that run on the three computers. The software modules are gradient sensor, cone image manager, and application. They are summarized below.

3.1 A projector

The projector projects two cone images to a screen. These cone images have been resized and moved due to the signals sent from the cameras as mentioned in Section **2**. The projector also projects the output of an application in our system. To register those three separately on the same screen, different color channels are adopted. The blue and the green channels are used for the two cone images and the red channel is used for the output of the application.

3.2 Cameras

A camera captures a cone image on the screen in order to report back. A user holds the camera and points it at a spot on the screen. For the iterations of the deepest descent method with visual feedback, as mentioned in Section **2**, the cone image gradually moves in the center of the camera and then the camera performs a laser pointer. A laser pointer allows a user to point on the screen in the distance so he/she should be allowed to do it by the camera as well. This however causes the camera to be out-of-focused easily. The cone image is scaled in a linear gradient, so that it is robust for blurring. The almost identical cone image will be sent to the next module even if the camera is out of focused.

3.3 Gradient sensors

A gradient sensor receives an image from the camera and calculates the gradient at the pixel in the center of the im-

age, and then sends it to the next module. The gradient sensor also sends the value of the pixel to the next module. For the gradient in x direction, the gradient sensor uses simple equations to obtain it.

$$\begin{aligned}
&\partial f_c(cam_x, cam_y)/\partial x \\
&\quad = \begin{cases} (z_{+d} - z_0)/d & if\ |z_{+d} - z_0| \geq |z_0 - z_{-d}| \\ (z_0 - z_{-d})/d & else \end{cases} \\
&\quad z_0 = f_c(cam_x, cam_y) \\
&\quad z_{\pm d} = f_c(cam_x \pm d, cam_y)
\end{aligned} \tag{8}$$

It holds similarly for the gradient in y direction. In our system, d is set to 20 pixels.

3.4 A cone image manager

A cone image manager receives data of the gradients and the values from the gradient sensors, and changes the size of the cone images and the positions of them as mentioned in Section **2**. Then the cone image manager sends the positions of the cone images to an application as the coordinates that represent spots on the screen. The cone images are also projected again to the screen for the next feedback.

A sequence of the positions of the cone image tracks the spot on the screen, where the camera has been pointed at. In practical, the track could be a series of zigzags due to pixel noise, hand jitters, etc. To reduce the effect of zigzags, a concept of the tracking menus of Fitzmaurice et al.[17] works effectively. In our system, a circular cursor is used. A circular cursor is a circle whose center is sent to the application as the coordinates. The center of the cone image zigzags within the circle. When the center of the cone image goes out of the circle, the circle will move directly toward the center of the cone image so that the circle catches the center of the cone image.

3.5 An application

An application receives the coordinates from the cone image manager and also sends the output to the projector. It is projected to the screen along with the cone images by the projector but they appear in different color channels on the screen as mentioned in **3.1**.

4. Performance

All pieces of equipment in our system have intrinsic processing delays and the network between them causes communication delays. They are accumulated over a period of the feedback process. The total of the delays decides the interval of the feedback. The interval was experimentally measured approximately 50 milliseconds in the system. It equals 20 fps. The longest delay was caused by the camera. It takes time to refresh images on the internal memory. It is possible to be up to 30 fps but the camera will send back the same image until the refreshment of images is completed. Thereby we put some delays to the feedback process on purpose.

Our system uses three color channels, red, green, and blue, to register three signals separately on the same screen. It is possible to make the cone images invisible to users by both of infrared techniques and polarization. Even if so, up to 2 cameras are allowed to be hooked up. For any systems with multiple cameras of more than 2, we need to take another approach to this.

5. Strengths and weaknesses

Our system does not require focused cameras and homographies so users just put a projector in front of a screen and hold cameras then the system is ready to start. It does not take time to setup the system. A user can point at a spot on the screen just by manipulating his/her camera and he/she is also allowed to move left and right, and back and forth as he/she usually does with a laser pointer.

The visual feedback with a cone image is, however, required to locate the spot where a user has pointed. The region where he/she can point is limited within the area illuminated by the projector. As we use a laser pointer, we can point at a spot on the screen as well as the nearby wall. It is convenient and natural to interact with both of them in the same manner in terms of seamless interaction. In our system, he/she needs to use his/her camera and laser pointer to interact with both the screen and the wall.

