Government College of Engineering, Jalgaon
(An Autonomous Institute of Government of Maharashtra)

A Project Report

Flat Surface into Touchscreen

Under the Course “Miniproject” (IN360U)

Submitted by

Third Year B. Tech. (Instrumentation)

Sr. No.	Student Name	PRN. No.
1.	Tushar Gajanan Ubarhande	1821028
2.	Dinesh Kanoba Katakamwad	1821016
3.	Sumeet Dnyandeo Dongardive	1821011
4.	Jaykumar Atmaram Pawar	1821024

Under the Guidance of

Prof. R. D. Kokate

Instrumentation Engineering Department

GOVERNMENT COLLEGE OF ENGINEERING, JALGAON
(An Autonomous Institute of Government of Maharashtra)

2020-2021
(The College is affiliated to the North Maharashtra University, Jalgaon)

CERTIFICATE

This is to certify that below mentioned students of T.Y.B.Tech. (Instrumentation) have successfully completed the project entitled **"Flat Surface into Touchscreen"** under the course "Miniproject" (IN360U). The content of this report, in full or in parts, have not been submitted to any other institution or university for the award of any degree.

Sr. No.	Student Name	PRN. No.
1.	Tushar Gajanan Ubarhande	1821028
2.	Dinesh Kanoba Katakamwad	1821016
3.	Sumeet Dnyandeo Dongardive	1821011
4.	Jaykumar Atmaram Pawar	1821024

Dr. R. D. Kokate
Project Guide and Principal.

Dr. G. M. Malwatkar
Head of the Department Instrumentation.

Examiner

Place: GCOEJ, Jalgaon
Date:

DECLARATION

We, the undersigned, the students of T. Y. B. Tech. (Instrumentation) hereby declare that the project the project entitled **"Flat Surface into Touchscreen"** under the course "Miniproject" (IN360U) is a genuine work conducted by us through practical on–site observations, and the data collected by us is true to the extent of our awareness.

Sr. No.	Student Name	PRN. No.
1.	Tushar Gajanan Ubarhande	1821028
2.	Dinesh Kanoba Katakamwad	1821016
3.	Sumit Dnyandeo Dongardive	1821011
4.	Jaykumar Atmaram Pawar	1821024

Place: GCOEJ, Jalgaon
Date:

ACKNOWLEDGEMENT

In the accomplishment of this project successfully, many people have best owned upon me their blessing and the heart pledged support, this time I am utilize to thank all the people who have been concerned with this project.

Primarily I would like to thank good for being able to complete this project with success. Then I would like to thanks our Mini project Guide Prof. R. D. Kokate, whose valuable guidance has been the ones that helped me patch this project and make it full proof success. His suggestion and his instruction have served as the major contributor towards the completion of the project.

Then I would like to thank my parents and friends who have helped me with their valuable suggestion and guidance has been very helpful in various phase of the completion of the project.

Last but not the least I would like to thank my classmates who have helped me a lot for completion of this project.

1. **Tushar Ubarhande**
2. **Dinesh Katatamwad**
3. **Sumeet Dongardive**
4. **Jaykumar pawar**

Time format:

WBS	TASK	START	END	DAYS	% DONE	WORK DAYS
1	**[Problem Definition]**					
1.1	Need Statement	08/04/2021	08/042021	1	100%	1
1.2	Problem Definition Process	10/04/2021	11-04-2021	2	100%	2
1.3	initial Research and Market Survey	12-04-2021	12-04-2021	1	100%	1
1.4	Literature Survey	13-04-2021	13-04-2021	1	100%	1
1.4.1	Papers, Magazines, Books, Journals	14-02-2021	15-04-2021	2	100%	2
1.4.2	Internet	16-04-2021	18-04-2021	3	100%	3
1.5	Finalising Project Objectives, Constraints and Functions	19-04-2021	23-04-2021	4	100%	4
1.6	Finalising Project Plan	24-04-2021	24-04-2021	1	100%	1
2	**Design**					
2.1	Component requirement	25-04-2021	25-04-2021	1	100%	1
2.2	Base Circuit Diagram	26/04/201	26-04-2021	1	100%	1
2.3	Hardware Design	27-04-2021	28-04-2021	2	100%	2
3	**Development**					

3.1	Hardware installation	29-04-2021	29-04-2021	1	100%	1
3.2	Programming Development	30-04-2021	01-05-2021	2	100%	2
4	**Testing and Demo**					
4.1	Hardware Testing	02-05-2021	03-05-2021	2	100%	2
4.2	Program Testing	03-05-2021	04-05-2021	2	100%	2

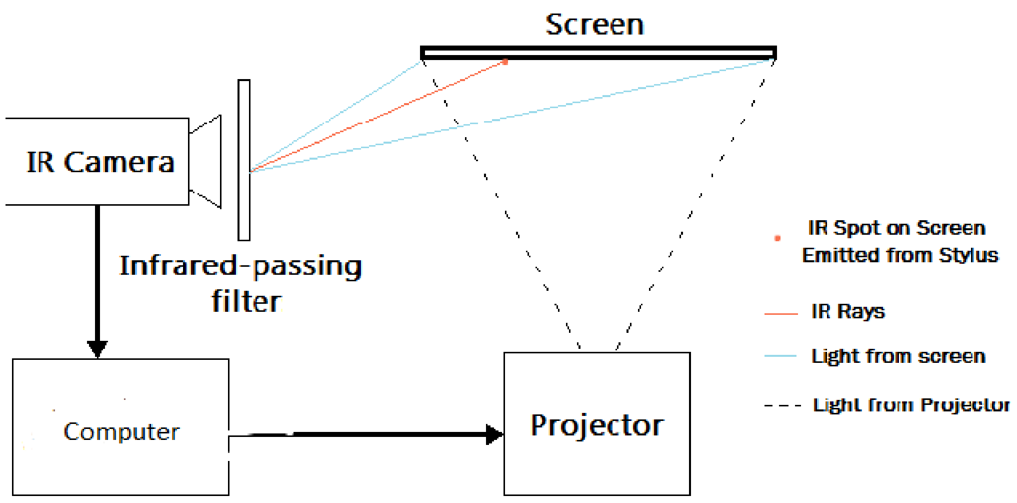
Table of content

INTRODUCTION

Multi-touch interaction with computationally enhanced surfaces has received considerable attention in recent years. Hardware implementations of multitouch interaction such as Frustrated Total Internal Reflection (FTIR) and Diffused Illumination (DI) have allowed for the low-cost development of surfaces. Although many of these technologies and associated applications have been presented in academic settings, the practicalities of building a high-quality multi-touch enabled surface, both in terms of the software and hardware required, are not widely known. We draw upon our extensive experience as developers of multi-touch technology to provide practical advice in relation to building, and deploying applications upon, multi-touch surfaces. This includes technical details of the construction of optical multi-touch surfaces, including: infrared illumination, silicone compliant surfaces, projection screens, cameras, filters, and projectors, and an overview of existing software libraries for tracking.

1.1 OVERVIEW

Tablets and all-in-one PCs featuring a touchscreen-enabled display are convenient and fun to work with. But a touchscreen monitor isn't quite affordable for most just yet. Using a Wii remote (Wiimote), we can make your regular monitor (LCD or CRT) behave as a basic touch panel.



Block Diagram for Touch Screen Interface.

1.2 LITURATURE REVIEW / BACKGROUND: -

This paper presents a novel and cost-effective approach to convert any projected display on a flat surface into interactive touch board, through the aid of computer vision techniques. An interactive touch board is similar to that of a touch screen available in mobile phones and tablets, but then uses a special stylus for its input and a LCD projector for projecting the display on the surface. Commonly used techniques fall under any of the four categories – Resistive, capacitive, Electromagnetic and Optical. The paper proposes a method pertaining to the optical technique – involving an infrared camera, light emitting stylus and an image processing algorithm to determine the position pointed by the user. The stylus has an infrared LED at its tip and capable of producing a tiny spot of light on the projected surface, which is invisible to human eyes. But an IR camera with IR pass filter can be used to capture the light spot and can be processed further. This is the principle technique behind the proposed method. The image from camera is processed using an algorithm, which determines the co-ordinates of the IR light. This obtained position along with some calibration parameters is sufficient to calculate the mouse cursor co-ordinates and thus the cursor can be moved to the intended position. Further, the paper deals with the study of potential application in education and scientific research.

1.3 Main objectives: -

The Main Objectives from this course are:

- Introduce a new technology that will help in making touch surfaces available in our country.
- Produce a low-cost system that can replace the traditional expensive touch screen systems.
- Improve people knowledge in how modern technologies used in many modern devices can be used to other purposes that make everybody’s life easier.

Hardware Implementation

On the hardware frontier, project aims to be an informational resource hub for others interested in prototyping and/or constructing, a low cost, high resolution open-source multi-input hardware system. There have been improvements to existing multi-touch systems as well as the creation of new techniques that allow for the development of multi-touch hardware systems. At the moment there are five major techniques that allow for the creation of a stable multi-touch hardware systems; these include: Jeff Han's pioneering Frustrated Total Internal Reflection (FTIR), Rear Diffused Illumination (Rear DI) such as Microsoft's Surface Table, Laser Light Plan (LLP) pioneered in the community by Alex Popovich and also seen in Microsoft's Laser Touch prototype, LED-Light Plane (LED-LP) developed within the community by Nima Motamedi, and finally Diffused Surface Illumination (DSI) developed by Tim Roth. These five techniques being utilized on the principal of Computer Vision and optics (cameras). While optical sensing makes up the vast majority of techniques, there are several other sensing techniques that can be utilized in making natural user interface and multitouch devices. Some of these sensing devices include proximity, acoustic, capacitive, resistive, motion, orientation, and pressure. Often, various sensors are combined to form a particular multitouch sensing technique.

2.1 Non-optical multi-touch approaches

Before describing optical multi-touch systems in more detail its appropriate to review alternative technologies. There is no fundamental characteristic that that optical approaches superior to the alternatives. Indeed, many of these alternatives have already found their way into consumer products, albeit in smaller interactive surfaces (e.g. mouse pads on laptops and touch-screens in phones). However, as already described, the principal drawback is that resistance, capacitance, or surface wave-touch screens, require industrial fabrication facilities.

2.1.1 Resistance-based Touch Surfaces

Resistance-based touch panels generally consist of two clear sheets coated with transparent conductive substances such as indium tin oxide. These surfaces are separated by an insulating layer, typically tiny silicon dots. The front of the panel is often made of a flexible hard coated outer membrane while the back panel is typically a glass substrate. A controller alternates between the layers, driving one with a specific (electric) current and measuring the current of the other. When users touch the display, the conductive layers are connected, establishing an electric current that is measured both horizontally and vertically (by the controller) to resolve the exact position of the touch event. Such touch surfaces have the advantage of low power consumption, are used in mobile devices such as the Nintendo DS, mobile devices and digital cameras, and can be operated using fingers or a stylus. However, resistance-based technologies generally yield low clarity interactive surfaces (i.e. 75%–85%) and additional screen protection cannot be added without significantly impacting on their sensitivity.

2.1.2 Capacitance-based Touch Surfaces

Capacitance based (multi-) touch surfaces can be broadly subdivided into two classes depending on the underlying sensing mechanism: (1) Surface Capacitance; and (2) Projected Capacitance. Both technologies were originally developed for single touch interaction, and one advantage of capacitive touch surfaces over competing Schoning et al. ¨ technologies is their high clarity; making capacitive touch surfaces very suitable for use where the display and touch sensitive surface are integrated (i.e. beyond simple touch pads). Capacitive touch screens are generally durable, reliable and can be operated by any conductive device and hence are not limited to finger-based interaction. However, they are relatively expensive to manufacture and are therefore usually reserved for use in rugged environments such as in public displays and industrial applications. Although it is possible to manufacture capacitive multi-touch surfaces, typically the number of simultaneous touches is limited by firmware and/or by the design of the controller. Furthermore, accuracy decreases when performing touches with more than one object, although a number of capacitance-based technologies have been developed that overcome many of these restrictions in order to allow many simultaneous touches (e.g. MERLs DiamondTouch).

