GESTURE VIEWPORT: INTERACTING WITH MEDIA CONTENT USING FINGER GESTURES ON ANY SURFACE

Author(s) Names(s)

Author Affiliation(s)
abc@xyz.com

ABSTRACT

In this paper, we describe *Gesture Viewport*, a projector-camera system that enables finger gesture interactions with media content on any surface. We propose a novel and computationally very efficient finger localization method based on the detection of occlusion patterns inside a virtual sensor grid rendered in a layer on top of a viewport widget. We develop several robust interaction techniques to prevent unintentional gestures to occur, to provide visual feedback to a user, and to minimize the interference of the sensor grid with the media content. We show the effectiveness of the system through three scenarios: viewing photos, navigating Google Maps, and controlling Google Street View.

***Index Terms*—** Projector-camera system, finger gesture interaction, viewport

1. INTRODUCTION

A viewport is a graphical user interface (GUI) widget by which a user may interact with media content such as panning, zooming, or rotating an image. On a touch display, a user may perform the interactions using finger gestures. It is highly desirable that a user may also perform the interactions using finger gestures on displays that lack input support, such as a non-touch display or projector screen.

Projector-camera systems can potentially turn any surface into an interactive workspace. Most existing projector-camera systems that support finger gesture control for various purposes exploit skin colors or depth images to track finger movements, which tends to be not-so-robust and/or computationally expensive [1][2][3]. To address these problems, we propose a novel and computationally very efficient finger localization method based on the detection of occlusion patterns inside a virtual sensor grid rendered in a layer on top of a viewport widget. We have developed *Gesture Viewport*, a projector-camera system that uses this method to enable finger gesture interactions with media content on any surface. We demonstrate this system with three scenarios for viewing photos, navigating Google Maps, and controlling Google Street View.

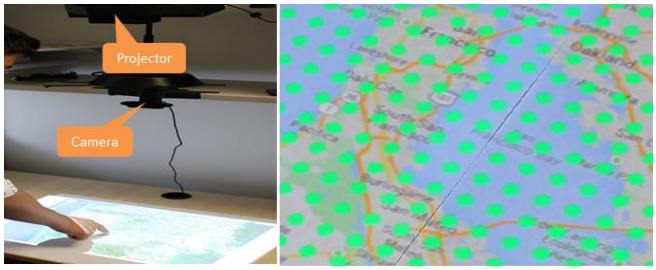

Figure 1: (a) Typical hardware setup of Gesture Viewport consisting of an off-the-shelf projector and camera. (b) Close-up camera view of a sensor grid rendered in a layer on top of the viewport.

2. SYSTEM AND APPROACH

Gesture Viewport projects a standard viewport widget onto a surface with a projector and uses a camera to detect occlusion patterns inside a sensor grid rendered in a layer on top of the viewport. A typical hardware setup of the system is shown in Figure 1(a). A close-up camera view of the sensor grid overlaying the media content in the viewport is illustrated in Figure 1(b). We explain several important concepts as follows:

A *sensor blob* is a blob of pixels with a distinct color which is aware of its location in both viewport and camera coordinate systems. The camera detects the change of color that occurs over the center of a sensor blob and triggers an *occlusion event* at the sensor blob's location.

A *sensor grid* is a grid of evenly spaced sensor blobs that cover the entire viewport. Each sensor blob inside the sensor grid functions independently, and all the location-bearing occlusion events are aggregated and analyzed.

Occlusion patterns (OPs) are specifically designed patterns of sensor blob locations where occlusion events occurred. Once detected, an OP represents the presence of a finger at an anchor location (e.g. the center) of the pattern. In our system, an OP is defined as a 3-by-3 block of sensor blobs where the center sensor blob has triggered an occlusion event and at the same time at most two other adjacent sensor blobs have also triggered occlusion events. Figure 2 illustrates several examples of OPs. Multiple OPs may be detected in the sensor grid simultaneously, which

implies that multiple locations of finger presence may be detected simultaneously. This enables the multi-finger and multi-user interaction capability of the system.

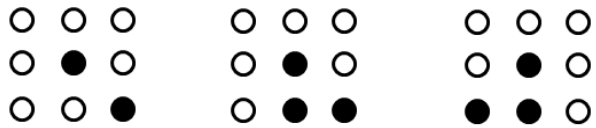

Figure 2: Examples of OPs. A solid circle represents a sensor blob location where an occlusion event has occurred.