As a simple application of our system, it allows a user to manipulate a cursor on a screen in the distance, in principle wherever the screen can be seen. This kind of application is presented in the next section. For more complex one, our system can be applied to a remote pointer. A user can point at a remote object directly through the screen. In general, to build that pointer, complex homographies are required. Our system, however, makes it convenient to build that pointer because our system works without any homographies.

6. Ping pong

A camera-based pointer needs several iterations of visual feedback to locate the spot where a user has pointed by a camera. This section presents Ping pong that works with two camera-based pointers in order to show that a camera-based pointer technique is practical. Ping pong requires two camera-based pointers to work correctly in order to make the rackets function.

Figure 3 shows the setups of camera-based pointers for Ping pong. A projector is placed on the desk at the bottom and a screen can be seen in the background, and two cameras are put on the table in the center. The projector and two cameras are hooked up to each of three computers to the left side. Three computers are not visible in this figure The positions of the screen and the cameras are neither registered nor tracked, and any fiducial points to indicate the

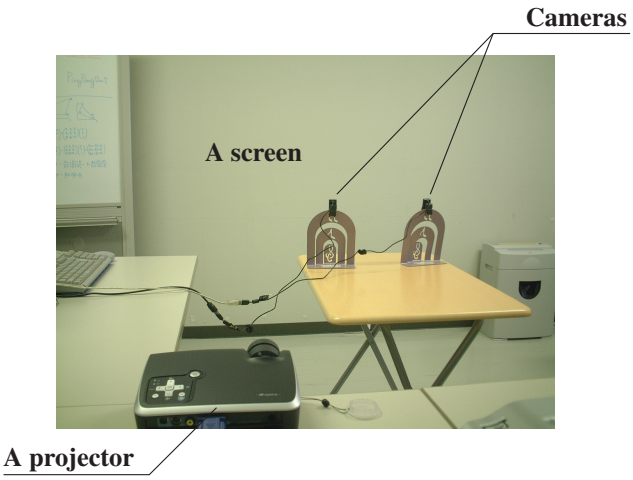

Figure 3. Setups of camera-based pointers for Ping pong. A projector is placed on the desk at the bottom and a screen can be seen in the background. Two cameras are put on the table in the center. The projector and two cameras are hooked up to each of three computers to the left side. Three computers are not visible in the figure.

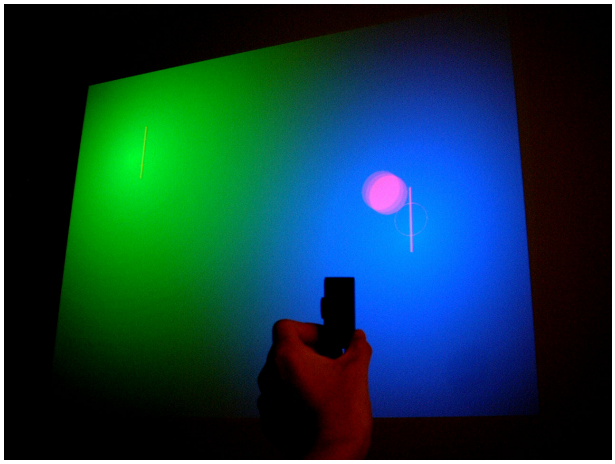

Figure 4. A snap shot of the screen as two players compete against each other in Ping pong that works with two camera-based pointers. They manipulate their cameras to control their own rackets represented by vertically long bars. The racket tracks the center of the cone image. One is on the left half and the other is on the right half. A solid red circle represents a ping-pong ball. In this scene, the right player just hits the ping-pong ball back to the left side.

screen are not stuck on the wall. At this point, any homographies are not used. It is time to enjoy Ping pong.

Figure 4 is a snap shot of Ping pong while two players compete. The field is projected on the wall. There are two cone images on the field. One is the green cone image and the other is the blue cone image. The green cone image works for the left player and the blue one does for the right player. The left player is not visible in this snap shot while the left hand of the right player holding a camera can be seen at the bottom. In the center of the cone image, a vertically long bar can be seen. It represents a racket. The racket tracks the center of the cone image so the player moves his/her racket by manipulating his/her camera. A solid circle represents a ping-pong ball. In this scene, the right player just hits the ping-pong ball back.

The green cone image is a gray-scaled cone image in the green channel and the blue one is another gray-scaled cone image in the blue channel. The camera captures images in the specified channel so that the iterations of visual feedback proceed. The rackets and the ball are drawn on the same field in the red channel. They are registered separately on the screen, so they are independent of each other as they overlap.