2.1.2.1 Surface Capacitive Touch Surfaces

Surface capacitive touch panels consist of a uniform conductive coating on a glass layer. Compared to resistive technologies, a much higher clarity can be achieved by again using indium tin oxide as the conducting material (it is transparent as well as colourless when used in very thin layers). From each side of the touch panel electrodes maintain a precisely controlled store or electrons in the horizontal and vertical directions thereby setting up a uniform electric field across the conductive layer. As fingers (and other conductive objects) are also electrical devices capable of storing charge and supporting electric fields, touching the panel results in a small transport of charge from the electric field of the panel to the field of the touching object. Current is drawn from each corner of the panel; this process is measured with sensors located in the corners, and a microprocessor interpolates an exact position of the touch based on the values measured. Panels based on surface capacitive technology can provide a high positional accuracy.

2.1.2.2 Projected Capacitive Touch Surfaces

Of all the technologies we describe projected capacitive touch devices are the most expensive to produce. Their performance is also rather worse than competing technologies; however, they afford superb mechanical resilience. Projected capacitive surfaces can also be covered by a non-conductive material (up to a maximum thickness of approximately 20mm) without significantly impacting on their functionality. When used for (multi-) touch displays, as described by Rekimoto) a very thin grid of microphone wires is installed between two protective glass layers. When touched, the capacitance between the finger and the sensor grid, and the touch location can be computed based on the electrical characteristics of the grid layer. The accuracy of projected capacitive technology is similar to that of surface capacitive technology although light transmission is often better since the wire grid can be constructed such that it is nearly transparent. The technology is also very suitableBuilding interactive multi-touch surfaces 5 for rugged environments such as public installations, as a protective layer (such as thick glass) may be added without drastically decreasing the sensitivity. Finally, compared to surface capacitance technology, multiple simultaneous touches can be more easily interpreted.

2.1.3 Surface Acoustic Wave Touch Surfaces (SAW)

In surface acoustic wave surfaces transmitting and receiving piezoelectric transducers, for both the X- and Y-axes, are mounted on a faceplate, and ultra-sonic waves on a glass surface are created and directed by reflectors. By processing these electronic signals and observing the changes when the faceplate is touched, it is possible to calculate the position of that interaction. Most SAW systems can support dual-touch.

2.2 OPTICAL MULTI-TOUCH TECHNOLOGIES.

Optical or light sensing (camera) based solutions make up a large percentage of multi-touch devices. The scalability, low cost and ease of setup are suggestive reasoning for the popularity of optical solutions. Stereo Vision, Overhead cameras, Frustrated Total Internal Reflection, Front and Rear Diffused Illumination, Laser Light Plane, and Diffused Surface Illumination are all examples of camera based multi-touch systems. Each of these techniques consist of an optical sensor (typically a camera), infrared light source, and visual feedback in the form of projection or LCD.

2.2.1 Infrared Light Sources: -

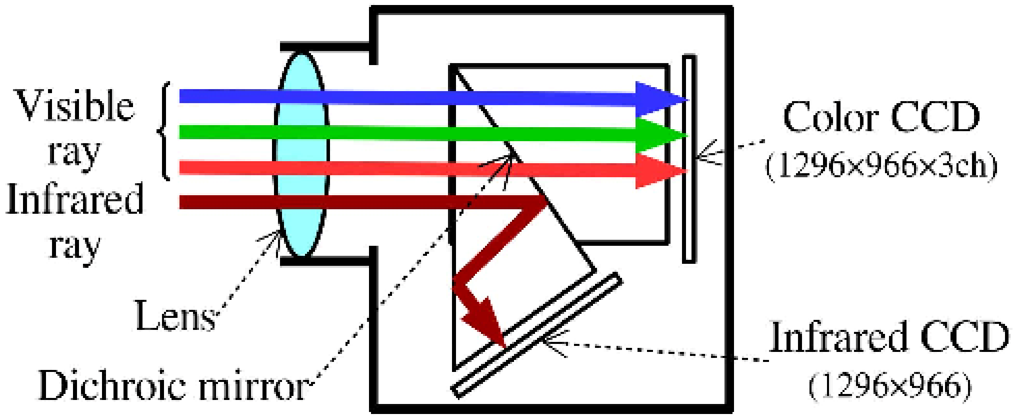
Infrared (IR in short) is a portion of the light spectrum that lies just beyond what can be seen by the human eye. It is a range of wavelengths longer than visible light, but shorter than microwaves. 'Near Infrared' (NIR) is the lower end of the infrared light spectrum and typically considered wavelengths between 700nm and 1000nm (nanometers). Most digital camera sensors are also sensitive to at least NIR and are often fitted with a filter to remove that part of the spectrum so they only capture the visible light spectrum. By removing the infrared filter and replacing it with one that removes the visible light instead, a camera that only sees infrared light is created. In regards to multitouch, infrared light is mainly used to distinguish between a visual image on the touch surface and the object(s)/finger(s) being tracked. Since most systems have a visual feedback system where an image from a projector, LCD or other display is on the touch surface, it is important that the camera does not see this image when attempting to track objects overlyaed on the display. In order to separate the objects being tracked from the visual display, a camera, as explained above, is modified to only see the infrared spectrum of light; this cuts out the visual image (visible light spectrum) from being seen by the camera and therefore, the camera is able to see only the infrared light that illuminates the object(s)/finger(s) on the touch surface.

LEDs can be bought in the following forms.

- Single LEDs
- LED ribbons
- LED emitters

2.2.2 Infrared Cameras: -

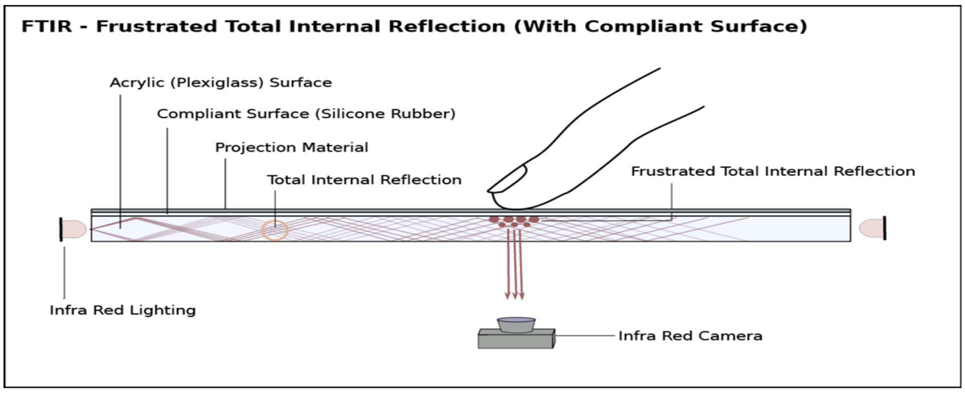
Simple webcams work very well for multitouch setups, but they need to be modified first. Regular webcams and cameras block out infrared light, letting only visible light in. We need just the opposite. Typically, by opening the camera up, we can simply pop the filter off, but on expensive cameras this filter is usually applied directly to the lens and cannot be modified. Most cameras will show some infrared light without modification, but much better performance can be achieved if the filter is replaced. The performance of the multi-touch device depends on the used components.



An infinite dot pattern is projected on a screen by a projector so that infrared cameras can capture the scene textured by the dots and the depth can be estimated even where the surface is not textured. The cost volumes are calculated for the infrared and colour stereo images for each frame and are extended the cost volume filter in the time direction by modifying the cross-based local multipoint filter (CLMF) and applying it to the cost volumes in order to restrain flicker on the time-varying depth maps. To get a reliable cost volume, infrared and colour cost volumes are integrated into a single cost volume by selecting the cost of either the infrared or the colour cost volumes according to the size of the adaptive kernel used for the CLMF. Then, a graph cut is executed on the cost volume in order to estimate the disparity robustly even when the baselines of the stereo cameras are set wide enough to ensure spatially high resolution in the depth direction and the shapes of blocks are deformed by the affine transformation. A 2D graph cut is executed on each scan line to reduce the processing time and memory consumption. We experimented with the proposed method using infrared colour stereo data sets of scenes in the real world and evaluated its effectiveness by comparing it with other recent stereo matching methods and depth cameras.

2.2.3 FTIR filter: -

FTIR is a name used by the multi-touch community to describe an optical multi-touch methodology developed by Jeff Han (Han 2005). The phrase actually refers to the well-known underlying optical phenomenon underlying Han's method. Total Internal Reflection describes a condition present in certain materials when light enters one material from another material with a higher refractive index, at an angle of incidence greater than a specific angle (Gettys, Keller and Skove 1989, p.799). The specific angle at which this occurs depends on the refractive indexes of both materials, and is known as the critical angle, which can be calculated mathematically using Snell's law. When this happens, no refraction occurs in the material, and the light beam is totally reflected. Han's method uses this to great effect, flooding the inside of a piece of acrylic with infrared light by trapping the light rays within the acrylic using the principle of Total Internal Reflection. When the user comes into contact with the surface, the light rays are said to be frustrated, since they can now pass through into the contact material (usually skin), and the reflection is no longer total at that point. [Fig. 2] This frustrated light is scattered downwards towards an infrared webcam, capable of picking these 'blobs' up, and relaying them to tracking software.


FTIR - Frustrated Total Internal Reflection (With Compliant Surface)
Acrylic (Plexiglass) Surface
Compliant Surface (Silicone Rubber)
Projection Material
Total Internal Reflection
Frustrated Total Internal Reflection
Infra Red Lighting
Infra Red Camera

2.2.4 Compliant surfaces: -

The compliant surface is an overlay placed above the acrylic waveguide in a FTIR based multi-touch system. The compliant surface overlay needs to be made of a material that has a higher refractive index than that of the acrylic waveguide, and one that will "couple" with the acrylic surface under pressure and set off the FTIR effect, and then "uncouple" once the pressure is released. Note that compliant surfaces are only needed for FTIR - not for any other method (DI, LLP, DSI). The compliant surface overlay can also be used as a projection screen. The compliant surface or compliant layer is simply an additional layer between the projection surface and the acrylic. It enhances the finger contact and gives you more robust blobs, particularly when dragging as your finger will have less adhesion to the surface. In the FTIR technique, the infrared light is emitted into side the acrylic waveguide, the light travels inside the medium (due to total internal refection much like a fibre optic cable), when you touch the surface of the acrylic (you frustrate this TIR effect) causing the light to refract within the medium on points of contact and creating the blobs (bright luminescent objects). There is much experimentation ongoing in the quest for the 'perfect compliant layer'.

