The Pennsylvania State University

The Graduate School

College of Engineering

DEVELOPMENT AND INTEGRATION OF AN INFRA-RED BASED PLATFORM INDEPENDENT PROJECTABLE TOUCH SCREEN

A Thesis in

Electrical Engineering

by

Atriya Ghosh

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

May 2015

The thesis of Atriya Ghosh was reviewed and approved* by the following:

Zhiwen Liu
Professor of Electrical Engineering
Thesis Advisor

Iam-Choon Khoo
William E. Leonhard Professor of Electrical Engineering

Kultegin Aydin
Professor of Electrical Engineering
Head of the Department of Electrical Engineering

*Signatures are on file in the Graduate School.

ABSTRACT

While giving a presentation, or while sharing contents of a phone or laptop, it is always a cause of inconvenience for the user to interact with the said device while the screen is being projected, or cast to an external, non-interactive, screen. Through this research I attempt to create a smart touch screen. The user is able to interact with the projected screen instead of the device, and it is thus more convenient and ergonomic. My technology uses an infra-red camera and a touch activated stylus to convert any screen into a touch screen. The screen might be one being projected by a projector, being cast onto a television or an external screen. The user can fully control the device just by interacting with the projected screen. This will also convert a laptop, which is not a touch screen one, to a fully touch controlled laptop. So, we have an interactive whiteboard, a powerful sharing tool and a multi-touch enabled screen at a very nominal cost. This technology is also independent of the surface being projected on, the projection distance and size.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

My research was supported by the Department of Electrical Engineering at the College of Engineering. I would like to acknowledge the contribution my thesis advisor, Dr. Zhiwen Liu, who supported me at all stages of development of the technology, as well as teaching me the concepts of light propagation and detection, which helped me a considerably in this development of this technology. He supported my idea and gave provided me with all the guidelines I needed to complete this research. My research group at the ultra-fast and non-linear optics laboratory also provided considerable support and insight.

I would also like to thank the people who helped me in the Android developers’ forum, who answered a lot of my questions, developers of the Smoothboard© software and Google Chromecast©. Finally I would like to acknowledge the work of Dr. Johnny Lee. My work is an extension of the work he did on WiiMote projects.

CHAPTER 1: INTRODUCTION

An interactive touch-screen is synonymous with almost every technology nowadays. For my thesis I designed a system that would be able to convert any projected screen into a touch screen. My focus was mainly on the screen of a smart-phone, or more specifically, an Android smartphone. With this technology, we were able to project or cast the screen of the phone onto any surface, and then interact with the projected screen to control the entire phone. Though originally conceptualized to work only with Android devices, I was able to extend this technology to be platform independent. As a result, this technology will work on a varied range of devices running different operating systems. For my research, I was successfully able to demonstrate it working on Windows and Android platforms.

The main function of this technology is to convert any screen into a touch-screen. The screen can be one projected by a projector, a screen mirrored onto an external monitor, cast, wirelessly or otherwise, onto a television or similar projections. It can also make any laptop screen a touch screen. This technology supports the multi-touch function, and can detect up to four simultaneous touches. It is ideally independent of the surface being projected on and the size or distance of the projection. These parameters, though, are affected by the quality of hardware used. When used, the user can control the entire device only by interacting with the projected screen.

My technology uses an infra-red camera and a touch-activated stylus. The stylus has an infra-red LED* at the tip, which emits a flash of light whenever it touches any surface [18]. This setup is considerably simple to facilitate and is cost effective, especially when compared to the commercially available interactive smartboards. Below I explain the setup and the software that is used in this technology.

*LED stands for light emitting diode. From now we will be using this acronym.

CHAPTER 2: THE APPROACH

The approach to this software is pretty simple to look at. Below I explain the setup and the software that is used in this technology. Also, I separately explain how the software functions differently for Windows and Android.

2.1 The Setup

The complete setup is shown in figure 2.1. We project the relevant screen of a device on an external medium, either by using a projector or simply onto an external monitor. Then we setup an infra-red camera to look at the screen. We have to keep in mind that different cameras have different fields of vision. For the camera I have used here, a Nintendo wiimote, the field of vision is about forty-five degrees [2]. So if we draw a normal from the center of the screen projecting outwards, the camera can be placed on either side of that normal as long as the angle between the normal and the line drawn from the camera lens to the center of the screen does not exceed forty-five degrees. If we exceed that angle, the camera will not be able to see the entire screen and we lose tracking resolution [1]. Also, the camera has to be connected to the device that is projecting the screen. Here, we connect the camera using wirelessly using Bluetooth, which gives us the freedom to move it around freely.

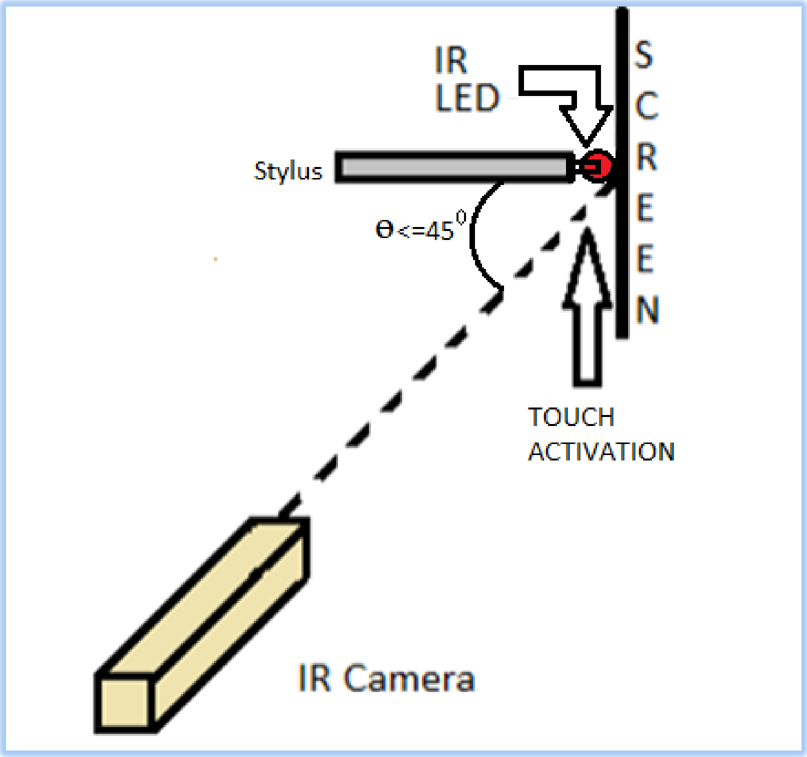


Figure 2.1: The Setup

Once the camera is setup, we touch the projected screen using the stylus, and the infra-red LED attached to the tip of the stylus flashes and sends with every touch. This infra-red flash is then detected by the camera and sent as an input to the device. The remaining task is achieved by the software, which is explained in the subsequent sections.

2.2 The software

The software is at the core of this technology. It's called *Project Touch* and should be installed in the device for the technology to work. The coding being done in JAVA, the software is considerably platform independent and, with minor modifications to the source code, work on a wide variety of operating systems. It is robust and platform independent in most cases. Like mentioned before, for this project I have demonstrated it working on Windows and Android operating systems.

