

Configurable Hotspots for Ubiquitous Interaction

Alejandro Jaimes

FXPAL Japan, Fuji Xerox Co. Ltd.
430 Sakai 2nd Floor, Nakai-machi
Ashigarakami-gun, Kanagawa 259-0157, Japan
alex.jaimes@fujixerox.co.jp

Jianyi Liu

FXPAL Japan, Fuji Xerox Co. Ltd.
430 Sakai 2nd Floor, Nakai-machi
Ashigarakami-gun, Kanagawa 259-0157, Japan
liu.jianyi@fujixerox.co.jp

ABSTRACT

In this paper we describe our configurable hotspot framework for ubiquitous interaction. A camera points to a physical space, the user defines interaction areas, and designs gestures by combining hotspots (rectangles) that detect simple 2D motions (left_right, top_bottom, etc.).

Keywords

Computer vision, learning, skin detection.

1 Introduction

Gesture-based interaction is attractive because it is natural in human-human communication. Furthermore, lower hardware costs and higher computational power make it promising for ubiquitous interaction. However, there are some difficulties in creating effective gesture-based frameworks, particularly those that have the following desired qualities: (1) flexibility to allow the user to create his own gestures; (2) robustness; and (3) work in real time with inexpensive hardware.

Our proposed approach addresses these three issues using simple and easily re-configurable 2D hotspot components: a gesture is recognized as a sequence of hotspot area activations. This yields effective and robust results because the gestures that each individual hotspot recognizes are very specific.

In our framework, a camera points to a physical area and the video captured by the camera appears in a nearby monitor. For example, the camera can be placed on top of the computer monitor so the video captured by the camera appears in a resizable window on the screen (in Figure 1, left, the capture area is behind the keyboard).

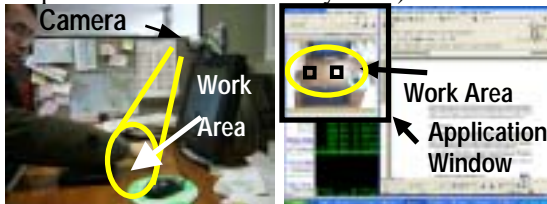


Figure 1. The camera points at an area defined by the user. The application window (right), which shows the captured video and hotspots can be resized so it does not interfere with the applications being used.

A hotspot is defined as a 2D area in the captured video image that is activated when the user moves his hands or fingers *across* the area in a particular direction. In

the application window (Figure 1, right) the user can draw new hotspot rectangles and assign commands either to individual hotspots or combinations. Figure 2 (right) shows a sequence of hotspots combined to represent a single action. Such configurations can be saved, so the user could have a library of configurations for different applications, appliances, or interaction spaces if desired. In an apartment, the camera could point to the wall above the couch, for example, and be used to control lighting, the TV (e.g., waving a hand from left to right to switch channels), or other appliances (the image captured could be projected or shown in a nearby monitor).



Figure 2. Image captured by the camera. On the right image a composition of hotspots represents a single action. The same templates could be used with the camera pointed at a wall, or any other area for a different application.

Our system has the following advantages: (1) individual components can be very robust and easily combined for different purposes (users can create their own gestures); (2) users do not need to learn complex gestures (simple 2D motion gestures suffice); (3) users do not need to remember gestures (hotspots are visible on the screen); and (4) the framework works in real time with an inexpensive web camera.

1.1. Related Work

Physical devices [4], and objects [1][7] have been used for interaction. In [1], for example, interface objects are learned for programming. In contrast, we focus on the final user and on simple hand gestures (not objects). The system in [5] uses skin and face detection for video indexing, and the framework of [12] uses modular “Visual Interface Cues” and Hidden Markov Models to learn gesture dynamics. Our approach does not require training (except for the skin filter), and allows the user to combine simple hotspot components for gesture recognition. Most gesture recognition approaches [9][10] image the entire hand (or hands) or use tracking [8]. The authors of [1] define three types of widgets: button, linear, and circular. We use only one camera, and interaction is based on 2D *configurations* of simple, directional hotspot components.

2 Hotspot Activation

Hotspots are defined by four coordinate points ($x1, y1, x2, y2$), and an activation constraint. A hotspot is activated when a skin area enters the hotspot and exits it in one of four directions: *left_to_right*, *right_to_left*, *top_to_bottom*, and *bottom_to_top*. Each hotspot has only one constraint, which makes it very robust. For example, a *left_right* hotspot activates *only* if a skin area enters it from the left and exits on the right. The hotspot activation process is depicted in Figure 3 (further details in [6]).

- **Skin filter construction:** we construct a skin filter using machine learning. First, we obtain a set of skin images and a set of images that do not contain skin. We convert each image to HSV color space (because it is perceptually uniform: points that are close in HSV space are also perceptually similar). This yields a training set of skin and non-skin pixels. We use a machine learning algorithm from Weka [11] to build a skin classifier. Since different skin colors map to the same area in HSV space the approach is independent of race (different skin colors map the same HSV area).

