

TUIO, Touchlib, reactIVision and Community Core Vision

Qian Liu

Media Arts and Technology
University of California, Santa Barbara
qian@mat.ucsb.edu

Abstract

As the development of all different kinds of multi-touch technology, more and more multi-touch related software come into people's sight. This paper introduced TUIO, a widely used protocol for multi-touch technology based on OSC (open sound control). And compared popular software for detecting multi-touch events: Touchlib, reactIVision and Community Core Vision from a user's point of view.

Keywords: *multi-touch, TUIO, reactIVision, Touchlib, Community Core Vision, finger tracking*

1. Introduction

Nowadays, Multi-touch has been implemented in several different ways, depending on the size and type of interface. The hardware part of Multi-touch technology generally included four different kinds of settings: Frustrated total internal reflection(FTIR), Diffused Illumination(DI), Laser Light Plane(LLP) and Diffused Surface Illumination(DSI). The software part, one of the most popular API is Tangible User Interface Object(TUIO).

TUIO is an open framework that defines a common protocol and API for tangible multitouch surfaces. The TUIO protocol allows the transmission of an abstract description of interactive surfaces, including touch events and tangible object states. This protocol encodes control data from a tracker application and sends it to any client application that is capable of decoding the protocol. TUIO is based on Open Sound Control, which is a protocol for communication among computers, sound synthesizers, and other

multimedia devices that is optimized for modern networking, and can be therefore easily implemented on any platform that supports OSC.

This paper will briefly introduce TUIO protocol and compare three kinds of software: Touchlib, reactIVision and Community Core Vision, which are based on TUIO protocol and popular used in multi-touch technology, also include the reason to choose Community Core Vision as the software for TableMAT.

2. TUIO

The initial goal of the TUIO protocol definition was to provide a simple description of pointer and token states in the context of a two dimensional table surface, where pointers are defined as untagged points with normalized Cartesian coordinates, while tangible tokens provide an additional identification tag and rotation angle. Although this is a very simplified view of an interactive surface context, this description provides a basic solution for the

implementation of multi-touch surfaces and the tracking of tagged physical objects.

As TUIO protocol is encoded using OSC, the TUIO message can be basically transmitted through any channel that is supported by an actual OSC encapsulation of the binary OSC bundle data within UDP packets sent to the default TUIO port number 3333.

The TUIO protocol defines two main classes of messages: SET messages and ALIVE messages. SET messages are used to communicate information about an object's state such as position, orientation, and other recognized states. ALIVE messages indicate the current set of objects present on the surface using a list of unique Session IDs. In addition to SET and ALIVE messages, FSEQ messages are defined to uniquely tag each update step with a unique frame sequence ID. At typical TUIO bundle is therefore typically comprised of at least three messages, while the set messages can be accumulated in order to fully use the available space of a UDP packet. An optional SOURCE message identifies the TUIO source in order to allow source multiplexing on the client side.

There are two basic profiles for the description of pointers and tokens (here cursors and objects), which are commonly used within the context of a 2D surface. There exist additional profiles for 2.5D environments, which include the distance to the surface, as well as 3D environments, which also provide 3D rotation information for the object profiles. All profile types are generally designed to describe the surface or the space above an interactive table environment. Most currently available TUIO implementations are concentrating on the 2D profiles though.

The nice thing about TUIO is it is available for most common programming languages and media environments. TUIO has Application Frameworks for: open frameworks, Unity3D, Java, Python, Pure Data, Processing, Cocoa, QT, Max/MSP, etc; Client implementations: AS3, Objective C, Python, Ruby, VVVV, Open Frameworks, Firefox, JavaScript, Matlab; Tracker implementations for: reacTIVision, Community Core Vision, Touchlib, xTouch, BBTouch, TuioTablet, TuioTouch, etc.

3. Touchlib

Touchlib is a library for creating multi-touch interaction surfaces. It handles tracking blobs of infrared light, and sends your programs these multi-touch events, such as 'finger down', 'finger moved', and 'finger released'. Touchlib can broadcast events in the TUIO protocol. This makes touchlib compatible with several other applications that support this protocol, such as vvvv, Processing, PureData, etc.. This also makes it possible to use touchlib for blob detection / tracking and something like vvvv or Processing to write applications.

As Touchlib is written in C++, Visual Studio, it is currently just run in Windows system. However, install Touchlib in Mac OS X system is possible now, but needs a lot of pre-installation. To install Touchlib in Mac OS X system you need at least 6 steps:

Install Apple Developer Tools.

Install Darwinports and required packages using Darwinports.

Install OpenCV

Install OSCPack

Install SVN

Install Touchlib

This is why a lot of Mac OS X users tend to use other software instead of Touchlib.