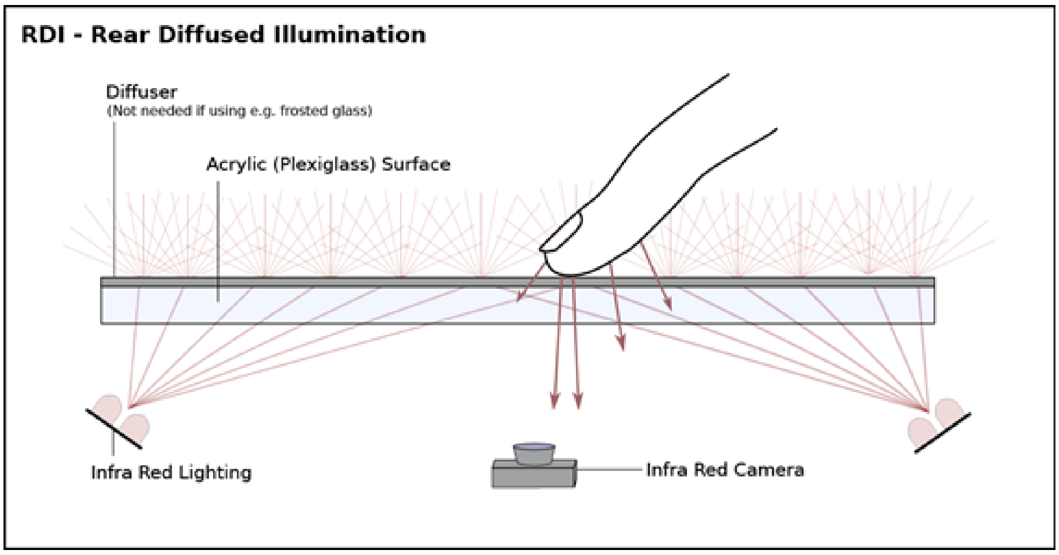
Developing a compliant surface for a LCD FTIR setup is difficult, as the surface must be absolutely clear and distortion-free, so as to not obscure the LCD image. This difficulty is not present in projection-based FTIR setups, as the image is projected onto the compliant surface itself. To date, no one has successfully built a LCD FTIR setup with a 100% clear compliant surface. However, several individuals have had success with LCD FTIR setups, with no compliant surface whatsoever. It appears that the need for a compliant surface largely depends on how strong blobs are without one, and specific setup

2.2.5 Diffused Illumination: -

Diffused Illumination (DI) comes in two main forms: Front Diffused Illumination and Rear Diffused Illumination. Both techniques rely on the same basic principles - the contrast between the silent image and the finger that touches the surface.

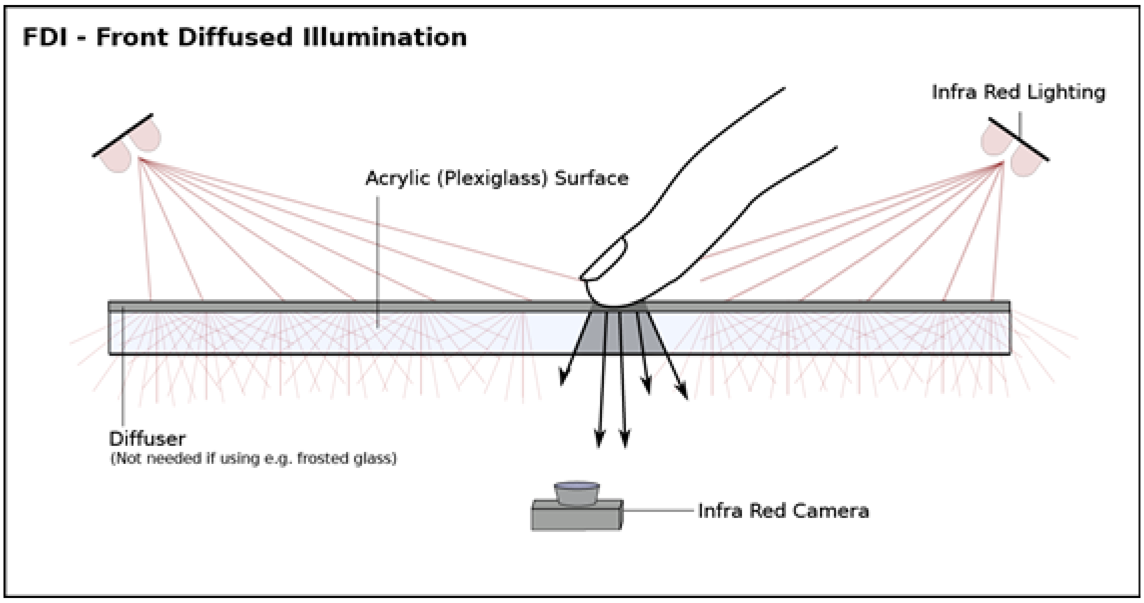
2.2.5.1 Rear DI: -

In Rear DI, infrared light is shined at the screen from *below* the touch surface. A diffuser is placed on top (preferred) or on bottom of the touch surface. When a finger or object touches the surface, the infrared light hits the object and is reflected downward and seen by an infrared camera below the surface. Depending on the diffuser, this method can also detect hovering objects, hands, and fingers above the surface. An image is displayed by using a projector connected to a computer pointed towards the projection material.


RDI - Rear Diffused Illumination
Diffuser
(Not needed if using e.g. frosted glass)
Acrylic (Plexiglass) Surface
Infra Red Lighting
Infra Red Camera

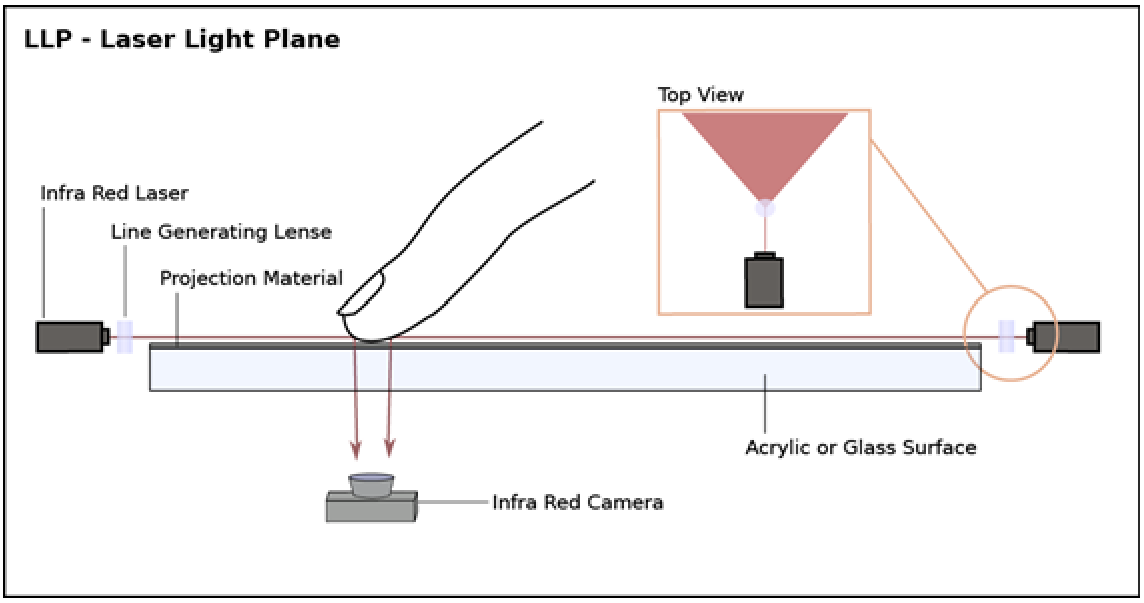
2.2.5.2 Front DI: -

In Front DI, infrared light is shined at the screen from *above* the touch surface. A diffuser is placed on top (preferred) or on bottom of the touch surface. When a finger or object touches the surface, a shadow is created under the object and seen by an infrared camera below the surface. An image is displayed by using a projector connected to a computer pointed towards the projection material. Visible light (often from the ambient surroundings) is shined at the screen from above the touch surface. A diffuser is placed on top or on bottom of the touch surface. When an object touches the surface, a shadow is created in the position of the object. The camera senses this shadow


FDI - Front Diffused Illumination
Infra Red Lighting
Acrylic (Plexiglass) Surface
Diffuser
(Not needed if using e.g. frosted glass)
Infra Red Camera

2.2.6 Laser Light Plane: -

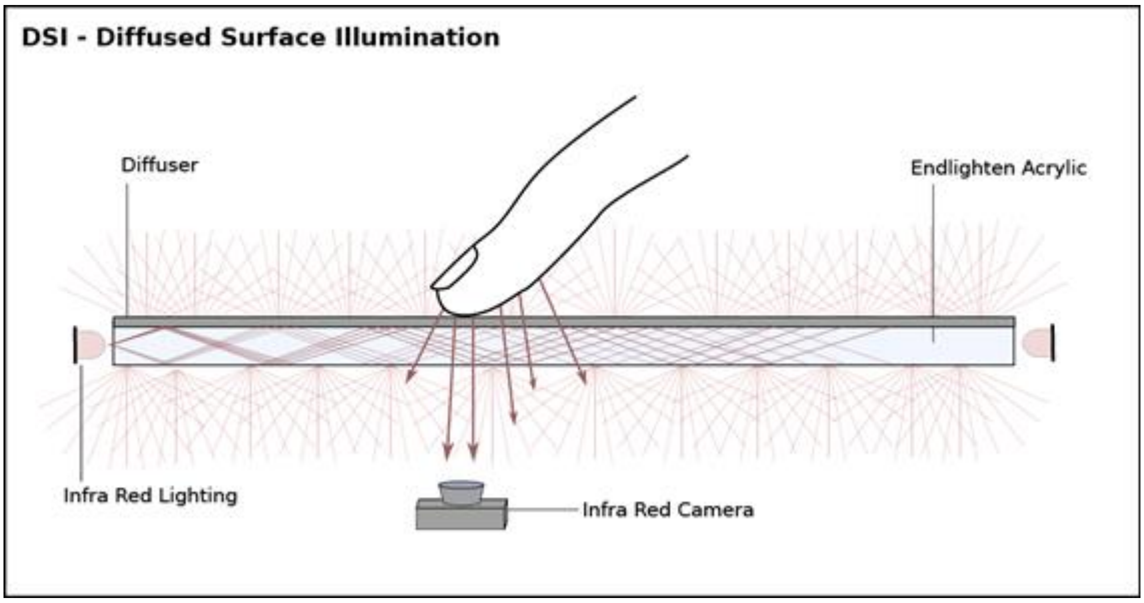
In LLP, infrared light from single or multiple lasers shine just above the surface. The laser plane of light is about 1mm thick and positioned as close to the touch surface as possible. When a finger or object hits the light plane, the object lights up and is seen by an infrared camera below the surface. An image is displayed by using a projector connected to a computer pointed towards the projection material.



Laser Light Plane (LLP) Infrared light from a laser(s) is shined just above the surface. The laser plane of light is about 1mm thick and is positioned right above the surface, when the finger just touches it, it will hit the tip of the finger which will register as a IR blob. Infrared lasers are an easy and usually inexpensive way to create a MT setup using the LLP method. Most setups go with 2-4 lasers, positioned on the corners of the touch surface. The laser wattage power rating (mW,W) is related to the brightness of the laser, so the more power the brighter the IR plane will be. The common light wavelengths used are 780nm and 940nm. Laser modules need to have line lenses on them to create a light plane. The 120 degree line lens is most commonly used, so as to reduce the number of lasers necessary to cover the entire touch surface. Safety when using lasers of any power is important.