When we first start the program, it establishes the connection to the camera and configures itself to receive inputs from it. It is not aware in anyway of the screen being projected. After a connection to the camera has been, it asks the user to calibrate the touch. This is done by projecting four targets, one by one, at four corners of the projected screen, as shown in figure 2.2. As the user touches each target, the software records its position relative to the field of view of the camera [1, 2]. When the user has touched all four points at the four corners one by one, the software records their position, and virtually joins the four points together to form the screen. This is how the software becomes aware of the exact position of the screen within the field of vision of the camera, and also triangulates the three-dimensional angle at which the camera is looking at the screen. We should keep in mind that it is up to the user to place the camera at a location from where the camera will be able to view the entire projected screen, else the calibration will not be successful. Also, for the best tracking resolution, the user should ensure that the projected screen covers as much as possible of the entire view of the camera. Once the calibration is complete and successful, the user will be able to see the exact orientation and size of the screen as seen through the camera. If the user is not satisfied with it, or the calibration is not successful, the user can adjust the location of the camera and re-calibrate again.

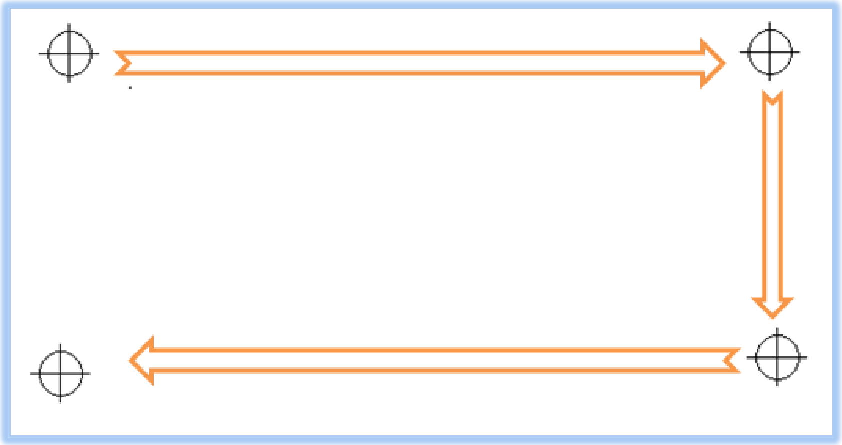


Figure 2.2: The Calibration Screen

Once the calibration is successful and to the liking of the user, the touch screen becomes fully operational. When the user touches any point of the projected screen, the stylus emits an infra-red flash, which is picked up by the camera, and sent as an input to the software. The software then triangulates the position of this touch in reference to the four pre-calibrated points, as shown in figure 2.3, and thus identifies its position on the screen. So, now the software knows the exact point on the screen the user has touched. Now, as its output, the software ensures that the relevant action pertaining to a touch at that point is performed. This is different for different operating systems. The output of the software for Windows [2] and Android systems are explained in detail in the subsequent sections.

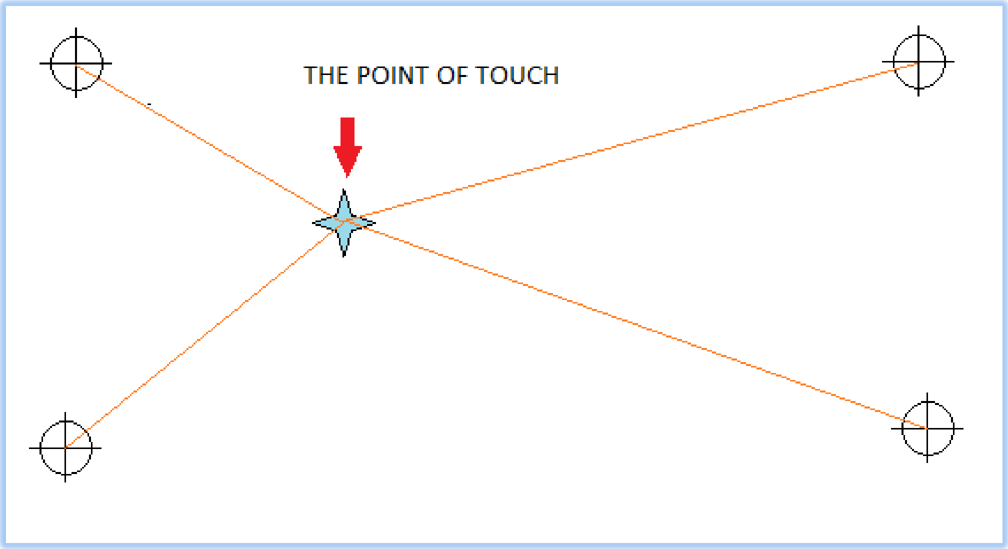


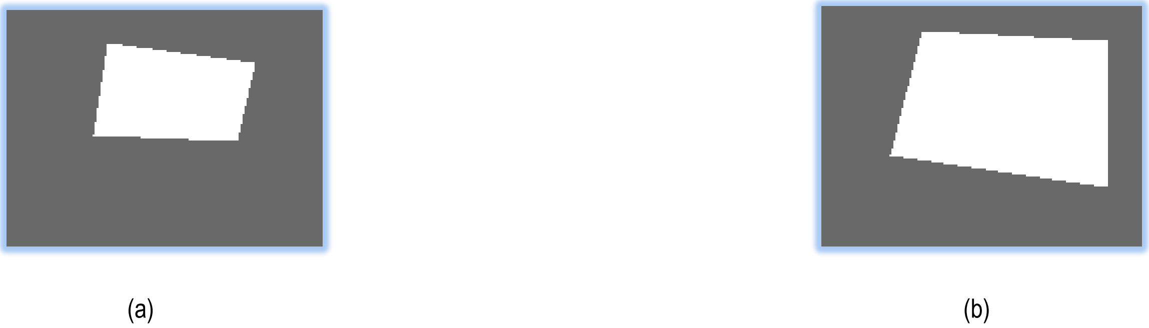
Figure 2.3: Touch Sensing

CHAPTER 3: THE ENTIRE SYSTEM

Any system consists of three basic parts, the input, the abstracted calculations inside the system, the output. In this section I attempt to detail each part separately, braking each section down step-by-step.

3.1 The Input

The input for this system comes from the infra-red camera that is connected to the device (Laptop or Smartphone) using Bluetooth. Like detailed before, first we calibrate its field of view, and define the area covered by the screen in its field of view using the four point touch calibration by touching the four corners of the screen. After the calibration, Figure 3.1 shows what we can see if we look through the camera.



(a) (b)

Figure 3.1 Looking through the camera: (a) low-sensitivity input (b) High-sensitivity input.

Figure 3.1 shows two incidents of what we see when we look through the camera. The grey area is the entire field of view of the camera, and the white rear inside it is what the camera recognizes as the projected screen based on the four point touch calibration. Looking at figure 3.1(a), we see that the screen covers a small area of the entire field of view. This causes a loss in sensitivity of the touch screen and a loss in tracking resolution. In Figure 3.1(b), the screen covers a larger area, and facilitates greater sensitivity and tracking resolution. In general, during calibration, we should always attempt to make sure the screen covers as much as possible of the field of view.

