Pointer

Marcello Bastéa-Forte, Ron Yeh, Scott R. Klemmer
Stanford University HCI Group
{graphite, ronyeh, srk}@cs.stanford.edu

INTRODUCTION

With multi-touch, multi-user display devices becoming more prominent we wanted to make it easy to rapidly protoype new interactions with these devices.

We built **Pointer**:

- abstracts & aggregates multiple input sources
- framework to help test and develop software that utilizes multiple input sources

POINTER TOOLKIT

The toolkit consists of two components:

- API for aggregating and disseminating events
- debugging tool for analyzing and recording input as the application is running

Pointer API

- based on the Java Event architecture
- leverages existing APIs to provide connectivity to available hardware
- provides a standard input listener interface for applications to listen for point events
- point events represent hover, pressing, dragging, and releasing with coordinate, pressure, and size information

Pointer Debugging Interface

Central input aggregation let's us provide a number of techniques to help the developer debug their application, including:

- implicit logging of all events to a file for later playback and testing
- a sidebar GUI that provides real-time feedback of all input sources (figure 2 inset)

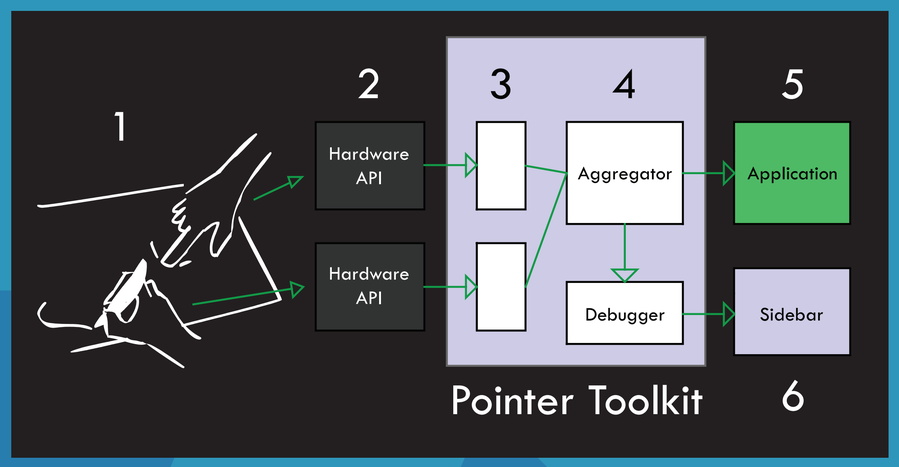

Figure 1: Input flow: user input (1) is read by an existing hardware API (2) which is sent through an input connector to the InputAggregator (4), which finally dispatches to the application (5).

Figure 2: The photo tagging application we developed using the Pointer API. Users can move pictures with their hands and annotate them with the Anoto pen. The inset shows the debug sidebar.

POINTER APPLICATIONS

Hello World! enumerates all event sources and displays each individual input as a collection of circles. This application demonstrates how to use the Pointer API and provides a starting point for new developers.

Photo Tagging integrates the Anoto digital pen with DiamondTouch table with a table-sized Anoto print (figure 2):

- DiamondTouch table recognizes multiple input points from up to four users
- Anoto pens accurately read stylus input across the entire table from multiple sources

We use this to give the user two modes of interaction using handlers:

- move, scale, and rotate photos with hands through the manipulatin handler
- annotate photos by writing on them through the stroke handler

CONCLUSIONS AND FUTURE WORK

Our maiden venture into the world of multi-point input APIs shows that it is worth exploring new toolkits for new interaction models:

- Pointer made rapid multi-input application development easy
- homogeneous input events encourages homogenous input handling—resulting in applications behaving as the user expects

We would like to expand our debugging interface to a more robust system that we can use to evaluate our system with developers and iterate on its design.