Active Shadows: Real-Time Video Segmentation within a Camera-Display Space

I-Jong Lin { ijonglin@hpl.hp.com }
Hewlett Packard Laboratories, Palo Alto, CA, USA

February 12, 2004

This paper introduces the Active Shadows algorithm that produces low latency (under 200ms) and high quality video object segmentation of a person, occluding a computer-controlled display of any type (LCD, projector, CRT, plasma) in front of video camera device. This algorithm sets up a causal video feedback loop that can resolve ambiguous visual occlusions by adaptively modifying the displayed image in real-time. These real-time modifications to the display manifest themselves as if the camera were a virtual light source and was casting a reverse shadow onto the display. Active Shadows gives the same output as a chromakey system except that the user is physically interacting with the displayed image, instead of a colored background. With this setup, the system produces segmented video at approximately 5 fps and seamlessly composites presentation slides and segmented video of the speaker to create a multi-layered video representations.

1 Introduction

Display technologies (LCDs, OLEDs, Plasma Screens, and digital projectors) beyond cathode ray tubes (CRTs) are becoming a ubiquitous part of our living and working environments as they are driven to be less bulky with larger display areas at cheaper cost. When coupled with a camera, these display technologies become opportunities for viewers to interact with the displayed digital information.

In this paper, we will concentrate on a computer vision solution for interaction that segments out occluding objects in front of the display within a camera view, i.e. separates visual information of occluding objects from the background [5]. This special case of the video object segmentation problem [7] has four conditions: 1) it happens in real-time, 2) the area of interest is confined only to the front of the display, 3) the solution controls **both** display and camera, and 4) one person at a time is interacting with the display.

The Active Shadows algorithm has four major applications: 1) inexpensively adding interactive capabilities to wide range of display devices, 2) creating rich media composition, 3) enabling real-time interaction with video and, by extension, a collaborative rich media authoring space, and 4) the front-end feature extraction for a vision-based recognition system.

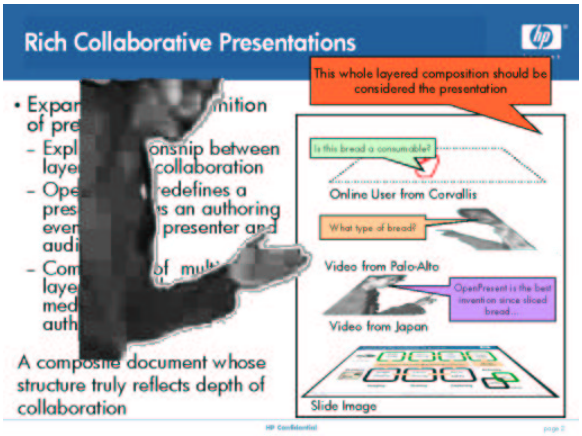

Figure 1: Active Shadows segmentation output overlaid upon a presentation slide image. The gray image represents a video layer of a person interacting with the projected slide image.

2 Setup

For this algorithm, there are two two system requirements, to be defined in this section: 1) the system must have *a causal video feedback loop* and 2) the system must know its *camera-display pixel correspondence.*

First, we define the *causal video feedback loop* as a special case of the video feedback loop [2]. To create a generic video feedback loop, we must set up a feedback circuit of visual information within our system. As shown in Figure 2, our system has three major elements: 1) the visual input device (a digital camera or webcam), 2) a visual output device (projector, plasma display, CRT display, etc.) and 3) a computer

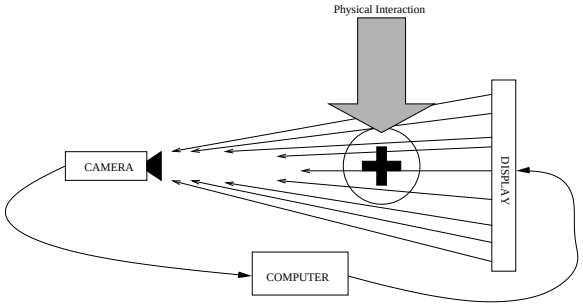

Figure 2: The basic video feedback loop: From computer to display, from display to camera, from camera to computer and then back again.

that drives this display and takes in the digital images from the camera. To connect the visual output to the visual input, we aim camera at the display.