Now, after calibration, when the user touches a point on the screen with the stylus, the infra-red LED flashes with each touch. This flash is recorded by the camera as long as it is inside the projected screen. This flash, superimposed onto the screen, serves as an input for the system. Figure 3.2 shows the image that is provided as an input from the camera to the software in the device.

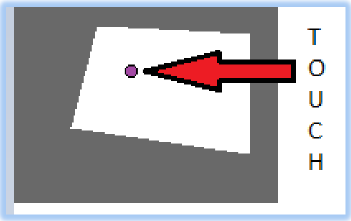


Figure 3.2 the Final Input

3.2 Inside the System

Like stated above, the input to the system is the image sent to the software by the infra-red camera. Once the software gets the image, it superposes and X-Y grid on the screen. Like mentioned before, during the touch calibration we touch the four corners of the screen one by one to define the screen for the system. The Y-axis is formed by drawing a straight line through the first and the last point touched during the touch calibration. The X-axis is formed by drawing a straight line through the last and second-last (third) point touched during the touch calibration. Then the position of the touch is defined by identifying the X and Y values of the position at which the camera recorded the flash of the infra-red LED. The X and Y values are then placed onto a 2X1 matrix, and this Matrix serves as the output of the software. Figure 3.3 shows a figurative representation of this procedure. The software then sends this output to the operating system, which is discussed below.

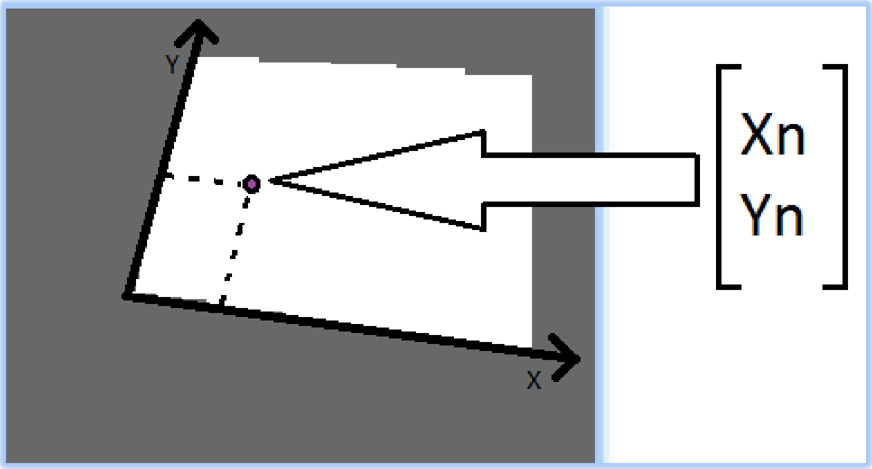


Figure 3.3: Recognizing the position of the touch

3.3 The Output

Like stated above the 2X1 matrix, that holds the X and Y co-ordinates of the point where the user has touched on the screen, is the output of the software. This output is then sent to the operating system. This part differs slightly based on the device being used for this purpose. These devices can be broken down into two wide categories:

- Devices using a mouse pointer
- Devices using a capacitive touch screen.

I have discussed a breakdown of how we come to the final output for each one of these categories separately below.

3.3.1 For Windows (or devices using a mouse pointer)

I have used a Windows© laptop running Windows 8.1 operating system to demonstrate the output of the system. But, since my codes are written in JAVA, which is platform independent coding language, they should work on any device running operating systems which use the mouse pointer, like Ubuntu or Macintosh.

Like stated above the output of the software is a 2X1 matrix that holds the X and Y co-ordinates of the point where the user has touched on the screen. This output is then sent as an input to the windows operating system, more specifically the pointer position handling chip of windows. Thus, this matrix mimics the controller input that provides to Windows the position where a mouse click occurs. So Windows reads this touch as a mouse left-click occurring at that position, and carries out the relevant function that a mouse click at that position would bring about [1, 2]. This completes the entire system. So, to summarize, **a**

touch with the LED stylus on the projected screen brings about the same function as a mouse left-click at that position on the actual screen would achieve. Figure 3.4 shows a diagrammatic representation for such a system.

Also, it's worth mentioning here that any device using a pointer also accepts an input corresponding to a right-click with the mouse. To right about the recognition of a mouse right-click, the user must touch and hold the LED stylus on the screen. So, the camera recognizes subsequent frames where the LED is flashing at the same location. The software is calibrated in a way which tells it that if it receives 10 frames (within a 2 second time interval) where the point of touch has the same X and Y co-ordinates, it recognizes it as a right-click. Then along with of a 2X1 matrix, it also sends to the operating system a single binary bit, 1, which tells the operating system that it is a right-click. Again, to summarize, touching and holding the LED stylus on any point in the screen facilitates the same action as a right click at that point on the actual screen would do.

The last thing worth mentioning here is that the software is programmed to receive around 35 pictorial inputs per second, and process them at the same rate. So, it is possible to project a white board on the screen and draw on it using the stylus. It will read each point and thus facilitate a continuous pattern being drawn on the screen. This is how the projected touch screen can be used as a whiteboard, or annotate on any part of the screen. The appropriate software for annotating on the screen is, however, required. For my project, I have used the annotating software "Smoothboard©".

3.3.2 For Android (or devices using a capacitive touch screen)

I have used an Android© smartphone running Android 4.4.4 operating system to demonstrate the output of the system. But, since my codes are written in JAVA, which is platform independent coding language, they should work on any device running operating systems which use a capacitive touch screen, like iOS or Windows 8.1 mobile.

Like stated above the output of the software is a 2X1 matrix that holds the X and Y co-ordinates of the point where the user has touched on the screen. This output is then sent as an input to the Android operating system, more specifically the touch position handling chip of Android. Thus, this matrix mimics the controller input that provides to Android the position where the touch occurs. So Android reads this touch as a touch on the capacitive touch screen occurring at that position, and carries out the relevant function that a touch at that position would bring about. This completes the entire system. So, to summarize, **a touch with the LED stylus on the projected screen brings about the same function as a touch on the capacitive touchscreen at that position on the actual screen would achieve.**

How does a capacitive touch screen work?

To fully understand the system, we first have to briefly go over how a capacitive touchscreen actually works. A touchscreen is made up of the following components:

- The glass display screen

- Layers of protective coating to protect the finger touching the screen, and the screen itself
- A horizontal grid that provides the initial charge
- A vertical grid of capacitors. (A standard touch screen has anywhere between 250 to 500 capacitors)

For our purpose, we will only be focusing on the capacitive grid. A touchscreen works by taking advantage of the electrical conductivity of a human finger. All the capacitors in the capacitive are charged to an initial charge by the horizontal grid. When we touch any pint on the screen, the capacitor present at the point of touch is discharged through the finger. Charge flows into our finger and leaves through it to complete the circuit, like illustrated in Figure 3.4.