Finger gesture interactions in Gesture Viewport consist of panning, zooming, and rotating the media content in the viewport via finger movements. Our system supports both one-hand and two-hand interactions. The analysis of OPs over time triggers a sequence of finger movement events, allowing the viewport to implement the panning, zooming, or rotating operation. We develop several robust interaction techniques to prevent unintentional gestures to occur, to provide visual feedback to a user, and to minimize the interference of the sensor grid with the media content.

The technique to prevent unintentional gestures to occur is described as follows: We introduce the *press* event and *release* event. In general, these events are unlikely to be triggered by accident. A press event occurs if the continuous presence of a finger at the same location lasts for a certain period of time (e.g. 200ms). A press event must occur before any of the panning, zooming, and rotating operations. A release event occurs if there is a press event occurring for a finger, and one of the following two conditions are met: 1) The continuous presence of this finger lasts for a certain time period (e.g. 200ms) at the same location; 2) The continuous non-presence of this finger lasts for a certain time period (e.g. 200ms). A release event marks the end of any of the panning, zooming, and rotating operations. Since a panning, zooming, or rotating operation must begin with a press event and end with a release event, unintentional gestures that could cause disorder of the viewport's content are prevented.

The technique that minimizes the interference of the sensor grid with the media content is described as follows: Before a press event occurs, all the sensor blobs in the sensor grid are visible, providing maximum coverage of the viewport. Upon a press event, only those sensor blobs that fall within a certain distance from the press event location are visible, with all remaining sensor blobs becoming invisible. That is, the sensor grid turns into a sensor dish centered around the finger location. See Figure 3(3) for an example. This minimizes the interference of the sensor grid with the viewport's content during gesture interactions. The disk of visible sensor blobs also serves as a visual guide and feedback for a user to move his or her finger. For example, the radius of the disk determines how far and fast a user may move his or her finger. Upon a release event, all the sensor blobs in the sensor grid become visible again, providing maximum coverage of the viewport. See Figure 3(2).

To further minimize the interference by the sensor grid, we only make the sensor grid visible when a user is actually performing gesture interactions. A user can turn on and off the sensor grid at any time using a widget (e.g. a button) external to the viewport. See Figure 3(2).

Figure 3 illustrates several steps of the process of gesture interactions in Gesture Viewport in a realistic scenario.

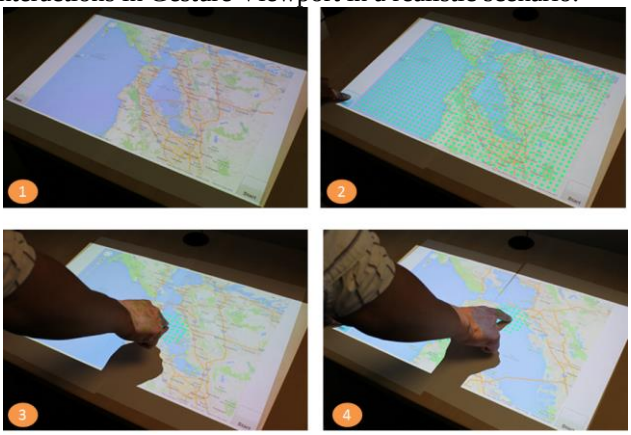

Figure 3: Several steps of the process of gesture interactions in Gesture Viewport. (1). The sensor grid is invisible, and no gesture interaction is allowed; (2). A user turns on the sensor grid using a button external to the viewport; (3). A user triggers a press event by aiming his finger at a sensor blob location for 200ms, and the sensor grid shrinks into a disk; (4). The user moves his finger to perform a pan operation on Google Maps.

3. APPLICATIONS

We show the effectiveness of the Gesture Viewport system through three application scenarios: viewing photos, navigating Google Maps, and controlling Google Street View. A demo video clip is attached for review. Live demos will be set up and shown on site.

4. REFERENCES

[1] Kane, S.K., D. Avrahami, J.O. Wobbrock, B. Harrison, A.D. Rea, M. Philipose, and A. LaMarca. Bonfire: a nomadic system for hybrid laptop-tabletop interaction. *Proc. of UIST '09*, pp. 129-138.

[2] Kjeldsen, R., C., Pingali, G., Hartman, J., Levas, T., Podlaseck, M. Interacting with steerable projected displays. *Intl. Conf. on Automatic Face and Gesture Recognition (FGR '02)*, pp. 402-407.

[3] Wilson, A.D. Using a depth camera as a touch sensor. *Proc. ITS '10*, pp. 69-72.