2.2.7 Diffused Surface Illumination (DSI) : -

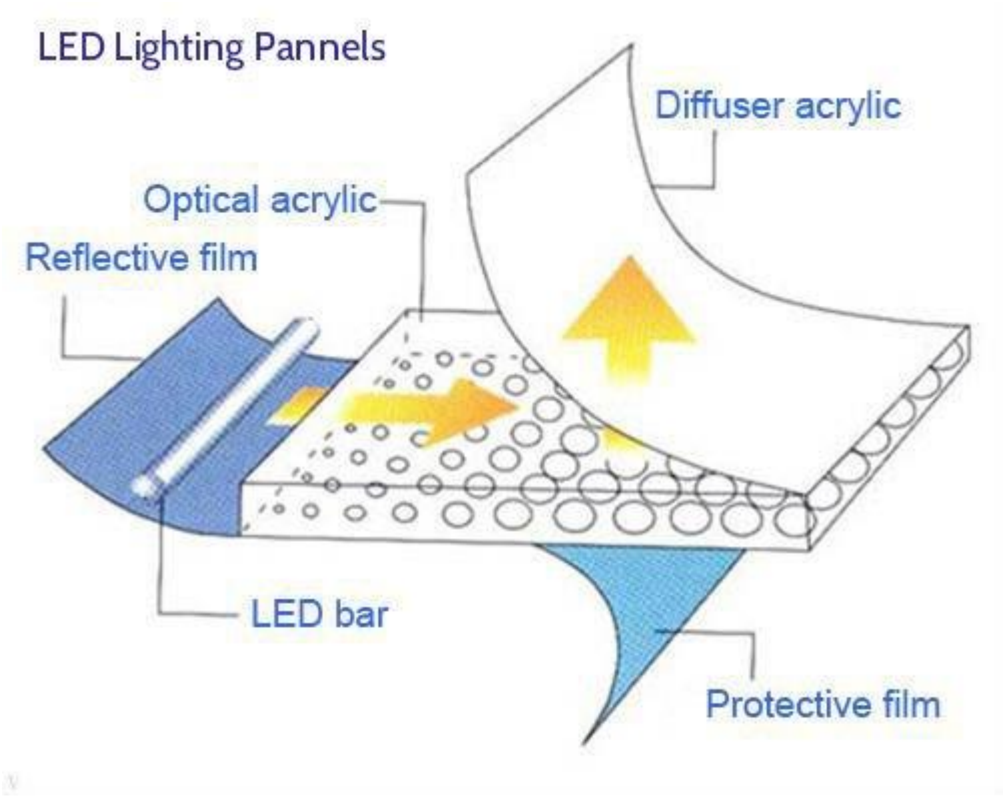
DSI is similar to FTIR in that infrared light is placed at the edges of an acrylic panel directed towards the inside. A special acrylic with small particles inside, acting like thousands of small mirrors, is used to evenly distribute the infrared light. When a finger or object touches the diffuser, it lights up from the infrared light escaping from within and is seen by a camera below the surface. The effect is similar to diffused illumination. An image is displayed by using a projector connected to a computer pointed towards the projection material.



DSI uses a special acrylic to distribute the IR evenly across the surface. Basically, this works will with the use of standard FTIR setup with an LED Frame (no compliant silicone surface needed), and just switching to a special acrylic. This acrylic uses small particles that are inside the material, acting like thousands of small mirrors. When we shine IR light into the edges of this material, the light gets redirected and spread to the surface of the acrylic. The effect is similar to DI, but with even illumination, no hotspots, and same setup process as FTIR.

2.2.8 LED panel lights: -

LED panel lights are one of the most commonly used LED backlit lamps, well-known for their even light output, streamlined design structure and zero glare problems. Nowadays, they have become a popular option for modern interior lighting. However, very few people have any idea about the structural components of these products. Let's have a look at some of them in detail:



Body Frame :
This is the edge cover designed specifically for ensuring thermal conductivity as well as inner protection. The body frame is generally made of aluminium, which provides a sleek outlook and proper heat dissipation. Some manufacturers are even making use of PVC for framing LED lighting panel. Although PVC is quite efficient when it comes to protection, the material is a very poor conductor of heat.

Guide Plate
The core component of any LED panel light is its guide plate. The design of this component is the most important technique dictating the effect of output light. Poor guide plates can cause dark spots and non-uniform output. Therefore, most manufacturers try to use either an effective plate design or top quality guide plates.

Reflective Plate

This one is generally applied for improving light efficiency in order to ensure high brightness. It is normally made out of a nanoscale material, the reflective rate of which can hiked up to 92 percent.

Light Diffuser
The diffuser helps in controlling output as well as the custom dimming options. It is manufactured using polymethyl methacrylate (PMMA) or polycarbonate (PC). Those made with PMMA are highly efficient, their performance often reaching upto 92 percent. On the other hand, PC made diffusers are very good at resisting oxidation. However, PMMA is used more commonly than PC at present.

Power Supply
For drawing power, LED panel lights have a driver system that ensures constant supply of current. The driver can either be isolated or non-isolated.

Light Source
The light source used in modern LED panel lights is a particular type of light emitting diode (2835). It is commonly used by most manufacturers due to its low power consumption and high luminous output.

Installation Accessories
These are the mounting brackets, suspension wires, etc. They are used for the primary purpose of installation. In order to increase sales, many manufacturers spend a great deal on these accessories and enhance their appearance, quality, design and performance

2.2.9 LCD Monitors: -

While projection-based displays tend to be the most versatile in terms of setup size, LCD displays are also an option for providing visual feedback on multi-touch setups. All LCD displays are inherently transparent – the LCD matrix itself has no opacity. If all of the plastic casing of the display is removed and IR-blocking diffuser layers are discarded, this LCD matrix remains, attached to its circuit boards, controller, and power supply (PSU). When mounted in a multi-touch setup, this LCD matrix allows infrared light to pass through it, and at the same time, display an image rivalling that of an unmodified LCD monitor or projector. Naturally, in order for an LCD monitor to produce an image when disassembled, it must be connected to its circuit boards. These circuit boards are attached to the matrix via FFC (flat flex cable) ribbon that is of a certain length. A LCD monitor is unusable for a multi-touch setup if these FFC ribbons are too short to extend all components of the LCD monitor far enough away from the matrix itself so as to not inhibit transparency of infrared light through it. Databases of FFCcompliant LCD monitors exist on websites such as Lumen Labs, up to 19”. LCD monitors that are not FFC compliant are still potentially usable for multi-touch setups, if the FFC ribbons are extended manually. FFC extensions can be found in electronics parts stores, and can be soldered onto existing cable to make it the necessary length.

Software

3.1 Introduction to Software Programming: -

Programming for multi-touch input is much like any other form of coding, however there are certain protocols, methods, and standards in the multi-touch world of programming. Through the work of NUI Group and other organizations, frameworks have been developed for several languages, such as ActionScript 3, Python, C, C++, C#, and Java. Multi-touch programming is two-fold: reading and translating the “blob” input from the camera or other input device, and relaying this information through pre-defined protocols to frameworks which allow this raw blob data to be assembled into gestures that high-level language can then use to interact with an application. TUIO (Tangible User Interface Protocol) has become the industry standard for tracking blob data, and the following chapters discuss both aspects of multi-touch software: touch tracking, as well as the applications operating off of the tracking frameworks.

3.1.1 Tracking: -

Tracking for Multi-Touch Tracking is very important to multi-touch technology. It is what enables multiple fingers to perform various actions without interrupting each other. We are also able to identify gestures because the trajectory of each finger can be traced over time, impossible without tracking. Thankfully, today’s multi-touch hardware greatly simplifies the task of tracking an object, so even the simplest Kalman filter in actuality becomes unnecessary. In fact, much of the performance bottleneck of tracking systems tends to come from generating and maintaining a model of the background. Computational costs of these systems heavily constrict CPU usage unless clever tricks are employed. However, with the current infrared (IR) approaches in multi-touch hardware (e.g., FTIR or DI), an adaptive background model turns out to be overkill. Due to the fact that the captured images filter out (non-infrared) light, much of the background is removed by the hardware. Given these IR images, it is often sufficient to simply capture a single static background image to remove nearly all of the ambient light. This background image is then subtracted from all subsequent frames, the resultant frames have a threshold applied to them, and we are left with images containing ‘blobs’ of foreground objects (fingers or surface widgets we wish to track)

3.2 Python Multi-touch Modules and Projects: -

The following is a brief overview of some multi-touch related python projects and modules available:

PyMT

PyMT is a python module for developing multi-touch enabled media rich OpenGL applications. It was started as a research project at the University of Iowa, and more recently has been maintained and grown substantially thanks to a group of very motivated and talented open source developers. It runs on Linux, OSX, and Windows. PyMT is licensed under the GPL license.

touchPy

Python framework for working with the TUIO protocol. touchPy listens to TUIO input and implements an Observer pattern that developers can use or subclass. The Observer pattern makes touchPy platform and module agnostic. For example, it does not care which framework is used to draw to the screen. There are a series of great tutorials written for touchPy by Alex Teiche.

PointIR

PointIR is a python based multi-touch system demoed at PyCon 2008. While it certainly looks very promising, it seems to have vanished from the (searchable?) realms of the Internet. Specific license information is unknown.

libAVG

According to its web page, "libavg is a high-level media development platform with a focus on interactive installations". While not strictly a multi-touch module, it has been used successfully to receive TUIO input and do blob tracking to work as a multi-touch capable system. libavg is currently available for Linux and Mac OS X. It is open source and licensed under the LGPL.

pyTUIO

A python module for receiving and parsing TUIO input. pyTUIO is licensed under the MIT license.Software & Applications 39 2.4.3

PyMT

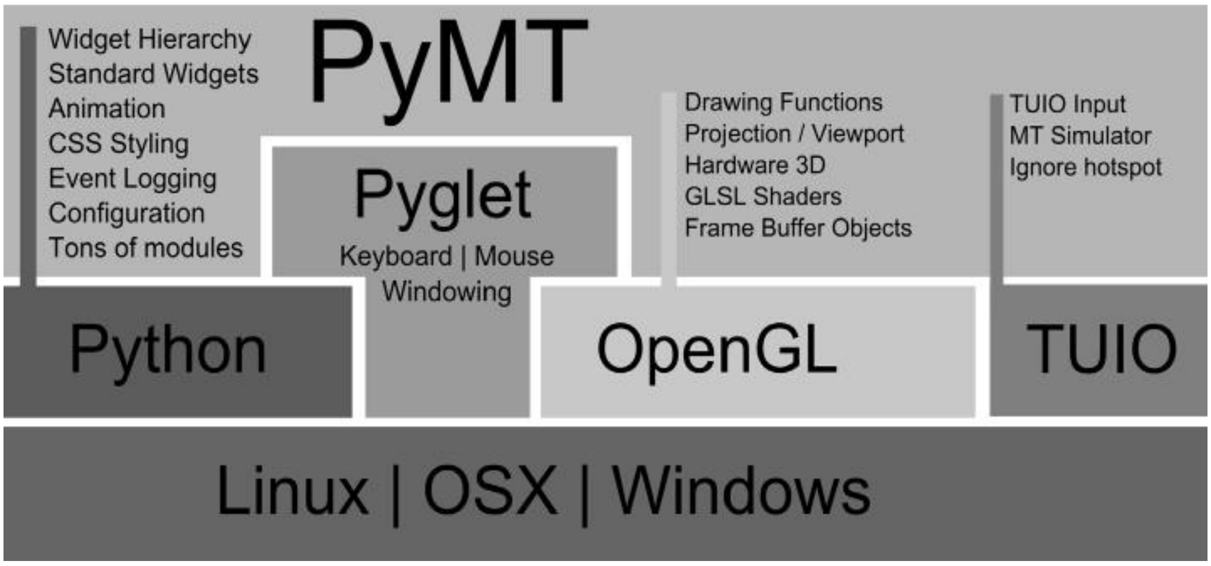
PyMT is a python module for developing multi-touch enabled media rich OpenGL applications. The main objective of PyMT is to make developing novel, and custom interfaces as easy and fast as possible. This section will introduce PyMT in more detail, discuss its architecture, and use it as an example to discuss some of the challenges involved in writing software for multi-touch user interfaces. Every (useful) program ever written does two things: take input and provide output. Without some sort of input, a program would always produce the exact same output (e.g "Hello World!"), which would make the program rather useless. Without output, no one would ever know what the program was doing or if it was even doing anything at all. For applications on multi-touch displays, the main method of input is touch. Graphics drawn on the display are the primary output. PyMT tries to make dealing with these two forms of input and output as easy and flexible as possible. For dealing with input, PyMT wraps the TUIO protocol into an event driven widget framework. For graphical output PyMT builds on OpenGL to allow for hardware accelerated graphics and allow maximum flexibility in drawing.