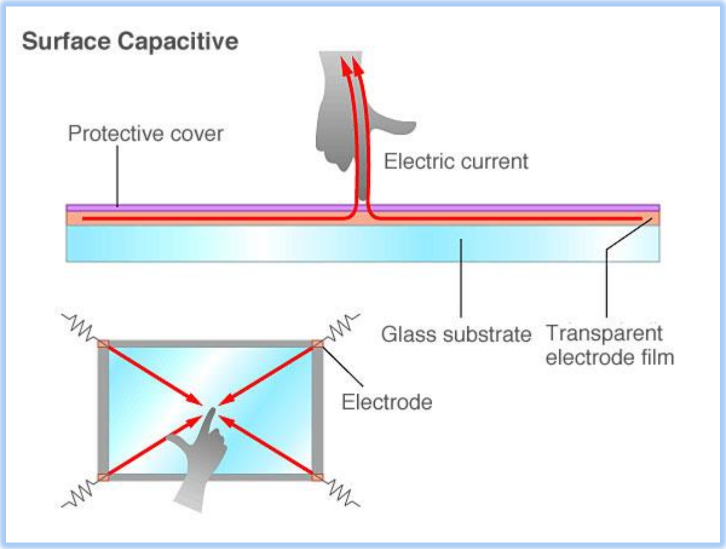


Figure 3.4: Capacitor discharge caused by a touch*

*Source: http://www.eizoglobal.com/library/basics/basic_understanding_of_touch_panel/

Now, when the capacitor at a particular point discharges, there is a distortion in the electrostatic grid in the touch screen, like illustrated in figure 3.5. The image processing controller inside the phone then reads this distortion, and locates the point of maximum surge in the current. The position of this surge is identified as the point of touch and the controller resolves it to a matrix containing the X and Y co-ordinates of this point, and relays it to the operating system. The operating system then brings about the action a touch at that point is supposed to.

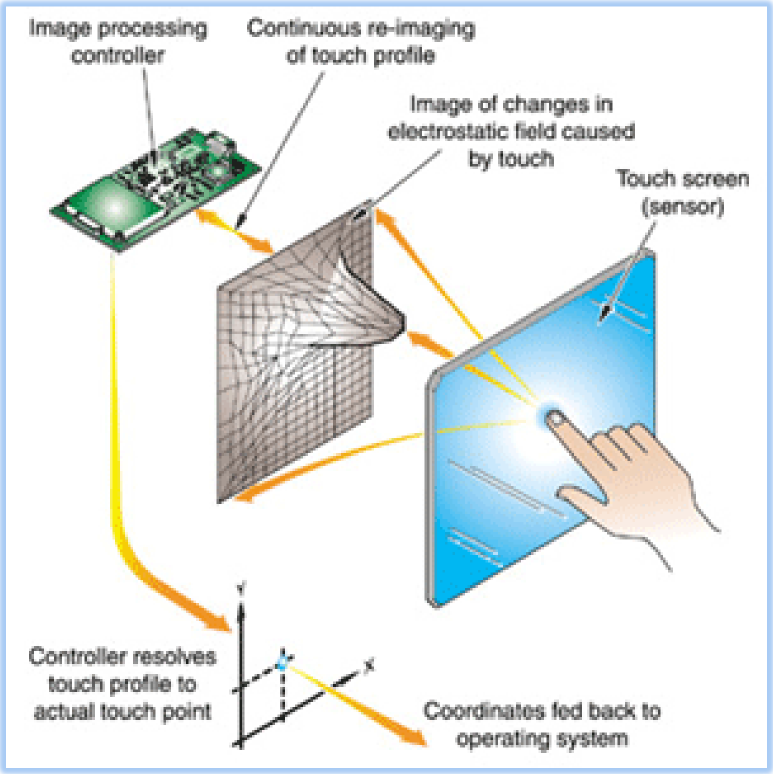


Image processing controller
Continuous re-imaging of touch profile
Image of changes in electrostatic field caused by touch
Touch screen (sensor)
Y
X
Controller resolves touch profile to actual touch point
Coordinates fed back to operating system

Figure 3.5: Identifying the touch on a screen*

*Source: http://wiki.fluidproject.org/display/fluid/Benchmarking+-+Touch+Screen+Options

It is worth mentioning here that since a touch screen makes use of the electrical conductivity of a human finger that is the reason it does not work when we touch the screen with gloves on, or with a non-conducting stylus. Also, the very sensitive touch screens work when we bring our finger close to the touch screen without actually touching, and when we touch it very lightly. This makes it highly sensitive. This is unlike a resistive touch screen, which has a grid of resistors and identifies the position based on the increase in resistance when the screen is pressed. This means that we have to press harder on a resistive touch screen, making it less sensitive, but functional when touched with any stylus. Thus, a capacitive touch is harder to replicate without actually touching the screen. My technology ideally only works for devices using a capacitive touch screen.

Difference between a touchscreen device and a pointer controlled device:

Now let us focus on the **difference between a device that uses a touch screen, and one that uses the pointer**. This difference is essential to understanding how the system mimics a touch on the projected screen for a touchscreen device. Normally, all devices using a pointer comes with a factory fitted mouse or a touchpad (for laptops). The operating system is designed to receive the information of the pointer location from this. But, when we attach an external mouse to any of the communication ports of the device, or wirelessly by Bluetooth, the operating system automatically configures itself to receive inputs from that channel, alongside the factory fitted mouse. Thus, **the controller in a pointer-using device is factory configured in a way such that it can not only receive the pointer position information from its default channel, but other channels as well.** Here lies its main difference with touchscreen device. **The controller in a touchscreen device is configures to receive the touch position from one and only one channel, that is the chip that tracks the distortion in the electrostatic grid**. So we cannot directly access the operating system without going through this chip.

How we do it:

Like mentioned above, for devices using a capacitive touchscreen, the operating system is only configured to receive inputs from the chip that translates distortion in the electrostatic field to X and Y co-ordinates. So, we cannot send inputs to it directly. For this reason, after the screen is calibrated, the software designs within itself a virtual grid replicating the capacitive grid of the touchscreen. When it receives the input from the infra-red camera and triangulates the X and Y co-ordinates of the point of touch, it is sent to the virtual mesh. Then a symmetric Gaussian distortion is formed on the mesh centered on the X-and-Y coordinates previously triangulated, as shown in figure 3.6. This grid is created using MATLAB for illustration purposes.

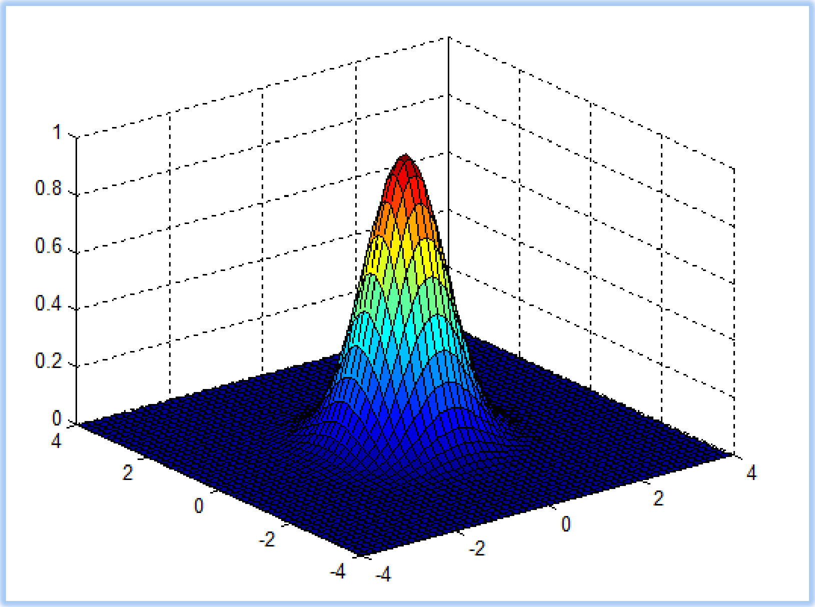


Figure 3.6: Generating a distorted mesh

This distorted virtual grid, replicating the distortion caused on the capacitive grid by an actual touch, is then sent to the controller chip. Then the normal functionality of the device takes over. The position of this surge is identified as the point of touch and the controller resolves it to a matrix containing the X and Y co-ordinates of this point, and relays it to the operating system. The operating system then brings about the action a touch at that point is supposed to. Thus, the entire system is completed.