4. reactTIVision

The reactTIVision system is software for tracking specially designed fiducials (markers) in a real-time video stream. ReactTIVision was designed to enable expressive gestural control of musical sound, and can track many markers at a high frame rate. The development of reactTIVision involved not only computer vision algorithms, but also the design of a new marker system.

reactTIVision is a standalone application, which sends TUIO messages via UDP port 3333 to any TUIO enabled client application. Alternatively reactTIVision is also able to send MIDI messages for the direct use with MIDI sequencers.

Same as Touchlib, reactTIVision can broadcast events in the TUIO protocol. reactTIVision's TUIO clients include: C++, Java, C#, Pure Data, Processing, Max/MSP, Quartz Composer, etc..

Fiducial tracking is one of the main functions of reactTIVision. When the software starts: the source image frame is first converted to a black&white image with an adaptive thresholding algorithm. Then this image is segmented into a tree of alternating black and white regions (region adjacency graph). This graph is then searched for unique left heavy depth sequences, which have been encoded into the fiducial symbol. Finally the found tree sequences are matched to a dictionary to retrieve an unique ID number. The fiducial design allows the efficient calculation of the marker's center point as well as its orientation. OSC messages implementing the

TUIO protocol encode the fiducials' presence, location, orientation and identity and transmit this data to the client applications.

Though reactTIVision does support finger tracking, as finger tracking was add at a later stage, it seems difficult to set up reactTIVision in order to achieve good tracking performance for both the fiducial symbols and the finger tips. Feedback from users shows that generally reactTIVision code for finger tracking has been much less robust than other trackers. So when it comes to finger tracking, users normally prefer other software.

5. Community Core Vision

Community Core Vision (CCV), also known as tbeta, is an open source/cross-platform solution for computer vision and machine sensing. It takes a video input stream and outputs tracking data, such as coordinates and blob size, and events, such as finger down, moved and released, that are used in building multi-touch applications. CCV can interface with various web cameras and video devices as well as connect to various TUIO/OSC/XML enabled applications and supports many multi-touch lighting techniques including: FTIR, DI, DSI, and LLP with expansion planned for the future vision applications.

Compare with reactTIVision and Touchlib, CCV has more filter option, which makes it easier for the users to balance and adjust the multi-touch table value. CCV's filter option includes: dynamic background subtraction, high-pass, amplify/scaler and threshold. This means it works with all optical setups (FTIR, DI, LLP, DSI). Also, CCV supports input switch, this feature has huge help for user who want to make applications themselves.

The test video makes it easier for users

to test their apps without connecting to the multi-touch table.

Name	Touchlib	reactTIVison	Community Core Vision
TUIO	Support	Support	Support
Operating system	Windows only. (complicated but possible to instill in Mac OS X)	Windows, Mac OS X, Linux	Windows, Mac OS X, Linux
Camera support	Webcam (USB or Firewire)	Windows: any camera with a proper WDM driver, such as USB, USB2, FireWire and DV cameras. Mac OS X: all FireWire cameras and any camera supported by QuickTime. Linux: FireWire cameras are best supported, as well as most Video4Linux2 USB cameras.	Windows: any camera with a proper WDM driver, such as USB, USB2, FireWire and DV cameras. Mac OS X: all FireWire cameras and any camera supported by QuickTime. Linux: FireWire cameras are best supported, as well as most Video4Linux2 USB cameras.
Finger tracking ability	Good	OK	GOOD
Other features		Able to send MIDI message, Fiducial tracking.	Test video input, dynamic filter options, dynamic mesh calibration: for different size of multi-touch surface, add calibration points or create less points while maintaining the same speed and performance.

6. Conclusion

By comparing Touchlib, reactTIVision and Community Core Vision and think about TableMAT project' need, CCV is more fit for this project. As the group used to have a debate about FTIR or DI setups, dynamic filters and support all kinds of optical setups make CCV more convenient for the group to work on. Also, cross-platform ensures all group members can work on there own application for the table. And the build in test video makes CCV even better.

In general, Touchlib, reactTIVision and Community Core Vision all have there own strength, it should be the user's choose that which one is more fit for the project.

7. References

- [1] TUIO Org, 1 April 2010. [online:]
<http://www.tuio.org/>
- [2] Multi-touch Wiki, 9 June 2010. [online:]
<http://en.wikipedia.org/wiki/Multi-touch>
- [3] Open Sound Control Org. [online:]
<http://opensoundcontrol.org/introducti-on-osc>
- [4] Touchlib, Febrary 2010. [online:]
<http://mtaha.wordpress.com/touchlib/>
- [5] Installing Touchlib on Mac, 5 November 2009. [online:]
http://wiki.nuigroup.com/Installing_Touchlib_on_Mac_OS_X
- [6] reactTIVision 1.4. [online:]
<http://reactivision.sourceforge.net/>
- [7] Community Core Vision. [online:]
<http://ccv.nuigroup.com/>

[8] Ross Bencina and Martin Kaltenbrunner, "The Design and Evolution of Fiducials for the reactTIVision System", 2009

[9] Martin Kaltenbrunner, "reactTIVision and TUIO: A Tangible Tabletop Toolkit", 2009