PyMT Architecture

Architecture displays the main architecture of PyMT. As already mentioned it uses TUIO and OpenGL for input/output. The current version of PyMT relies on pyglet, which is a cross-platform OpenGL windowing and multimedia library for Python. Especially pyglet's multimedia functionality makes dealing with images, audio and video files very easy. Most media files can be loaded with a single line of python code (and drawn/played to an OpenGL context with a single line as well). This subsection briefly discusses the role of OpenGL as a rendinering engine as well as the event system and widget hierarchy at the heart of PyMT.

OpenGL

Using OpenGL as a drawing backend has both advantages and disadvantages. While it allows for ultimate performance and flexibility in terms of 2D and 3D rendering, it is also very low-level. Therefore, it's learning curve can be rather steep and requires a good understanding of basic of computer graphics. PyMT tries to counteract the need for advanced OpenGL knowledge by providing basic drawing functions that act as wrappers to OpenGL. For example, PyMT includes functions such as drawCircle, drawRectangle, drawLine, draw TexturedRectangle and many others that require no knowledge of OpenGL. Using OpenGL as an underlying rendering engine however, lets more advanced users take full control over the visual output of their applications at peek performance. PyMT also provides helper functions and classes to aid advanced OpenGL programming. Creating, for example, a Frame Buffer Object or shaders from glsl source can be done in one line. PyMT also handles projection matrices and layout transformations under the hood, so that widgets can always draw and act in their local coordinate space without having to deal with outside parameters and transformations. The main idea is to do things the easy way most of the time; but if advanced techniques or custom drawing is required to provide easy access to raw OpenGL power.



3.3 Programming: -

Programming for Multi-Touch Input Programming applications and interactions for multi-touch devices is very different from developing classic mouse-based interfaces. The major differences lies in having to handle multiple simultaneous touches/cursors. While being able to interpret multiple touches hugely expands the possibilities for potential interactions and interfaces, it also adds a layer complexity to the programming and computational aspect of development that is far from trivial. The way TUIO, and most other multi-touch protocols and frameworks handle the issue of receiving event information about touch input defines three types of basic messages/events. These signal a new touch, movement of an existing touch, or the removal of a touch. The messages are always annotated with a “touch ID”, so that the program and its callback function can handle the event in the context of how that specific touch and others have been presented in prior events. As seen in the example from Listing1, these events arrive in PyMT as:

• on_touch_down(touches, touchID, x, y)

• on_touch_move(touches, touchID, x, y)

• on_touch_up(touches, touchID, x, y)

Each event carries in addition to the touchID the x and y location in pixels co-ordinates of the event relative to the containing window. Also, a dictionary with touchID’s of all currently alive touches as keys is provided with the touches hashmap. Using the touches dictionary, the programmer can inspect all current touches and their locations at the time of the current event. In fact, the touches dictionary holds TUIO2DCursor objects as its values, which hold additional information such as relative movement and acceleration as defined by the TUIO protocol. PyMT also provides on_object_* events for TUIO object messages (e.g. used with fiducial markers). It quickly becomes clear that interpreting multiple simultaneous touches becomes much more complex than handling a single pointer for e.g. a mouse. The context of the user interface can change with each individual touch, and subsequent event handlers must be aware of all the interactions happening before making a decision about how to deal with a touch event. 46 Multi-Touch Technologies Handbook A couple of general programming techniques have proven helpful while experimenting with PyMT. For example, the concept of giving a widget ownership of a specific touch has proven useful in various cases. When using this technique other widgets ignore subsequent touch_move or touch_up events with a specific touchID. Only the widget that has taken ownership will handle these events.

3.3.1 Drawing with Matrix Transforms: -

It describes the first part needed for the rotate/scale/move interaction. Here the drawing function draws the object its working with. transform_mat is a transformation matrix. So far it only holds the identity matrix, which leaves points unchanged when multiplied. Implementing the interaction is now a question of modifying the transformation matrix appropriately based on the touch input.

```
#a matrix that holds the transformation we are applying to our object
self.transform_mat = IdentityMatrix()
#a hashmap to keep track of the touchID’s we have seen
self.touches = {}
#this function draws the object using the transformation matrix
def draw(self):
        glPushMatrix()          #save the current matrix state
glMultMatrixf(self.transform_mat)       #apply transformation
        self.draw_object()      #draw the object as usual
        glPopMatrix()           #restore the current matrix state
```

3.3.2 Parameterizing the Problem and Computing the Transformation: -

Next, the problem lies in determining how to transform the transformation matrix of the object. An important observation to make is that for any given event only one of the two touches can have moved. This is because every event only caries information for one touch at a time. To compute the new transformation the following parameters are required. The distance between the two fingers before and after the event (d1 and d2). The angle R by which to rotate the object. And the point Pi around which to rotate/scale (By our observation this is one of the two original points)

Another noteworthy observation is that no translation (plain move) is ever applied. Using this technique, the movement of the of the object is achieved by rotating around the point that didn't change in the last event, or scaling and shrinking it in different directions subsequently. The translation happens automatically as events occur after each other. However, to do e.g. a one finger drag, which translates the object, a translation would have to be added as it is in the actual implementation of ScatterWidget in PyMT.

```
#this functions applies the newly computed transformations
#to the old matrix
def apply_angle_scale_trans(self, angle, scale, point):
        glPushMatrix()            #save the current matrix state
        glLoadIdentity()
        glTranslated(point.x, point.y,0)            #5. move back to intersection
        glScaled(scale, scale,1)            #4. Scale around (0,0)
        glRotated(angle,0,0,1)            #3. Rotate around (0,0)
        glTranslated(-point.x, -point.y,0)            #2. Move intersec- tion to (0,0)
        glMultMatrixf(self.transform_mat)            #1. apply transform as was
        #now save the modified matrix back to self.trans- form_mat
        glGetFloatv(GL_MODELVIEW_MATRIX,self.transform_mat)
        glPopMatrix()            #restore the current matrix state
```

3.3.3 Interpreting Touch Locations to Compute the Parameters: -

Finally, the program must compute the parameters solely based on the location provided by the touch events. For simplicity, this code assumes a function get_second_touch to get the information about the second touch involved in the interaction. A function like this can be implemented e.g. by using a dictionary to keep track of the touchID's, making sure only two touches are in it at any given point in time, and then returning the one not equal to the argument passed. The code also assumes some basic vector functions for computing lpoint distances and angles between lines. Implementations for these can be found in PyMT's vector class. The computations basically boil down to the magnitude of a vector and the dot product between two vectors. To angle of rotation is given by the angle between the lines of A1-B (where the current touch used to be (A1) and the second touch) and A2-B (where the touch Software & Applications 51 is now and the second touch). The scale factor is easily computed by the ratio of the new distance d2 divided by the old d1. Where d1 a= |A1, B| and d1 a= |A2, B|.

```
def rotate_zoom_move(self, touchID, x, y):
        intersect = Vector(0,0)
        rotation = 0
        scale = 1
        #we definitly have one point, so if nothing else we drag/translate
        A1= Vector(*self.touches[touchID])
        A2= Vector(x,y)
        #get the second touch
        #(not the on with touchID of current event)
        second_touch = self.get_second_touch(touchID)
        B = Vector(*self.touches[second_touch])
        #compute scale factor (d1 and d2 are floats)
        d1= A1.distance(B)
        d2= A2.distance(B)
        scale = d1/d2
        #compute rotation angle
        old_line = A1 - B
        new_line = A2 - B
        rotation = -1.0 * old_line.angle(new_line)
        #apply to our transformation matrix
        self.apply_angle_scale_trans(rotation, scale, B)
        #save new position of the current touch
        self.touches[touchID] = Vector(x,y)
```

3.4 Frameworks and Libraries: -

Bespoke Multi-Touch Framework

The Bespoke Multi-Touch Framework is a feature-rich and extensible software framework for developing multi-touch interfaces. Licensed under the BSD opensource license, you are free to use and extend the source code to suit your purposes. The framework can be paired with any vision-based multi-touch hardware platform (e.g. FTIR, or Diffused Illumination). The package includes a number of example applications, a Windows mouse emulator, 2D Ink/Symbol recognition, 4-point calibration, and independent presentation layer (support for XNA and WinForms included), and OSC network support using unicast, multi-cast, and broadcast UDP/IP.

Programming Language: C#

License: BSD License

Link: http://www.bespokesoftware.org/multi-touch

reacTIVision

reacTIVision is an open source, cross-platform computer vision framework for the fast and robust tracking of fiducial markers attached onto physical objects, as well as for multi-touch finger tracking. It was mainly designed as a toolkit for the rapid development of table-based tangible user interfaces (TUI) and multi-touch interactive surfaces.

Programming Language: C++60 Multi-Touch Technologies Handbook

License: GPL license

Link: http://reactivision.sourceforge.net/

Community Core Vision (CCV)

Community Core Vision, CCV for short, and formerly tBeta, is a open source/ cross-platform solution for computer vision and multi-touch sensing. It takes an video input stream and outputs tracking data (e.g. coordinates and blob size) and touch events (e.g. 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 enabled applications and supports many multi-touch lighting techniques including: FTIR, DI, DSI, and LLP with expansion planned for the future (custom modules/filters).

Programming Language: C++

License: MPL or MIT (not defined)

Link: http://tbeta.nuigroup.com

Touché

Touché is a free, open-source tracking environment for optical multitouch tables. It has been written for MacOS X Leopard and uses many of its core technologies, such as QuickTime, Core Animation, Core Image and the Accelerate framework, but also high-quality open-source libraries such as libdc1394 and OpenCV, in order to achieve good tracking performance.

Programming Language: Cocoa (Mac)

License: LGPLv3

Link: http://gkaindl.com/software/touche

http://code.google.com/p/touche/

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'. It includes a configuration app and a few demos to get you started, and will interace with most types of webcams and video capture devices. It currently works only under Windows but efforts are being made to port it to other platforms.Software & Applications 61

Programming Language: C++

License: New BSD License

Link: http://nuigroup.com/touchlib/

http://code.google.com/p/touchlib/

3.5 Gateway Applications: -

Multi-Touch Vista

Multi-Touch Vista is a user input management layer that handles input from various devices (touchlib, multiple mice, wii remotes etc.) and normalises it against the scale and rotation of the target window. It will allow standard applications to be scaled and rotated in a multi-touch style and receive standardised input. It will also provide a framework on which to build multi-input WPF applications. It supports Windows Vista and XP.

Programming Language: C#

License: GNU General Public License (GPL) v2

Link: http://www.codeplex.com/MultiTouchVista

PyMT

PyMT is a python module for developing multi-touch enabled media rich OpenGL applications based on pyglet. Currently the aim is to allow for quick and easy interaction design and rapid prototype development. There is also a focus on logging tasks or sessions of user interaction to quantitative data and the analysis/ visualization of such data.

Programming Language: Python

License: GPL v3

Link: http://pymt.txzone.net

TouchPy

TouchPy is a bare bones light weight Python Multi-Touch framework that does not bind you to any specific GUI Toolkit. It is simple to use, and is the most versatile Python Multi-Touch framework.

Programming Language: Python

License: GPLSoftware & Applications 63

Link: http://touchpy.googlecode.com

Hardware Setup

Since flat surface can be converted into touchscreen in many ways, some of the ways are given below.