Again, it's worth mentioning here that this process is not very efficient. There is a very high redundancy because first the X and Y coordinates are transformed into a distorted grid and fed into the controller, which does the exact opposite, i.e. extracts the X and Y coordinates from the grid. Formation of the distorted mesh and feeding it into the controller requires a very high amount of RAM usage of the phone, and thus we can always observe a lag between touching on the projected screen and the appropriate action being performed. Also, this is the reason that, for now, this technology only works when we are inside the application. For the application to run in the background and enable the entire system being controlled from the projected screen, we would need a device with very high processing power, or find a way to bypass the position determining controller chip and feed the received X and Y co-ordinates directly to the operating system. An example of the entire system is given in figure 3.7

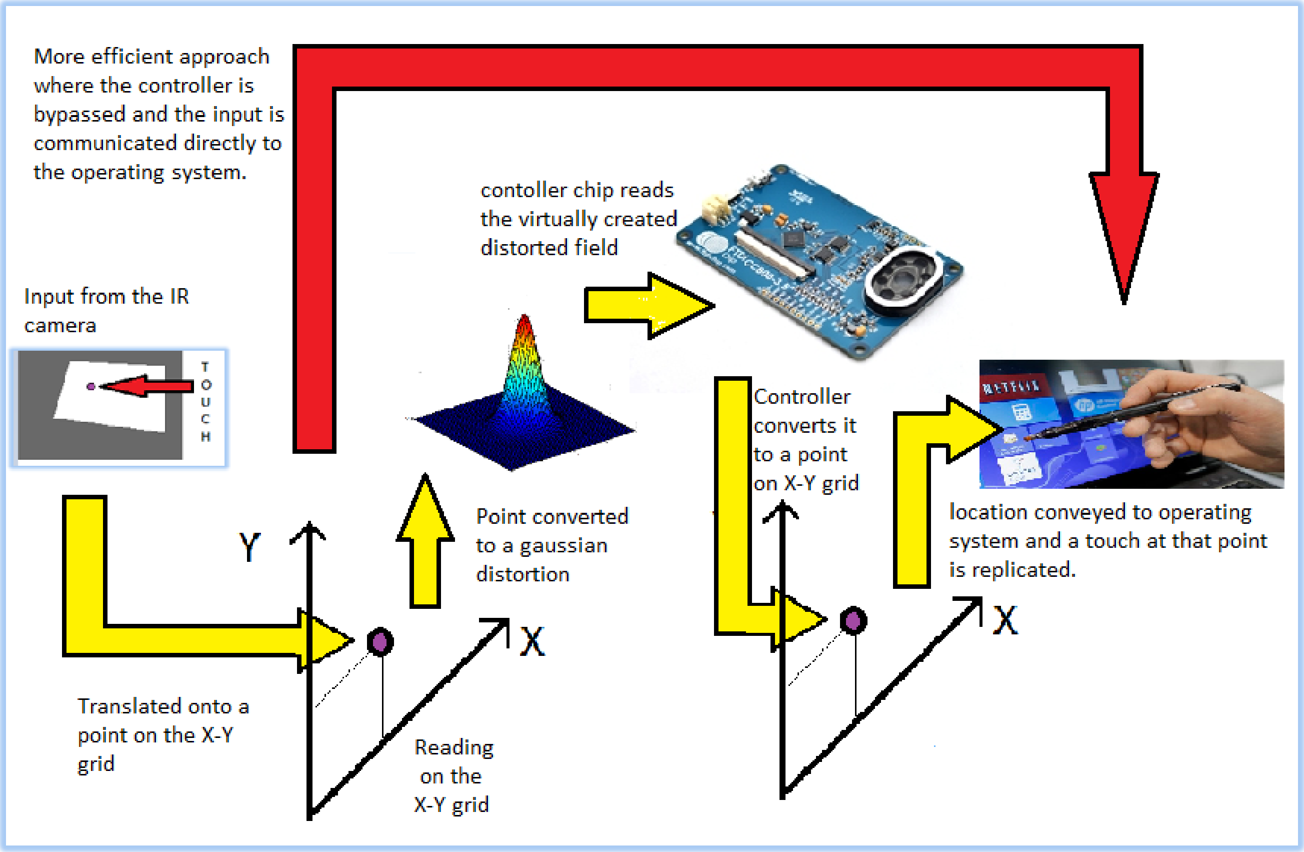

More efficient approach where the controller is bypassed and the input is communicated directly to the operating system.
contoller chip reads the virtually created distorted field
Input from the IR camera
TOUCH
Controller converts it to a point on X-Y grid
Y
Point converted to a gaussian distortion
location conveyed to operating system and a touch at that point is replicated.
X
X
Translated onto a point on the X-Y grid
Reading on the X-Y grid

Figure 3.7: The entire system for devices using a touch screen.

CHAPTER 4: PARAMETERS TO CONSIDER

Though this technology operates relatively without problem, there are some parameters we should consider while setting up and calibrating it, because they hamper the accuracy of determining the touch. The most important of such parameters are discussed below:

1. **Resolution of the camera:** The resolution of the camera is of high importance in this setup based on the size of the screen it is looking at. We can use a low resolution camera to look at a projected screen comparable to the size of the original screen. But if the size of the projected screen is substantially bigger than the actual screen, we need to look at it using a high resolution camera. Only then will it be able to transpose each point of the small screen to a grid on the big screen and have a more accurate tracking resolution.

2. **Angle at which the camera views the screen:** Different cameras have different fields of view. We should make sure that the camera is looking at the screen in a way which offers it an optimal view of the complete screen without being blocked when the user interacts with the projected screen. So, it is always good to have the camera looking at the screen at an angle instead of orthogonally to prevent the LED flash being blocked by the user or the body of the stylus.

3. **Interference:** This technology works on the basic principle of infra-red light being detected by the camera. If there are other sources of infra-red light of comparable

wavelength present in the room, it may interfere with the detection of the needed LED flash, and thus affect the functioning of the system.

4. **Diffraction:** The bending of light when it travels through a medium (air in this case) is called diffraction. The ray of light from the LED flash, when the stylus touches the screen, bends when travelling from the LED to the camera through air. This reduces tracking accuracy. But if the distance is not very large, this effect can be generally ignored as it does not hamper the functioning of the system in a noticeable way.