We define a *causal video feedback loop* as a video feedback loop with guarantee that, after an image is drawn to the display, a computer can capture this drawn image. With this guarantee, we define an iteration of this causal video feedback loop as the following three step sequence: 1) display of image, 2) capture of displayed image, and 3) computer analysis of the capture which may determine the image of the display at the next iteration.

Once a causal video feedback loop is set up, we obtain *camera-display pixel correspondence*, i.e. the one-to-many mapping of a given camera pixel to the set of display pixels that are seen by the given camera pixel. In particular, we cite [1] as a robust technique of finding this correspondence. Given this correspondence and proper manipulation of the display, we can decompose the behavior of the system into an array of causal video feedback loops that interact, but have independent computation.

3 Active Shadows Algorithm

When the setup of the previous section is done, the Active Shadows algorithm implements a virtual shadow on the display, to be explained in more detail in this section; a well-defined subset of this shadow is a high-quality video object segmentation. If we suppose that the camera is a kind of virtual light source, then the Active Shadows algorithm overlays a shadow of a chosen color on top of the displayed image itself where the camera's view is blocked. The shadow helps to resolve visual ambiguity of segmentation, indicates which part of the display is occluded from the camera view, and gives visual feedback to the user as well. The system gives the same output as a chromakey system [3], except that the user is physically interacting with the displayed image, instead of a colored background.

The Active Shadows algorithm hybridizes two different techniques: *passive visual testing* where we have no control over the visual field, and *active visual testing* where we can change the color of an object in the visual field to any color.

We begin with an example of *passive visual testing*, a simple foreground and background separation, a simplification of [6], only using image differencing. If we know the camera pixel colors for an unoccluded static background via calibration and *apriori* knowledge of display image, we subtract the current camera image from the background. The positions in the resulting image difference of high magnitude are assumed to be occluded. Many errors in segmentation stem from this visual ambiguity (e.g. objects that are same color as display image) and the ability to fully model the background.

We can force the background to a reserved color by changing the display as a kind of *active testing*. Assuming an object has a different color than the reserved color, then the previous solution has no ambiguity. However, the display can only shows this reserved color.

Active Shadows complement passive visual testing with active visual testing, while maintaining display functionality. To do so, we split the colorspace of the display into two parts: one for the reserved color and the remainder of the colorspace in which to draw the displayed image. Although this split of the colorspace degrades the dynamic range of the display, Active Shadows can independently choose either active or passive testing for each camera pixel. Most of the time, we use the passive testing to look for occluded objects; however, when a camera pixel is suspected of occlusion, our camera-display setup uses active testing to double-check for occlusion. Only an object that can maintain ambiguity through the color change of the display will remain ambiguous.

To assist with the explanation of the Active Shadows algorithm, we define some basic variables:

Basic Variables	
$d(x_1, x_2)$	the distance in camera pixel between two camera pixels (x_1, x_2)
t	our time index which is the number of iterations in the causal video feedback loop as defined in section 2
$\vec{I}_t(x)$	color of the camera pixel x at time t
$C(\vec{x}, \vec{y})$	a boolean function that returns true when two camera pixel colors are visually equivalent

Variables from Calibration	
$\beta(x)$	the set of display pixels described in section 2 of camera-display pixel correspondence where x is a camera pixel
γ	$\{x \mid \beta(x) \neq \emptyset\}$, i.e. the set of camera points in the video feedback loop
$\vec{R}_c(x)$	colors of camera pixel x with the display showing only the reserved color
$\vec{R}_i(x)$	colors of a camera pixel x with the display showing only the displayed image
Algorithm Variables	
$S_t^{0\ldots2}(x)$	$x \in \gamma$, the state of the state machine associated with each pixel in γ, at time t. Superscript denotes steps between a single iteration
r_g	the shadow growth parameter as a distance in camera pixels
Output Variables	
P_t	the set of pixels on the display to be drawn with the reserved color (otherwise, the pixels are drawn to the display image; in terms of t, the effects of P_t are seen in camera image I_t)
V_t	the segmentation output of camera pixels, i.e. camera pixels determined to be occluded

With these variables, the Active Shadows algorithm associates a basic state machine with each pixel. The state transitions depend first on the incoming visual information and then the states of nearby pixels. The four states of the state machine that are coupled with each pixel are defined as follows: 1) Passive Testing **(PT)**, 2) Passive Suppressed **(PS)**, 3) Active Testing **(AT)**, and 4) Active Confirmed **(AC)**.

$$S_t^n(x) \in \{\mathbf{PT}, \mathbf{PS}, \mathbf{AT}, \mathbf{AC}\} \tag{1}$$

We define the initial conditions ($t = 0$) of the algorithm,

$$\begin{array}{c} \forall x \in \gamma, S_0^n(x) = \mathbf{PT} \\ P_0 = \emptyset \end{array} \tag{2}$$

For $t > 1$, we can define the computations as a series of state transitions pass over the set of camera pixels γ applied sequentially as shown in Figures 4, 3.