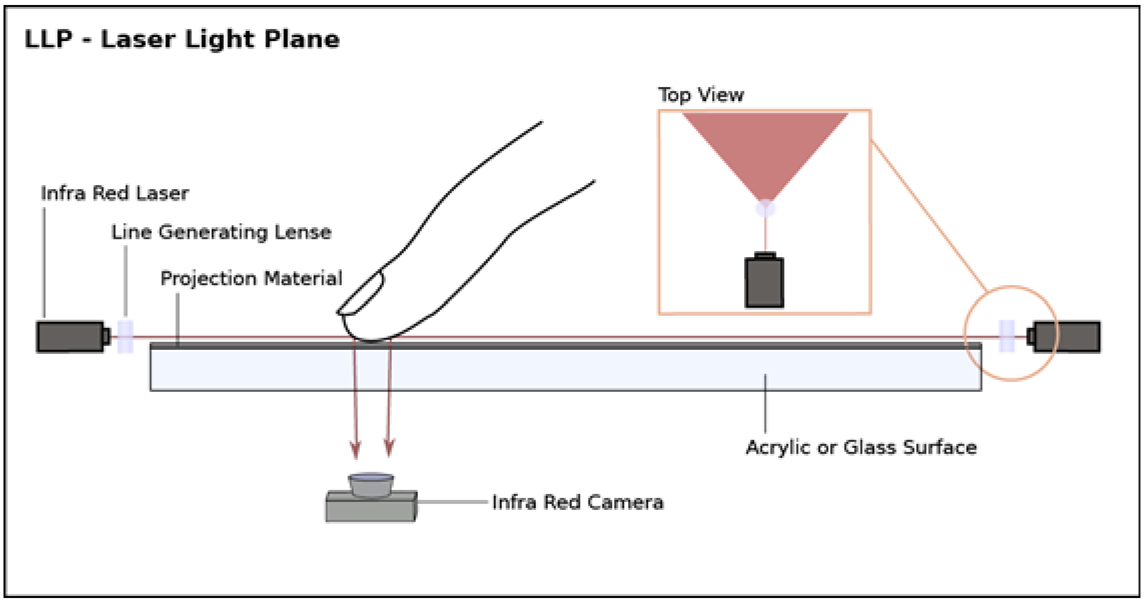
4.1 Building an LLP: -

setup LLP (Laser Light Plane) multi-touch varies from other technologies mentioned in this book in the sense that instead of using LEDs it uses lasers. These lasers are used to create a plane of infrared light above the surface, instead of inside it (like FTIR), or from below it (like DI). In this sense it can be seen as similar to LED-LP. LLP is known for its easy of setup and amazing (very high contrast) blobs with very little effort.

Step 1: Materials Needed

- Clear Surface (Acrylic, Glass, Air)
- IR Goggles
- Infrared (780 to 880nm) Lasers with Line Lenses
- Projection Surface/LCD Monitor
- Projector (Not necessary if using LCD Monitor)
- Camera with IR Bandpass Filter to match lasers

As shown in the diagram, all the IR light in an LLP setup is above the surface, as close to the surface as can be. When an object obstructs this plane, the light is scattered downward and picked up by the camera.



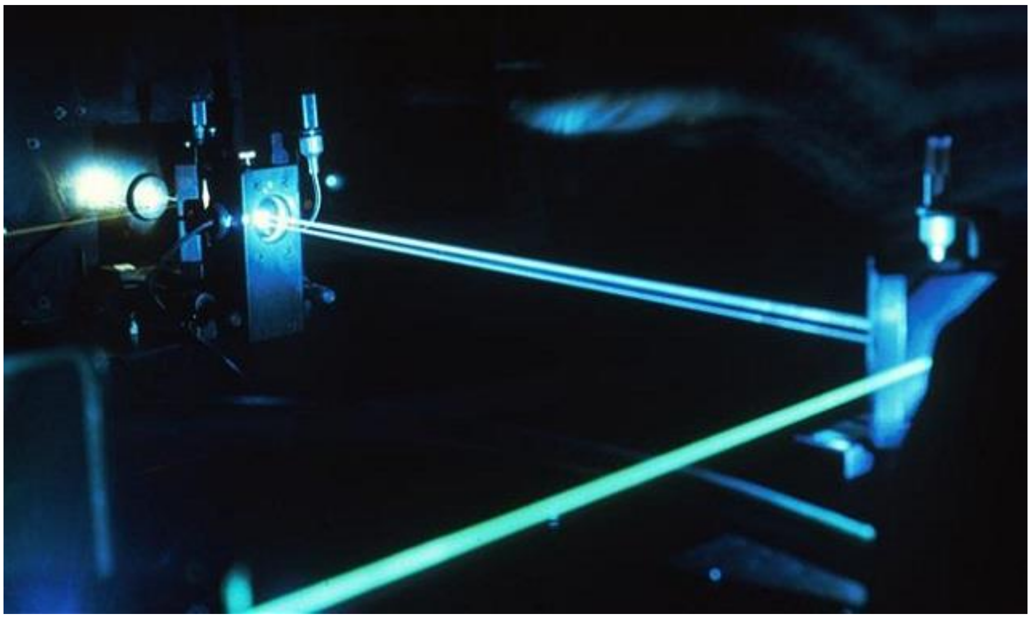
Step 2: Lasers

Obviously if one took a standard laser and stuck it over a glass surface it would not work very well. What would result is a single-touch single-dimension slider, like a fader on an audio interface. There are two reasons for this: First, the touch closest to the laser would cause total occlusion, and any objects past that object would have no light to scatter downward. Finger one is scattering all the laser light down towards the camera, and not leaving any for Finger two. Finger one would show up as a brilliant blob, however Finger two would not show up at all. The obvious solution to this is to add another laser. While then would only allow for two touches, we can ignore this issue because when we move to 2D touching this becomes a non-issue. The most obvious problem to this setup is that it is single dimensional. In order to make this sense touches in the x and y plane we need a plane of light. To get this plane of light we use an optical lens called a Maddox Rod. A Maddox rod is a bunch of little semicircles lined up, and when a line of light enters it gets split up into a plane on light. From now on the Maddox Rod will be referred to as a "Line Lens", as that is the common term used around NUI Group. The illustration above shows what a normal laser beam looks like, the type used in our example above. The next illustration shows a laser with the line lens attached. This laser is generating a plane of light, and we are only viewing one slice of it. It would be impossible for us to view the whole plane with current technology. This is the kind of laser that is used in an LLP multi-touch setup.

Step 3: Selecting Lasers

There are many different kinds and specifications of lasers, and it is important to get the right for your setup. Like anything else, higher quality lasers cost more money. There are four key things to consider when buying lasers.

- Number of lasers being used
- Power of Lasers (in Milliwatts) being used
- Wavelength of light being emitted by lasers
- Angle of Line Lens



Step 4: Getting a Power Supply

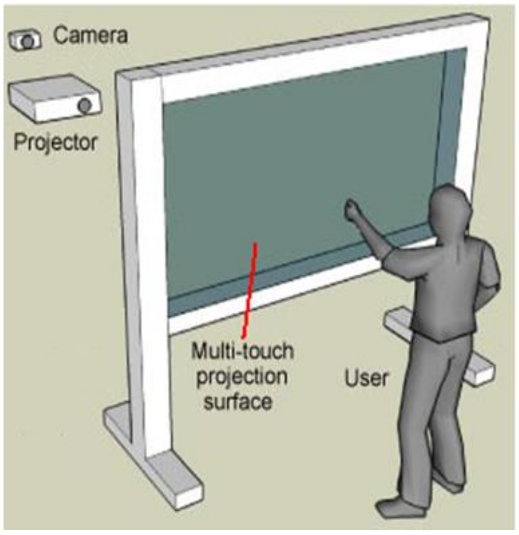
You will need a way to power these lasers. The first thing is voltage. Most lasers are either 3.3V or 5V. Your power supply must match the voltage of your lasers, or you risk damaging your lasers. Running a 5V laser at 3.3 volts probably won't hurt it, it just won't turn on, and if it does it will not be bright. Running a 3.3V laser at 5V is almost guaranteed to fry it.

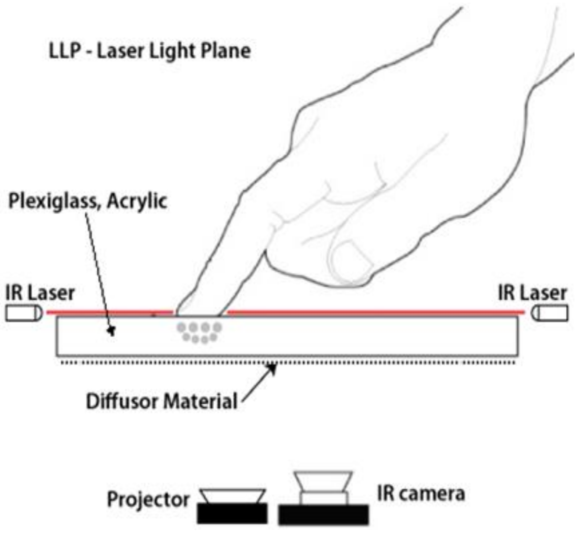
Step 5: Attaching the line lens and focusing

The instructions in this part may differ, but the general idea should be the same. There should be three parts to your lasers, the Line Lens, Line Lens Mount, and the laser itself. Start by taking the line lens and placing it in the mount. The ridged side should be pointing towards the laser diode. If it is backwards this will become apparent when focused, the line will be very wavy and choppy. Next unscrew the black part from the laser completely, and screw it into the Line Lens mount until its snug.

Step 6: Mounting and aligning lasers

There is no recommended or specific way that lasers should be mounted, its very setup-specific. People have done everything from duct tape to custom designed and fabricated mounts that allow for milli-meter accuracy when adjusting. Mounting options very much depend on the setup, and are not within the scope of this tutorial. Aligning the lasers is a different issue. The Line-Lens mount can be twisted about 180 degrees in either direction without losing focus, so that makes it very easy to line up. Once they are attached, fold a piece of paper in half so there is a very flat edge protruding from your surface to compare with. Power your lasers on, and go through each of them holding a digital camera up so the beam is visible, and twist the line lens until its flat. You also want to get the laser plane as close to the touch surface as possible, so it doesn't sense a blob before the object comes in contact with the surface. This can be achieved in most cases by lifting up the back of the laser with small pieces of paper under it, or you may search for a more permanent solution.



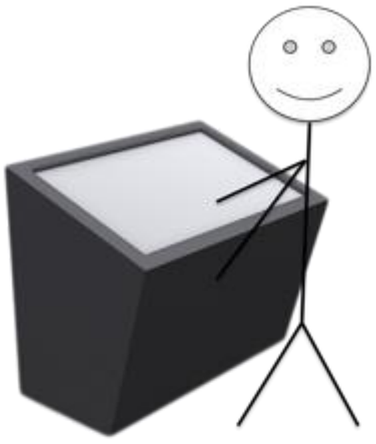


4.2 Building a Rear-DI Setup

Infrared light is shined at the screen from below the touch surface. A diffuser is placed on the bottom of the touch surface so that when an object touches the surface it reflects more light than the diffuser or objects in the background. This extra light is sensed by an infrared camera and (using tracking software) is filtered into ‘blobs’ which are then converted into (x,y) coordinates Below is a picture illustrating the effect of diffused illumination.