5. **Distance:** Like mentioned before, the projected screen should cover as much as possible of the view of the camera. If we move the camera too far back, the screen will only cover a small portion of the view. If we have it too close to the screen, the camera will not be able to see the entire screen. In both these cases, we will lose tracking resolution. So we should ensure the camera is as close to the screen as possible without losing sight of the any part of the screen.

CHAPTER 5: PIEZOELECTRIC SENSOR INTEGRATION

A piezo-electric element undergoes a small deformation when electricity is passed through it, and conversely, when it is deformed by an external pressure, a small charge is generated. Piezoelectricity refers to the generation of electricity or of electric polarity in dielectric crystals when subjected to mechanical stress and conversely, the generation of stress in such crystals in response to an applied voltage.

For this paper we use a piezo element to detect a touch. There is a small vibration or pressure difference when the piezo element is tapped or knocked on a surface. This leads to a small temporary deformation being brought about in the element. We can use these for tap or knock sensors pretty easily by reading the voltage on the output. The generated charge can then be used to flash an LED, thus confirming the detection of a touch.

The following discussion assumes that the output needs a moderate amount of amplification, and that the desired signal levels are in the 2-V to 4-V range for full scale. We use the circuit shown in Figure 4 to amplify the signal; a MOSFET can be used to turn on and off an LED. To prevent the LED from burning out due to the current that flows through it, we have placed a resistor in series before the LED. For a 5V supply, a typical value is approximately 1k ohm. In this way, we have the gain of your circuit as large as possible so that any touch will cause the amplifier to go to the positive supply voltage.

Figure 3.1 shows a charge mode amplifier circuit. Charge mode amplification is used when the amplifier is remote to the sensor. The charge mode amplifier will balance the charge injected into the negative input by charging feedback capacitor $\mathbf{C_f}$. Resistor $\mathbf{R_f}$ bleeds the charge off capacitor $\mathbf{C_f}$ at a low rate to prevent the amplifier from drifting into saturation. Resistor $\mathbf{R_f}$ also provides a dc bias path for the negative input [3]. The value of $\mathbf{R_f}$ and $\mathbf{C_f}$ set the low cutoff frequency of the amplifier. The protective resistance R will make sure the LED does not burn out due to a surge in the current flowing through it.

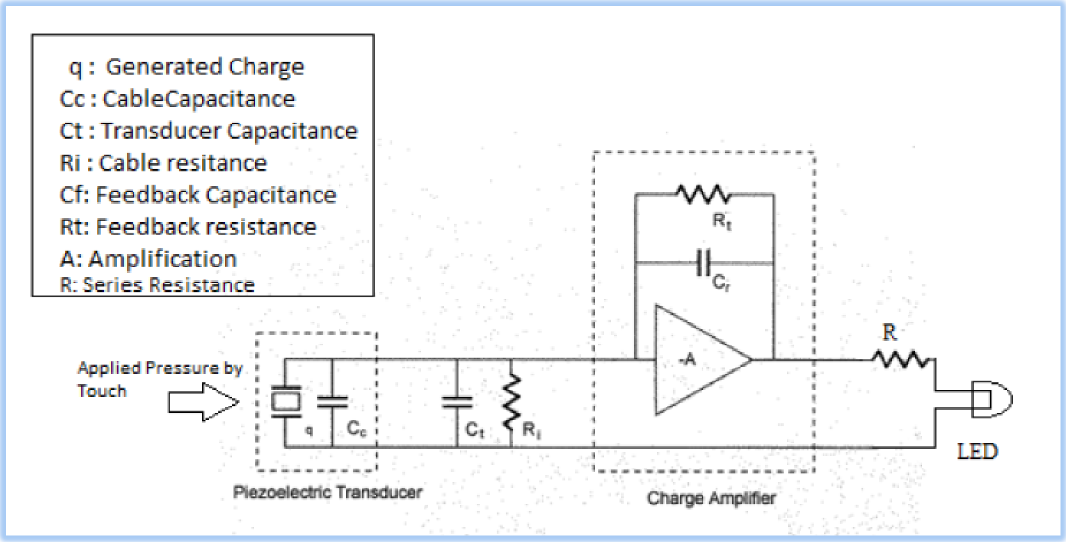


Fig 5.1: The Piezo-Electric Circuit

The action of the amplifier maintains 0 V across its input terminals so that the stray capacitance associated with interface cabling does not present a problem. Resistor $\mathbf{R_i}$ provides ESD protection. Resistor $\mathbf{R_i}$ and capacitors $\mathbf{C_p}$ and $\mathbf{C_c}$ combine to produce roll off at higher frequency [3].

5.1 The Application

The use of Piezo-electric sensors in this project mainly serves two main purposes. It makes the stylus battery independent, or any other external power source. This is helpful because the usage of an external power source would make the stylus bulky and less robust. The second and the more important reason is that the piezo-electric element is that it is used as a touch sensor. It can recognize when an actual touch is made, as opposed to the stylus being brought very close to the screen without actually touching. The use of an IR camera makes it possible to pick up the LED flash as soon as the touch is made, thus making an effective identification.

CHAPTER 6: ADVANTAGES, APPLICATIONS AND FUTURE WORK

In spite of some short-comings, this technology provides many advantages when properly implemented. Like mentioned before, the codes for the software that this program is written in JAVA, which is platform independent, has a compiler of its own and is completely robust. Other than the software, the infra-red camera which reads the input from the stylus is also relatively easy to connect to almost any device. This makes this technology relatively **platform independent,** and it can work with a variety of devices irrespective of whether it uses a mouse pointer, a touch screen or both. In theory it can work with any devices which rely on mapping an input point to a two-dimensional X_Y plain. Also, it works on a variety of operating systems since the software is coded in JAVA, like Windows, MacOS or Ubuntu for computers; Android, iOS or Windows mobile for smartphones and tablets.

Note: This system has been experimentally verified to be working with Windows 8.1 for computes and Android 4.4.4 for smartphones. In theory it is designed to work for all the operating systems mentioned above, but it has not been experimentally verified.

Throughout our discussion of the system, one thing is very clear, that the device is unaware of the screen being projected in detail. It is only aware of its position based on the four point touch calibration and the point where a touch occurs based on the LED flash from the stylus. The projected screen is changed after the touch is recognized, translated and acted on by the operating system. So, in other words, the projected screen changes because the original screen on the device is changed. This feature makes it pretty easy to understand that this technology **is independent of the surface being projected on**. We can use a projector to project onto a white screen, cast the screen onto a television, or just point the camera on our laptop screen to make it a touch operated device. As long as the surface has a clear unobstructed view of the entire projected screen, it is independent of the surface being projected on, and how we are projecting the screen and any other characteristics, like brightness or contrast, of the projection. For example, if we project the screen onto a normal table, and have the camera look at that, the table will serve as a smart table which we can interact with.

Another advantage of this technology is that it **supports the multi-touch feature** or more specifically, four simultaneous touches on the projected screen at a time. This means that the software is programmed to transmit a 2X4 matrix at a time. If there is only one touch, only the 1st column has real non-zero values, while the rest are zeros. If there are two points, the first two columns have real non-zero values while the other two have zero values. Thus, it can support up to four touches at a time with the 2X4 matrix.