The first pass, ($S_t^0 \rightarrow S_t^1$) blends in passive and active testing by choosing the method depending on the state associated with the camera pixel. Passive and active visual testing is, respectively, defined as:

$$\begin{array}{l} p_t(x) = C\left(\vec{I}_t(x), \vec{R}_i(x)\right) \\ a_t(x) = C\left(\vec{I}_t(x), \vec{R}_c(x)\right) \end{array} \tag{3}$$

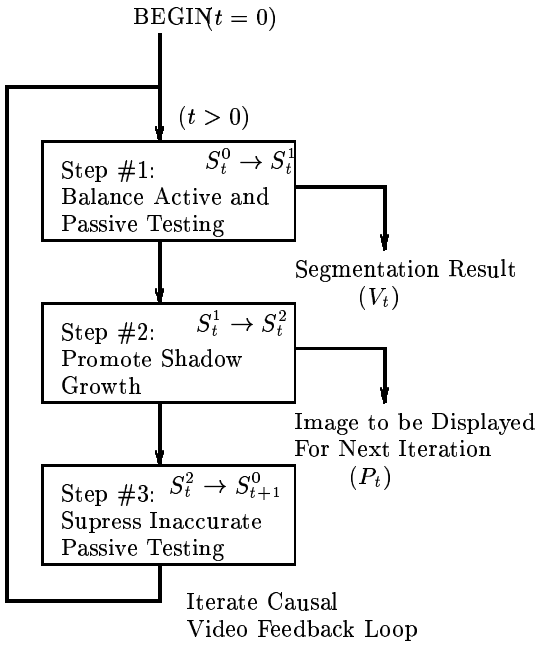

Figure 3: Algorithmic Steps For Active Shadows for each causal video feedback loop iteration

When the passive testing indicates occlusion, we move the pixel from **PT** to **AT** state, driving the display to a reserved color. On the next iteration, we double-check the pixel color. If the pixel x is found to be unoccluded by this active testing, then it moves back to **PT** state, drawing the display image to the display area $\beta(x)$. However, if it is found to be occluded with active visual testing in the next iteration, then it moves into a **AC** state and is considered to be occluded with a high degree of certainty. Thus, the output segmentation is defined as:

$$V_t = \left\{x \in \gamma | S_t^1(x) = \mathbf{AC}\right\} \tag{4}$$

In state **AC**, the pixel is actively tested until the occlusion is removed.

The second pass of state transition ($S_t^1 \rightarrow S_t^2$) encodes a locality heuristic:

$$g_t(x) = \begin{cases} 1 & \exists y \in V_t | (d(x,y) \leq r_g) \\ 0 & otherwise \end{cases} \tag{5}$$

i.e. if a camera pixel x is close to an occluded pixel y ($\in V_t$) as defined by the function $d(x,y)$, then the pixel x is probably being occluded. Camera pixels that pass this heuristic and were passively testing (**PT**) for occlusion or suppressed (**PS**) move into the active testing state (**AT**). In Eq. 5, the parameter r_g is inversely proportional to the speed the shadow grows across an region that is ambiguous to passive testing. The drawback of this heuristic is that there will always be a set of unoccluded camera pixels that