Step 1: Building the Box

The first step in making a Diffused Illumination (DI) Multi-touch table is to build the enclosure. DI setups require a mostly closed box so that no infrared light escapes. Were light to escape, there would be no uniform infrared brightness within the box, effectively rendering the effect useless. Below is a basic 3D model created using Google SketchUp of the ORION Multi-touch Table. The box is built out of 12mm craft wood/medium-density fibre board (MDF). There is a section on the top of the box so that a small mirror could be mounted to increase projection throw. This section is also designed to house some small speakers and possibly an external keyboard. The glass sheet pictured here was later replaced with a sheet of 4mm thick glass, 71cmx56cm frosted (sandblasted) on one side for the diffuser/projection surface.



Step 2: Projector

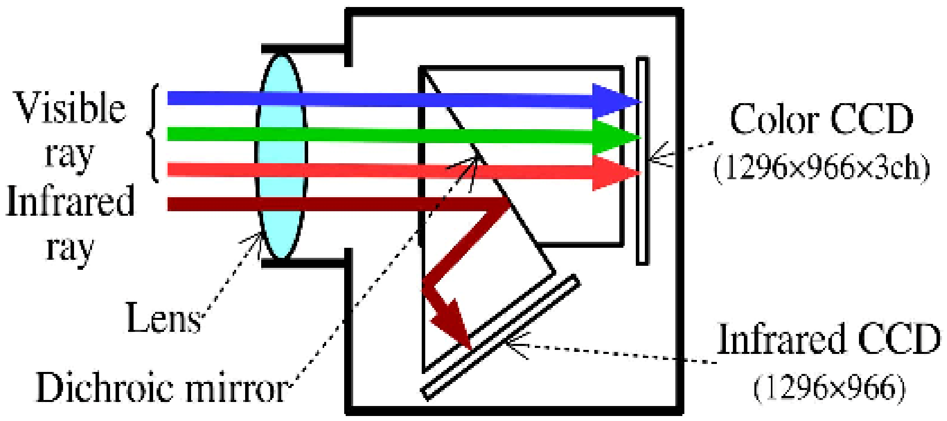
For the projection a SHARP DLP projector Model: PG-M15S is being used.

Step 3: Webcam

The camera used in this setup is an Xbox Vision 360 USB camera that can be used as a webcam on a regular PC. The resolution on the tracking software is 320x240 at 60fps. This camera was fairly easy to modify (removal of the IR filter), and was a cheap and easy option at the time for a reasonably good multi-touch camera.

Step 4: Infrared Illuminators

In the ORION mt there are three illuminators. A diagram illustrating the layout of the illuminators (1x 140 IR LEDs, 1x 80 IR LEDs and 1x 45 LEDs) is below.. The border represents the physical screen projection size in relation to the illuminators and the box. Because of the narrow beam of IR illuminator, there is a large hotspot on the glass. To get around this, the 140 IR Illuminator is angled so that the light is emitted and reflected off the side wall of the box.

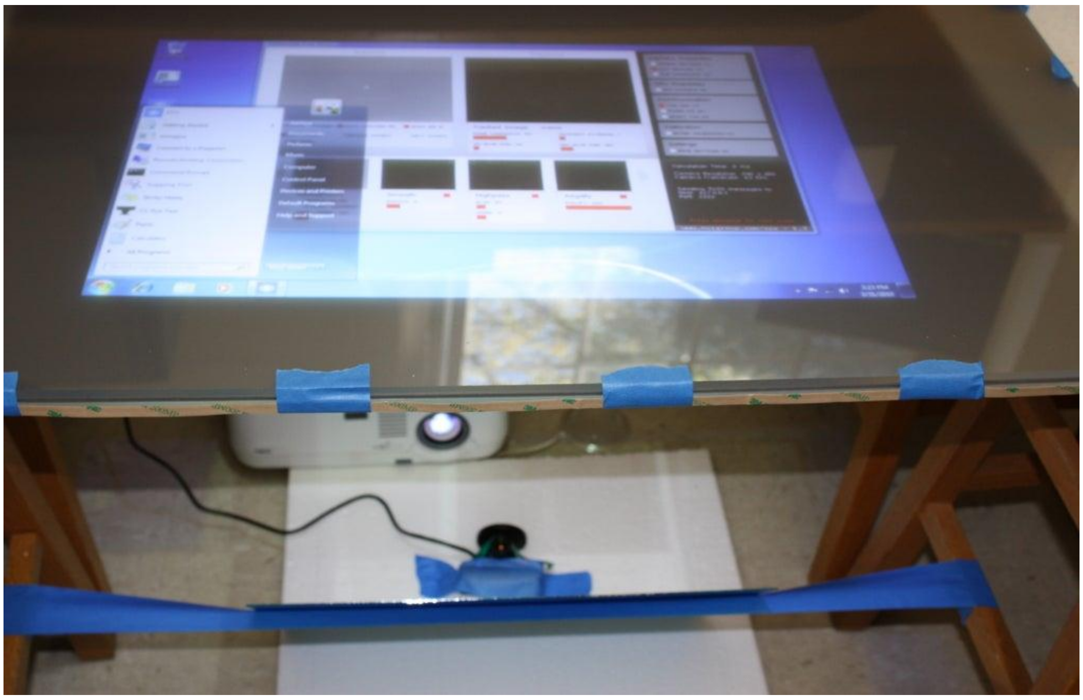


Step 5: Webcam and Projector Placement

Camera, mirror, and projector placements are important issues to watch for to ensure that the webcam can view the entire touch surface and the projector can project on the entire surface. Originally, there was a camera angle issue in the OrionMT where the camera could not capture the entire screen when placed in the middle of the floor. To overcome this, the cameras was moved to reflect of the 2nd mirror. That way, the entire screen can be captured (with overscan).

Step 6: Touch Surface

We need a thick enough surface to not flex when pressed (to avoid short-term hotspots becoming blobs) and some kind of surface for projection and diffusion. People have used vellum, tracing paper, fabric interfacing, and even cheap plastic tablecloths or shower curtains (in my case, what worked best) on top of the surface, in addition to frosted glass or acrylic. Before purchasing a frosted material and/or diffuser it’s wise to test the surface’s suitability both as a projection surface and as an IR diffuser.



Step 7: The Finished Product

The next step was to test the setup using the tracking software. If we look closely at the screenshot above we can just make out the 3 glowing sections where the 3 illuminators were placed from the raw camera feed.

4.3 Building an Interactive Floor: -

This chapter explains how to build an interactive floor. This floor differs mainly in the fact that multiple people can walk over and interact with the same floor space as opposed to single person interaction of regular interactive floors. A typical setup would only cover a part of the floor, i.e. a lift lobby, or right at the entrance door, because covering an entire lobby - although possible - is unpractical. Note that this should not be attempted as your first multitouch project. The reason for this is that when you build the interactive floor, you will then have the insight needed to identify problems before you go to far into the project. Depending on the setup of the room and people installing equipment in the roof, this project will take 4-8 hours. The money spent is extremely dependent on the size of the room and optional parts required. No fixed figure can be given.

The tools required for every interactive floor differs. This list is only a guide to what tools we would generally need.

• Projector: Try keeping the projector as light as possible. Also use a wide-angle lens for the projector – or a short-throw projector – to ensure the entire floor gets covered.

• Camera: Use a camera with a wide-angle lens to ensure the entire projected image gets covered.

• Hammer and nails or screwdriver and screws: Depending on how you plan to install the camera into the roof

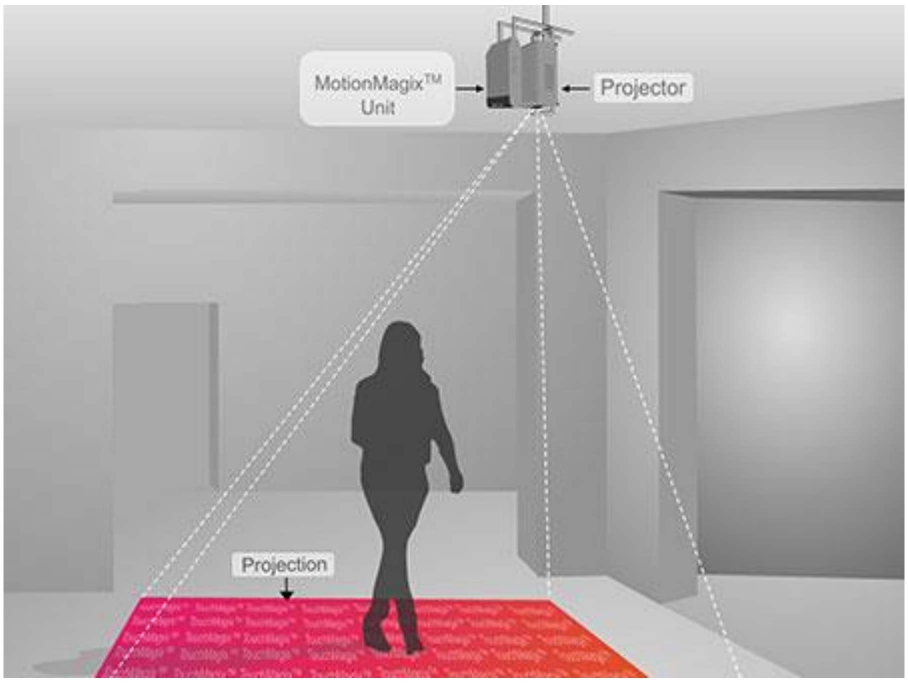
• Projector mount: As light as possible, but strong as well.

• Mirror: Any will do.

• Mirror mount: Usually a metal arm that can be attached to any surface.

Step 1: Mount the projector

First, we need to make a hole in the roof for the projector lens to fit through. Detailed intructions on this cannot be given because of the vast number of roof types. The most important point for this step is that the projector should be secured 100% in the middle of the roof. Also ensure that the required area is filled by the projection.

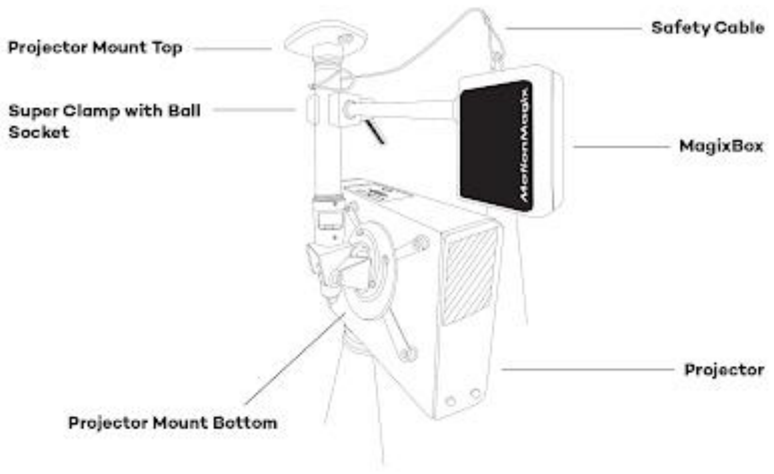


Step 2: Install the camera

A CCTV-like camera will be most preferred due to the fact that they come standard with roof mounts. If room does have a roof mount, use nails or screws to secure it to the roof. If camera does not have a roof mount, we will need to buy an extra camera mount and install it as per the mount’s instructions.

Step 4: Testing

At this point we should have the installation done and ready for testing. How we plug the projector and camera into the computer is up to us. We will most probably need extension cables for the projector and camera. Install the software of choice and test the setup, adjust as needed.



Interactive Floor Setup with Clamp

Software Used

In order to calibrate the system, we can use touchlib or Community Core Vision (CCV). Any tracking software that supports the filters needed for diffused illumination will also work. We need to set our software to only detect large blobs. People moving on the floor will then be picked up optimally. A midrange rectify and midrange blur is suggested. When setting up the software, ensure that you setup and test it in varying light conditions. This will greatly affect the setup. The recommended conditions are during the day, around 12:00, not overcast (if it can be avoided). This will give the greatest accuracy in how the floor will work on a day-to-day basis. The following are the steps on how to calibrate the floor using CCV or Touchlib.