Note: Though the multi-touch function is supported for this system, its working largely depends on whether the device it is being run on supports multi-touch or not. For example, Windows does not work with two simultaneous mouse left-clicks, but we can use multi-touch to scroll or magnify the screen. Similarly, we can use the system to scroll or magnify, but not to mimic simultaneous mouse clicks at two different points. Most smartphones and tablets, and the applications running on them use the multi-touch function, so the system generally works flawlessly with these systems.

One of the main advantages of this system is that it is very **cost effective**. Though most of the cost depends on the projection system and how good a camera we chose to use, it is still a lot less expensive than a smartboard while providing almost the same functionality. It is also worthwhile to remember at this point that the projection device is normally present by default at most places this system finds application. So, we could exclude its cost while calculating the total cost to build the system, since it does not require any projection system specifically designed to work with it, but any system present by default. The system I built to demonstrate the working of my technology cost me around $30 to make.

The versatility and robustness of this technology will find application in various fields when integrated properly. Below I attempt to list some of the main applications of the system and the future developments I envision for it.

1. **Teaching**: This technology can be very useful in the field of teaching. This will enable teachers to interact with their presentation on screen to emphasize something better, quickly solve equations on the same screen as the presentation without having to move to a separate board and quickly erase their work to move onto the next segment. This will also reduce the use of whiteboards and markers, in turn being beneficial for the environment as well.

2. **Sharing and collaborative work**: It is hard to share something with a large group of people using the small screen of a smartphone or a tablet. We may use a normal projection device to project the screen onto a larger one, but this would still mean that only one person has control of the projected screen through the smartphone. On the other hand, the use of the multi touch feature of this technology will enable multiple users to work on the same screen and dynamically present their ideas to the large group.

3. **Presentations**: It is very easy to see why this technology will be useful while giving presentations. The presenter can interact and annotate on his slides, write additional notes on for the better understanding of the audience and switch screen without having to walk up to his or her laptop.

4. **Integration with Automobiles**: One of the more interesting integrations of this system would be with automobiles. Now when the driver is following a GPS, or has to change any settings of the car, he has to take the eyes of the road to look sideways or downwards towards the dashboard. If we integrate this technology with automobiles and project all the dashboard controls, the speedometer, the fuel gauge and the navigation system onto the windscreen, the user can see and interact with the systems without having to take his eyes of the road. This is not only convenient, but also enhances safety.

5. **Interactive Holography**: One of the other interesting applications of this technology is the potential of its integration with holography. Since holographic images are made up of light, they are intangible. In other words, the human eye can see holographic images, but

humans cannot touch, or interact in anyway with such holograms. But, if we apply this technology to holograms, we can facilitate interaction with such holograms. I discuss a little bit more about such integration in the next section.

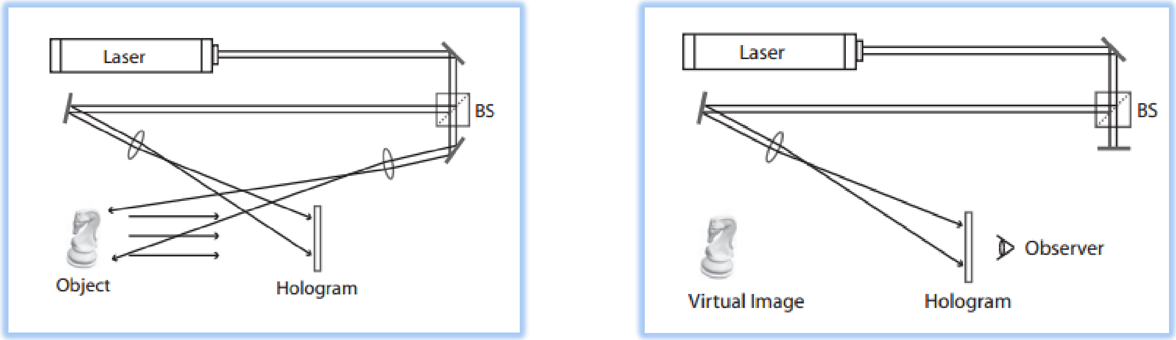
6.1 Integration with Holography

Holography is a method by which an exact replica of an object can be made. This is done by recording of both intensity and phase of an image [17]. This might be a three dimensional or two dimensional (depending on the original object) image created by scattering light rays from the original object [10]. The image can be reconstructed later by recording the interference pattern of light. Interference is created between the light rays from a point of light source and from each and every point of the actual object whose hologram is to be made. Interference can only be created when the beam from the fixed source of light and the beam form the object are of the same wavelength [11]. The process can be done by projecting the rays at particular angle, certain width of the rays and by using reflection or diffraction techniques. Holography can create a virtual image of the object in front of a person's eyes but since the object is being built with rays and perception of the human eyes it cannot be felt by human touch. This duplicated object is a hologram. Below is the list of some items which can be used to create a hologram:

- Laser
- Beam splitter
- Spatial filter

- Mirror
- Diverging Lens
- Film plate

Figure 8.1 shows a simple diagrammatic illustration of how a hologram is acquired and how it is replicated.



(a)

(b)

Figure 6.1: Holography (a) Acquisition of the holographic plate (b) Formation of an Hologram*

With the use of this technology, we have the potential to make holography interactive. Like mentioned above, human eyes can only perceive holograms, and not touch it because it’s a virtual image made up entirely of photonic interference. But if we integrate this system, we can interact with light using light itself. We have to ensure, however, that the LED flash of light from the stylus is of a different wavelength than that creating the holograms. If the light that detects a touch on the hologram is not isolated, it may lead to a misidentification of the touch and hamper functionality of the system.

*Source: http://physastro.pomona.edu/wp-content/uploads/2012/09/Wendt.pdf

The basic principle behind integrating this system to facilitate interactive holography is to ensure the holographic plate is rotated or turned following the movement of the stylus. The main system will remain the same. We have to first calibrate it to recognize the position of the holographic plate or the hologram itself. Then, when we touch the hologram with the stylus, it is fed as an input to the software, which in-turn tracks its movement on the X-Y plane and feeds it to a motor attached to the holographic plate. The holographic plate turns in a way to mimic the movement of the stylus, which would turn the hologram itself. This is the main theory behind creating interactive holograms using this system. This, however, has not been experimentally achieved and it is, for now, just in the conceptualization stage.

6.2 Future Work

This technology, though in its infancy, has the potential to be widely used in various fields, as mentioned in the applications. The main feature of this technology is that it is platform independent, independent of the surface and characteristics of the projected screen and cost effective. This makes the system dynamic and capable of being used on various applications. With proper implementation, this technology can make anything from a television to a simple table touch controlled. This will not only be ergonomic, but also beneficial to the environment as it helps reduce waste.

This technology, if integrated with automobiles promotes safety and easier driving. The integration of this technology with holography can eliminate the use of a screen on the whole, and it is one of most promising uses of it. Thus, as we see, even though the system is at its very initial stages of development, it already shows great potential and versatility. So it is a very strong possibility that this will have a very part to play in the shaping of human-computer interaction in the future.