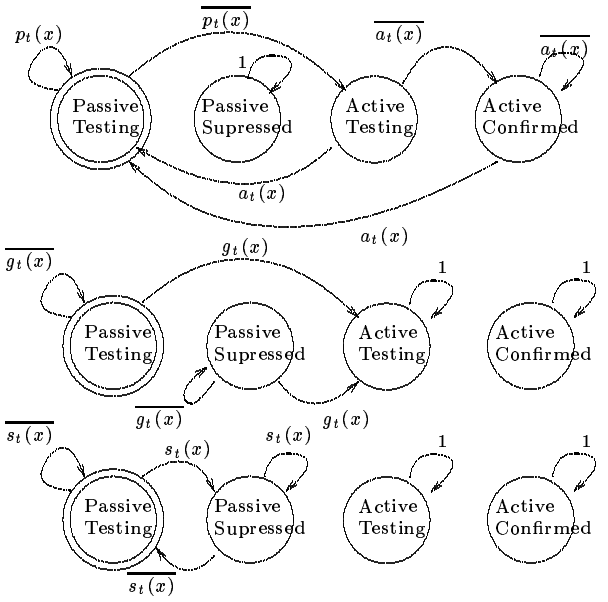

Figure 4: Three passes of State transitions in order from top to bottom, respectively. Transition variables are defined in Eq. 3, 5, 7.

will be drawn to white, even though they are not occluded. The setting r_g is directly proportional to the thickness of "halo" of display pixels, that are drawn to a reserved color, but are unoccluded.

After the second state transition pass, we determine the set of pixels are doing active visual testing, and drive the display accordingly:

$$\begin{aligned} \omega_t &= \{x \in \gamma | S_t^2(x) \in \{\mathbf{AC}, \mathbf{AT}\}\} \\ P_{(t+1)} &= \bigcup_{x \in \omega_t} (\beta(x)) \end{aligned} \tag{6}$$

The third state transition pass ($S_t^2 \rightarrow S_{t+1}^0$) ensures that active visual testing does not interfere with passive testing. If not properly suppressed, uncontrollable oscillations will occur in the video feedback loop.

$$s_t(x) = \begin{cases} 1, & if\,(P_{t+1} \cap \beta(x)) \neq \emptyset \\ 0, & otherwise \end{cases} \tag{7}$$

We move these affected pixels from the **PT** state into **PS** state to turn off passive testing, and move them back when the interference disappears.

After three passes of state transitions, the next iteration of the Active Shadow algorithm can occur. If we assume the pixel comparison operation C, d distance calculation, and state transitions are $O(1)$ operation, then one iteration of the Active Shadows algorithm is $O(n)$ where n is merely $\|\gamma\|$.

4 Results/Conclusion

With Active Shadows producing high-quality segmentation in real time, we implemented two major applications: 1) a virtual touchscreen and 2) real-time rich media capture of presentations. Active Shadows runs on a Compaq 800w notebook with Hewlett Packard XP8000 projector and an inexpensive webcam with Linux OS.

By binding the output segmentation (V_t) with an algorithm to discover the pointing [4], we can implement a robust virtual touch screen algorithm that also accepts pointing from a laser pointer. We convert these pointing events into mouse events for an X server into mouse events, enabling, for instance, web-surfing by directly touching links in Netscape application displayed on a projector.

As shown in Figure 1, the second application uses the output segmentation with correspondence information from the camera to the display to project the video of the physical person on top of the digital image. The physical correspondences (where I touch on the screen) are still maintained in the digital domain (the video of where I touch of the screen is overlaid directly on top of that area). The system layers a speaker's audio and physical presence in front of a display on top of a presentation image to create a recorded rich media.

Active Shadows is an expandable, powerful algorithm that can be amended with extra states and state transitions to include even more functionality.

References

[1] Nelson L. Chang. Efficient dense correspondences using temporally encoded light patterns. In *Proceedings of IEEE International Workshop on Projector-Camera Systems (PROCAMS)*, October 2003.

[2] J.P. Crutchfield. Space-time dynamics in video feedback. *Physica D*, 10D(1-2), January 1983.

[3] S. Gibbs, C. Arapis, C. Breiteneder, V. Lalioti, S. Mostafawy, and J. Speier. Virtual studios: An overview. *IEEE Multimedia*, 5(1), March 1998.

[4] I-J. Lin. Patent no. 6,542,087: System and method for extracting a point of interest ..., April 2003.

[5] I-J. Lin and S.Y. Kung. *Video Object Extraction and Representation: Theory and Applications.* Kluwer Academic Publishers, 2000.

[6] A. Neri, S. Colonnese, G. Russo, and P. Talone. Automatic moving object and background separation. *Signal Processing*, 2(66):219–232, 1998.

[7] ISO/IEC JTC1/SC29/WG11 Coding of Moving Pictures and Associated Audio MPEG98/W2194. Mpeg-4 requirements doc., March 1998.