1. Start CCV or Touchlib

2. Get a flashlight

3. Shine the flashlight on the floor and calibrate the filter of CCV or Touchlib to only see the flashlight. In other words, set rectify to a rather high amount.

4. Turn of the flashlight, recapture background (press ‘b’) in CCV

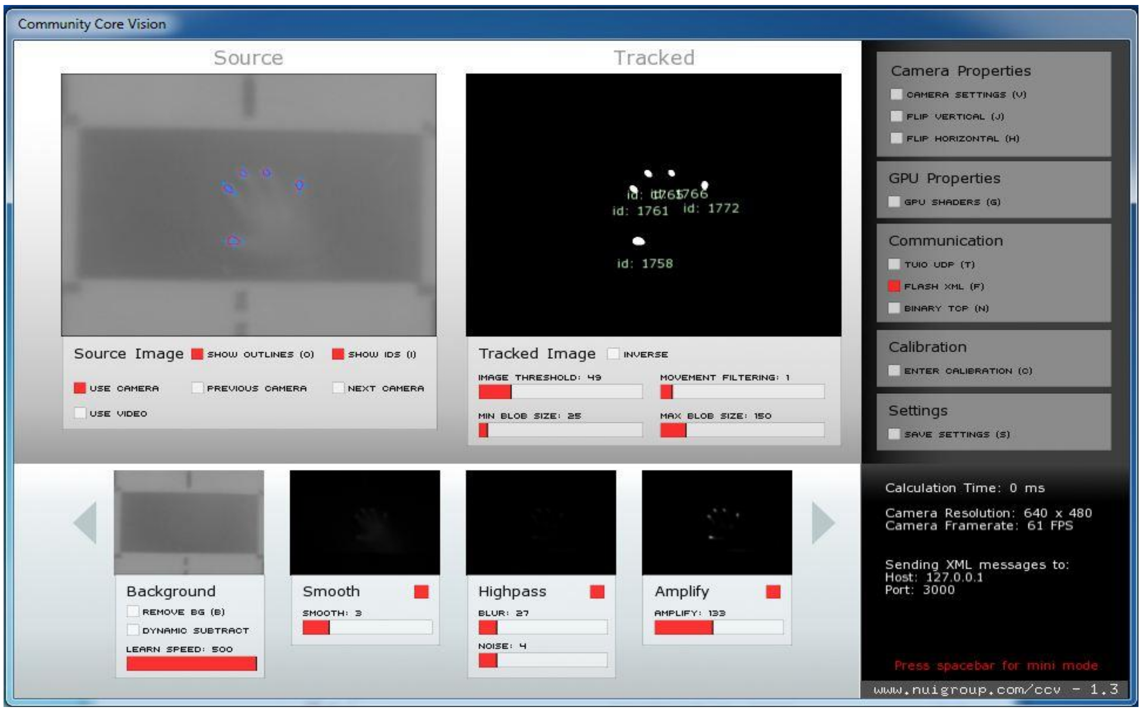
5. Press ‘c’ to bring up the calibration, then hit ‘c’ again to startthe calibration.

6. Aim the flashlight at the green cross with the red circle.

7. Turn the flashlight on and turn it off again as quick as you can. It should be as a quick flash. CCV or touchlib should have picked it up as if it was a touch to a normal screen.

8. Repeat steps 6+7 until calibration is complete for all the crosses.

9. We should now have a quite accurate calibration.


Community Core Vision

Source

Tracked

id: 1765 id: 1766
id: 1761 id: 1772
id: 1758

Source Image 🟥 SHOW OUTLINES (O) 🟥 SHOW IDS (I)
🟥 USE CAMERA ☐ PREVIOUS CAMERA ☐ NEXT CAMERA
☐ USE VIDEO

Tracked Image ☐ INVERSE
IMAGE THRESHOLD: 49
MOVEMENT FILTERING: 1
MIN BLOB SIZE: 25
MAX BLOB SIZE: 150

Camera Properties
☐ CAMERA SETTINGS (V)
☐ FLIP VERTICAL (J)
☐ FLIP HORIZONTAL (H)

GPU Properties
☐ GPU SHADERS (G)

Communication
☐ TUIO UDP (T)
🟥 FLASH XML (F)
☐ BINARY TCP (N)

Calibration
☐ ENTER CALIBRATION (C)

Settings
☐ SAVE SETTINGS (S)

Background
☐ REMOVE BG (B)
☐ DYNAMIC SUBTRACT
LEARN SPEED: 500

Smooth 🟥
SMOOTH: 3

Highpass 🟥
BLUR: 27
NOISE: 4

Amplify 🟥
AMPLIFY: 133

```
Calculation Time: 0 ms

Camera Resolution: 640 x 480
Camera Framerate: 61 FPS

Sending XML messages to:
Host: 127.0.0.1
Port: 3000
```

Press spacebar for mini mode

www.nuigroup.com/ccv - 1.3

Things to keep in mind while calibrating: Try to stand clear of the camera’s view area, this would interfere with the calibration. Try to get the flashlight as close as possible to the camera. In other words, try to keep the light from the flashlight as close to a circle as you can. If you stand at a very low angle compared to the camera you’ll see the flashlight making an elliptical light shape and thus making the calibration less accurate and more difficult to get right. If you can’t get very close to the camera, calibration is still perfectly possible, it will just take some extra time.

ONGOING RESEARCHES / FUTURE SCOPES

In this section we highlight ongoing work and useful websites and communities where users can find additional information. This collection of links is unsorted and in no particular order – our selection does not attempt include all webpages describing self-built multi-touch surfaces.

Microsoft Surface

The Microsoft Surface, is the multitouch product from Microsoft which is developed as a software and hardware combination technology that allows a user, or multiple users, to manipulate digital content by the use of natural motions, hand gestures, or physical objects. More information, videos and gallery available on their webpage http://www.microsoft.com/surface.

The setup is a slightly modified DI setup and explained on the (non-technical) website. With Microsoft Touch Wall and the Microsoft Touch Sphere Microsoft recently presents two new interactive surfaces. Touch-Light http://www.youtube.com/watch?v=F1br3fDZyUQ

Lucid-Touch is also an interesting project done by Microsoft research.

c-base MultiTouchConsole MTC

The c-base MultiTouchConsole MTC is a DI multi-touch table that was first presented by a group centered around Ulrich von Zadow at the 23rd Chaos Communication Congress in 2006. It is based on the open-source library libavg and served as a test-bed for it‘s the multi-touch features. The hardware setup and material needed to build a similar table are well-documented (Web page, german video), as is the libavg tracking setup. The c-base is a non-profit association in Berlin, Germany that provides an open space for technological experimentation. The MTC has accordingly been used as a basis for interaction experiments such as Jens Wunderling‘s LoopArena sequencer and sponc, a pong-like game, created by Martin Heisterman. The Archimedes MTT is a commercial version of the MTC14. It is sold as a complete self-contained table with high-quality 60 Hz tracking and 3500 Lumen highcontrast projection.

Pascal Schmidt German “Jugend forscht”

Pascal Schmidt, a student at a Germa high school (18), won the technical & scientific youth-contest in Germany 2008 with his home grown DI multi-touch table. The prototype was presented together with a company called mehrwert at CeBit 2008 in Early 2008. The web pages http://www.multitouch.sourceforge.net and http://www.mehrwert.cc/ containing more information on this project.

NUI Group

The Nui group – or Natural User Interface group – is one of the most active groups dealing with multi-touch interaction and interactive media. Their goal is it to create open-source machine sensing techniques which will benefit artistic and educational applications. Their focus is on "Open-Source Interface", which is solely for accelerating development of existing hardware and sensing solutions. On their webpage http://www.nuigroup.com they provide a wiki and a active forum. TouchLib, TouchEvent http://www.touchevent.riaforge.org and OpenTouch http://www.opentouch.info are their three main projects.

TISCH

The central element of TISCH is a multi-touch table (TISCH: 1) that provides room for 4 to 6 concurrent users. A frosted glass table of about 1.10 x 0.7 m is used as a back-projection surface. It is mounted on a robust aluminium frame which contains a projector, an infrared camera and a computer. An acrylic glass sheet placed on top of the projection area has 70 infrared LEDs (Osram SFH4250, pulsed at 2.4 Volts) attached around its rim to provide FTIR-based multitouch input to the computer via an IR camera.

Conclusion

This setup is fairly simple. Literally every setup differs so we need to play around with most of the steps to find what works best for our setup. The basic concepts covered here will always apply.

It is advised to have some multitouch experience before building this. That would help us identify problematic areas on the floor before we start, and will save time and money.

In summary, the real issue faced by touch projector technology is not whether it is completely good or bad but whether the device features is comprehensive enough to meet user's expectation whether in business, education or entertainment.

References

https://www.firstpost.com/tech/news-analysis/how-to-convert-any-surface-into-a-touchscreen-with-a-wiimote-3589215.html

https://www.udemy.com/course/how-to-turn-any-surface-into-touch-surface/

https://www.semanticscholar.org/paper/Interactive-touch-board-using-IR-camera-Ramsundar/b8b9429d2ae8748ef2dac000b9112da5f7360e42

https://en.wikipedia.org/wiki/Multi-touch

https://en.wikipedia.org/wiki/Infrared#:~:text=Infrared%20radiation%20extends%20from%20the,portion%20of%20the%20electromagnetic%20spectrum

https://www.semanticscholar.org/paper/Depth-Estimation-Using-an-Infrared-Dot-Projector-an-Hisatomi-Kano/19625a9d38251a9b9175013148cf501114212228

https://www.researchgate.net/publication/243490293_Frustrated_total_internal_reflection_A_demonstration_and_review/link/577e629008aeaee3b28351f3/download

https://www.dominikschmidt.net/2010/11/ftir-compliant-surface-tutorial/

https://sethsandler.com/multitouch/frontdi/

https://sethsandler.com/multitouch/reardi/

https://sethsandler.com/multitouch/llp/

https://sethsandler.com/multitouch/dsi/

https://wikiepedia.wordpress.com/2016/01/18/structural-components-of-led-panel-lights/

https://developer.android.com/training/gestures/movement

https://github.com/topics/touchscreen?l=python

https://www.youtube.com/watch?v=Wyv5TnkFuxE

https://dokumen.tips/documents/first-edition-community-release.html

http://pymt.txzone.net

http://mcsp.wartburg.edu/zelle/python/python-first.html

http://code.google.com/p/touchpy/

http://www.youtube.com/watch?v=bEf3nGjOgpU

http://www.libavg.de/

http://code.google.com/p/pytuio/

http://www.tuio.org/

http://xelapondsstuff.wordpress.com/

http://cs.uiowa.edu/tehansen

http://www.pyglet.org

http://www.opengl.org/

http://en.wikipedia.org/wiki/OpenGL

http://nehe.gamedev.net/

http://en.wikipedia.org/wiki/GUI_widget

http://pymt.txzone.net/docs/api/

books

[1] Boring, S, Baur, D., Butz, A., Gustafson, S., & Baudisch P. (2010). Touch projector: mobile interaction

[2] Cai, X., Xie, X., Li, G., Song, W., Zheng, Y., & Wang, Z., (2014). A New Method of Detecting Fingertip Touch for The Projector – Camera HCI system.

[3] Cauchard, J.R., Fraser, M., Han, T., & Subramanian, S. (n.d.). Designing mobile projectors to support interactivity.

[4] Johannes Schoning, Peter Brandl, Florian Daiber, Patrick Olivier, Tim Roth, Ulrich von Zadow. Technical Report TUM-I0833, Multi-Touch Surfaces: A Technical Guide.