7. Conclusion

We introduced a camera-based pointer. A camera-based pointer is a pointer system where a user uses a camera to point at a spot on a screen instead of a laser pointer. In the system, the spot on a screen is located by the deepest descent method with visual feedback. A cone image is projected on the screen and then it is reported back by the camera. This feedback moves the cone image towards the spot where a user has pointed by a camera so that the system will catch it in several iterations. Finally we demonstrated Ping pong that worked with two camera-based pointers in order to show that a camera-based pointer was practical.

For laser pointer interaction, it is a common way that the spot on the screen, which has been pointed by a laser pointer, is captured by a precisely focused camera and then the pixel on the camera image is mapped into the corresponding point in the screen coordinates system by the homographies between the camera and the screen coordinates systems. This study took an approach to a system that worked without focused cameras and homographies. The deepest descent method with a cone image as the feedback was employed to compensate for out-of-focused cameras and no homographies.

In future work, we will look into the ways to interact with both projector screens and everyday walls seamlessly by a camera-based pointer.

Acknowledgement

This work is partly supported by Computer Science Laboratory(CSL) of Fukuoka Institute of Technology(FIT).

References

[1] Ben Shneiderman and Catherine Plaisant. *Design the user interface: strategies for effective human-*

computer interaction –4th ed. Addison-Wesley Publishing Company, Boston, 2005.

[2] Tobias Höllerer and John Pavlik. Situated documentaries: embedding multimedia presentations in the real world. In *Proc. ISWC 1999*, pages 79–86, October 1999.

[3] Tobias Höllerer, Drexel Hallaway, Navdeep Tinna, and Steven Feiner. Steps toward accommodating variable position tracking accuracy in a mobile augmented reality system. In *Proc. AIMS 2001*, pages 31–37, August 2001.

[4] Jun Rekimoto. Navicam: a palmtop device approach to augmented reality. *Fundamentals of wearable computers and augmented reality*, pages 353–377, 2001.

[5] David Drascic and Paul Milgram. Perceptual issues in augmented reality. In *Proc. SPIE*, pages 123–134, February 1996.

[6] Morten Fjeld, Kristina Lauche, Martin Bichsel, Fred Voorhorst, Helmut Krueger, and Matthias Rauterberg. Physical and virtual tools: activity theory applied to the design of groupware. *A special issue of computer supported cooperative work: activity theory and the practice of design*, 11:153–180, 2002.

[7] Paul Beardsley, Jeroen Van Baar, Ramesh Raskar, and Clifton Forlines. Interaction using a handheld projector. *IEEE Computer graphics and applications*, 25(1):39–43, January/February 2005.

[8] Oliver Blimber, Gordon Wetzstein, Andreas Emmerling, and Christian Nitschke. Enabling view-dependent stereoscopic projection in real environments. In *Proc. ISMAR2005*, pages 14–23, October 2005.

[9] Grant Wallace, Han Chen, and Kai Li. Automatic alignment of tiled displays for a desktop environment. *Journal of Software*, 15(12):1776–1786, December 2004.

[10] James M. Rehg, Matthew Flagg, Tat-Jen Cham, Rahul Sukthankar, and Gita Sukthankar. Projected light displays using visual feedback. In *Proc. ICARCV2002*, pages 926–932, December 2002.

[11] Ramesh Raskar and Paul Beardsley. A self-correcting projector. In *Proc. CVPR2001*, pages 504–508. IEEE Computer Society, December 2001.

[12] Mark Ashdown and Rahul Sukthankar. Robust calibration of camera-projector system for multi-planar displays. In *Technical Report HP Labs HPL-2003-24*, January 2003.

[13] Takayuki Okatani and Koichiro Deguchi. Autocalibration of a projector-screen-camera system: theory and algorithm for screen-to-camera homography estimation. In *Proc. ICCV2003*, page 774. IEEE Computer Society, October 2003.

[14] Jr. Dan R. Olsen and Travis Nielsen. Laser pointer interaction. In *Proc. CHI2001*, pages 17–22. ACM, 2001.

[15] Sergey V. Matveyev and Martin Göbel. The optical tweezers: multiple-point interaction technique. In *Proc. VRST2003*, pages 184–187. ACM, October 2003.

[16] Ji-Young Oh and Wolfgang Stuerzlinger. Laser pointers as collaborative pointing devices. In *Proc. Graphics Interface*, pages 141–150, May 2002.

[17] George Fitzmaurice, Azam Khan, Robert Pieké, Bill Buxton, and Gordon Kurtenbach. Tracking menues. In *Proc. UIST*, pages 71–79. ACM, November 2003.