BIBLIOGRAPHY

1. Lee, Johnny Chung, Projector-Based Location Discovery and Tracking, May 2008
2. Lee, J., Hudson, S, "Foldable Interactive Displays" Proceedings of the ACM Symposium on User Interface Software and Technology, October 2008
3. Karki, J., “Signal Conditioning Piezoelectric sensors, Mixed Signal Products.
4. Gautschi, G (2002). Piezoelectric Sensorics: Force, Strain, Pressure, Acceleration and Acoustic Emission Sensors, Materials and Amplifiers. Springer.
5. J. Krautkrämer and H. Krautkrämer (1990). Ultrasonic Testing of Materials. Springer.
6. https://moodle.fp.tul.cz/nano/pluginfile.php/2476/mod_resource/content/3/FPM_Piezo_lecture1.pdf
7. Jacques and Pierre Curie (1880) "Développement par compression de l’électricité polaire dans les cristaux hémièdres à faces inclinées" (Development, via compression, of electric polarization in hemihedral crystals with inclined faces), Bulletin de la Société minérologique de France, vol. 3, pages 90 - 93. Reprinted in: Jacques and Pierre Curie (1880)Développement, par pression, de l’électricité polaire dans les cristaux hémièdres à faces inclinées," Comptes rendus ... , vol. 91, pages 294 - 295. See also: Jacques and Pierre Curie (1880) "Sur l’électricité polaire dans les cristaux hémièdres à faces inclinées" (On electric polarization in hemihedral crystals with inclined faces), Comptes rendus ... , vol. 91, pages 383 - 386.
8. Lippmann, G. (1881). "Principe de la conservation de l'électricité". Annales de chimie et de physique (in French) 24: 145.
9. Jacques and Pierre Curie (1881) "Contractions et dilatations produites par des tensions dans les cristaux hémièdres à faces inclinées" (Contractions and expansions produced by voltages in hemihedral crystals with inclined faces), Comptes rendus ... , vol. 93, pages 1137 - 1140.
10. Jeong, Tung H. et all (1999) Basic Principles and Applications of Holography
11. Yariv , Amnon (1978). Phase Conjugate Optics and Real-Time Holography
12. Rossing, Thomas D., and Christopher J. Chiaverina. Light Science, Physics and the Visual Arts. New York: Springer-Verlag, Inc., 1999.
13. Saxby, Graham, Practical Holography, 3rd edition. Prentice-Hall, Inc., 1996.
14. Jeong, Tung H. Laser Holography—Experiments You Can Do. Lake Forest, Illinois, Thomas Alva Edison Foundation, Integraf, 1987.
15. Vest, Charles M. Holographic Interferometry. New York: John Wiley & Sons, 1979. Bjelkhagen, Hans I. Silver-Halide Recording Materials for Holography and Their Processing, 2nd edition. New York: Springer-Verlag, 1996.
16. Pedrotti, Leno S., and Frank L. Pedrotti, Introduction to Optics, 2nd edition. Prentice-Hall, Inc., 1993.
17. Wendt, James B., Computer Generated Holography, 2009
18. Lee, Johnny C., et all Haptic Pen: A Tactile Feedback Stylus for Touch Displays, 2003

19. Holler, F. James; Skoog, Douglas A; Crouch, Stanley R (2007). "Chapter 1". Principles of Instrumental Analysis (6th ed.). Cengage Learning. p. 9. ISBN 978-0-495-01201-6

20. Badyopadhyay, D., Raskar, R., Fuchs, H., “Dynamic Shader Lamps: Painting on Real Objects.” IEEE ACM Int. Symposium on Augmented Reality (ISAR), New York, 2001.

21. Bellcore. The VideoWindow Teleconferencing Service Model. Special Report SR-ARH-001424. Morristown, NJ: Bell Communications Research, 1989.

22. Baar, J van., Willwacher, T.; Rao, S.; Raskar, R., "Seamless Multi-Projector Display on Curved Screens", Eurographics Workshop on Virtual Environments (EGVE), ISBN: 3-905673-00-2, pp. 281-286, May 2003

23. Baudisch, P., Good, N., and Stewart, P. “Focus Plus Context Screens: Combining Display Technology with Visualization Techniques.” In proceedings of ACM UIST ’01, Orlando, FL, November 2001.

24. Ben-Joseph, E., Ishii, H., Underkoffler, J., Piper, B., and Yeung, L. “Urban Simulation and the Luminous Planning Table: Bridging the Gap between the Digital and the Tangible.” In Journal of Planning in Education and Research, Volume 21, pp.195-202. 2000.

25. Anaheim, CA., Aug. 1993, Bimber, O., Raskar, R. "Spatial Augmented Reality: Merging Real and Virtual Worlds" Publisher: A K Peters, Ltd. (July 31, 2005)

26. Chen, N., et al. “Enhancing Document Navigation Tasks With a Dual Display Electronic Reader”, Proceedings of ACM Symposium on User Interface Software and Technology, October 2007.

27. Cotting, D.; Naef, M.; Gross, M.; Fuchs, H., "Embedding Imperceptible Patterns into Projected Images for Simultaneous Acquisition and Display." Third IEEE and ACM International Symposium on Mixed and Augmented Reality, 02-05 Nov. 2004 Page(s):100 109.

28. Cruz-Neira, Carolina, Daniel J. Sandin, and Thomas A. DeFanti. “Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE,” Computer Graphics, SIGGRAPH 1993.

29. DePiero, F.W., and Trivedi, M.M., “3-D Computer Vision using Structured Light: Design, Calibration, and Implementation Issues,” Advances in Computers(43), 1996, Academic Press, pp.243-278

30. Dietz, P., Leigh, D., “DiamondTouch: a multi-user touch technology”, Proceedings of the 14th annual ACM symposium on User interface software and technology, November 2001, Orlando, Florida

31. Gardner, M. "The binary Gray code", in Knotted donuts and other mathematical entertainments, F. H. Freeman and Co., NY, 1986

32. Gibbs, Simon, Constantin Arapis and Christian J. Breiteneder. “TELEPORT-Towards Immersive Copresence,” ACM Multimedia Systems Journal, 1998.

33. Gray, F., “Pulse Code Communications” United States Patent 2632058, issued 1953.

34. Han, J., “Low-cost multi-touch sensing through frustrated total internal reflection”, Proceedings of the 18th annual ACM symposium on User interface software and technology, October 2005, Orlando, Florida

35. Hashimoto, S., Matsumoto, T., "Pileus: The Interenet Umbrella". http://www.pileus.net 2006

36. Hinckley, K., “Bumping Objects Together as a Semantically Rich Way of Forming Connections between Ubiquitous Devices.” UbiComp 2003

37. Hinckley, K., Ramos, G., Guimbretiere, F., Baudisch, P., Smith, M., "Stitching: Pen Gestures that Span Multiple Displays." In Proceedings of ACM Advanced Visual Interfaces. 2004

38. Holman, D., Vertegall, R., Altosaar, M., Troje, N., Johns, D., "Paper windows: interaction techniques for digital paper" Proceedings of the SIGCHI conference on the human Factors in computing systems, 2005.

39. Horaud, R. Conio, B. Leboulleux, O. Lacolle, B.," "An analytic solution for the perspective 4-point problem" Proceedings of Computer Vision and Pattern Recognition, June 1989.