- **Skin detection:** only the area corresponding to the hotspot is processed every t milliseconds. In order to make processing more efficient, we examine only a subset of the pixels inside the hotspot rectangle: a pixel every d pixels is classified into skin or not-skin using the skin filter. Then we compute the center of mass of the skin pixels within the hotspot area (black spots in Figure 3).

- **Hotspot activation:** activation of a hotspot starts with the first frame in which skin pixels are found (hand enters hotspot) and ends when no skin pixels are found (hand leaves hotspot). This yields a sequence of center of mass points. We use the first n points in the sequence to compute a motion start point, and the last m points in the sequence to compute a motion end point. Using the two points we compute a motion vector that estimates the direction of motion. The hotspot is activated only if the vector's direction corresponds to the hotspot's constraint.

- **Gesture recognition:** a gesture is detected when all of its corresponding hotspots are activated in a particular sequence (e.g., Figure 2, right).

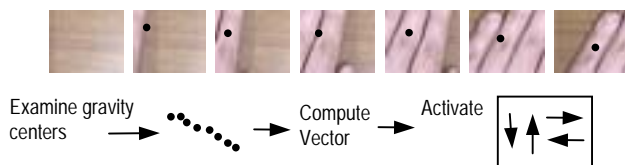


Figure 3. Hotspot activation starts when a skin area is detected (frame 2) and ends when no skin is present. In each sample, skin is detected and the center of mass is computed.

3 Discussion & Future Work

Our framework has two advantages: (1) combinations of simple components can yield higher accuracy than what

may be achieved with gesture recognition frameworks that require segmentation and tracking, and (2) high user customization—several hotspot configurations can be saved for different applications. Since the user views the image being captured by the camera, remembering specific gestures for particular applications is not necessary.

The approach is suitable for simple 2D gestures that use non-overlapping hotspots and where limb segmentation is not necessary (e.g., determine which hand was used, which part of the arm, etc.). It is not appropriate for complex 3D gestures (e.g., rotating hands, or in which there is occlusion), or for text input (e.g., for e-mail, word processing, etc.). On the other hand, the technique is independent of the area the camera points to, so it is scalable (not limited to desktop use) and useful in any scenario in which spatial 2D commands are desired. Due to the simplicity of the hotspots, the technique can be easily implemented for camera-equipped mobile devices.

Future work includes exploring usability aspects in actual applications (e.g., multi-screen; multi-document viewing applications), investigating ways to automatically determine hotspot area sizes, integrating audio, using in remote collaboration [3] and in mobile devices.

4 References

- [1] J. Alan, D.R. Olsen, "Light Widgets: Interacting in Everyday Spaces," in *proc. ACM 7th Intl. Conf. on Intelligent User Interfaces 2002*, San Francisco, CA, Jan. 2002.
- [2] J.A. Fails, D.R. Olsen, "A Design Tool for Camera-based Interaction," *CHI: ACM Conf. on Human Factors in Computing Systems, CHI Letters 5(1)*, pp. 449 – 56, 2003.
- [3] J. Foote, Q. Liu, D. Kimber, P. Chiu, and F. Zhao, "Reach-Through-the-Screen: A New Metaphor for Remote Collaboration," in *ACM/IEEE PCM 2004*, Dec. 2004, Tokyo, Japan.
- [4] S. Greenberg and C. Fitchett, "Phidgets: easy development of physical interfaces through physical widgets," *UIST: ACM Symp. on User Inter. Soft. and Tech., CHI Letters 3(2)*, pp. 209 –18, 2001.
- [5] A. Jaimes, Q. Wang, N. Kato, H. Ikeda, and J. Miyazaki, "Visual Trigger Templates for Knowledge-Based Indexing," in *ACM/IEEE PCM 2004*, Dec. 2004, Tokyo, Japan.
- [6] A. Jaimes and J. Liu, "Hotspot Components for Gesture-Based Interaction," *Interact 2005*, Sept. 2005, Rome, Italy.
- [7] S.R. Klemmer J.Li, J. Lin, and J.A. Landay, "Papier-Mâché: Toolkit Support for Tangible Input," *CHI: ACM Conf. on Human Factors in Comp. Systems, CHI Letters 6(1)*, 2004.
- [8] H. Koike, C. Xinle, Y. Nakanishi, K. Oka, Y. Sato, "Two-Handed Drawing On Augmented Desk," *ACMCHI 02*, 2002.
- [9] S. Marcel, "Gestures for multi-modal interfaces: A Review," Technical Report IDIAP-RR 02-34, 2002.
- [10] V.I. Pavlovic, R. Sharma and T.S. Huang, "Visual interpretation of hand gestures for human-computer interaction: a review", *IEEE PAMI*, 19(7):677-695, 1997.
- [11] I.H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999.
- [12] G. Ye, J.J. Corso, D. Burschka, and G.D. Hager, "Vics: A modular hci framework using spatio-temporal dynamics," *Machine Vision and Applications*, 16(1):13-20, 2004.