FPS Interactive Whiteboard

A Capstone Project

Presented to the

Department of Information Technology

Institute of Information and Computing Sciences

University of Santo Tomas

In Partial Fulfilment

of the Requirements in the Degree of

Bachelor of Science in Information Technology

Fajardo, Xavier Girard M.

Paman, Julian Patric Joshua G.

Parayno, John Angelo D.C.

Siy, Karlene C.

Tayuan, Ronina-Caoili

May 2016

APPROVAL SHEET

Thesis Title: FPS Interactive Whiteboard

Proponents:

1. **Xavier Girard M. Fajardo**
2. **Julian Patric Joshua G. Paman**
3. **John Angelo DC. Parayno**
4. **Karlene C. Siy**
5. **Ms. Ronina C. Tayuan**

In partial fulfillment of the requirements for the degree of ***Bachelor of Science in Information Technology***, the capstone project mentioned above, has been adequately prepared and submitted by the proponents. This thesis was duly defended in an oral examination before a duly constituted tribunal with a grade of

Ms. Maricel Afurong-Balais
Panel Member

Engr. Ma. Ian P. De Los Trinos
Panel Member

05/07/16

Mr. Jairus Llamas
Panel Member

Asst. Prof. Jerralyn T. Padua
Capstone Project Coordinator

Accepted in partial fulfillment of the requirements for the degree of Bachelor of Science in Information Technology.

Engr. Mia V. Eleazar
Chairperson
Department of Information Technology
Institute of Information and Computing Sciences

ABSTRACT

The FPS Interactive White-Board is a system created in C++ using the Kinect motion capture sensor to allow users interaction with a computer environment using the their hand movements as the input for cursor manipulation and the LED pens as the input for action identification. The main objective of this project is to improve upon the technologies and functionality of the previous capstone project entitled “Interactive Whiteboard using Raspberry Pi”, also improve on certain existing functionalities such as a more accurate way of tracking and translating of the users’ hand movements and a more apt response time when it comes to color detection for action identification.

The proposed system will benefit schools and other learning institutions that can incorporate interactive applications such as drawing applications, video applications and even educational games with their curriculum. The system can allow students a more hands on experience with certain applications as they can interact with those applications using the movement of their hands, this kind of interaction is also not limited with only one user but instead has a maximum capacity of 2 users.

The system is recommended to be used in an open space, a typical classroom setting or as specified in the latter parts of this documentation.

ACKNOWLEDGEMENT

The researchers would like to express their deepest gratitude to the people who had been involved in the realization of this capstone project. The team would like to extend their gratitude to Mrs. Ronina Caoili-Tayuan of the Institute of Information and Computing Sciences, Information Technology Department, for her unparalleled teachings during the development of the system and the documentation of the project.

The team would also like to thank our 2 thesis coordinators Mr. Karl Taag and Mrs. Jerralyn Padua for pushing us to do our best and guiding us throughout the first and second term for the Capstone project. The team would also like to thank Mrs. Maricel Afurong-Balais, Mr. Jairus Ferdinand Llamas, and Mrs. Ian Delos Trinos for giving us constructive criticism during our first and second round of defending this Capstone project as panelist.

The team would also like to thank Mr. Jose Francis Floresta, for sharing to us his knowledge about hardware technologies. Mr. RP Lafuente, for assisting the team with any hardware issues we encountered and also Mr. Jerome for assisting the team with the creation of the mold for the LED Pens.

To fellow classmates, family and friends, thank you for the unending provision. Thank you for believing and supporting all through times. Lastly we thank the almighty God for making all things possible.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

FPS Interactive Whiteboard came about from reviewing an IT Capstone Project of the year 2014, Interactive Whiteboard using Raspberry Pi**,** which aims to develop a prototype of a cost-effective interactive whiteboard that can be used for a computer-interactive education. It is a technology wherein a user is able to manipulate the objects projected into a whiteboard using a Raspberry microcomputer and a camera that will detect input from an LED pen. The proponents learned that project has more room for improvement it terms of functionality and usability. The proponents decided to make their own Interactive Whiteboard which aims to provide a better system for a computer-interactive education by utilizing a more accurate motion detection, providing multi-user functionality and creating a system usable on a lighted classroom environment.

Project Context

Modern technology influences many people nowadays. The continuous increase of innovation on modern computing has affected many students' learning style. Not long ago, professors or teachers used chalkboards in teaching students or presenting lessons, activities and quizzes. Whiteboards, which are introduced later than chalk, utilizes whiteboard markers which are dust-free as opposed from chalk sticks. At the present time, Liquid Crystal Display (LCD) projectors were now being used as a medium for teaching, it is being used for PowerPoint presentations or videos for visual learning.

During the 1990s, Xerox Parc, developed an interactive whiteboard (IWB), for small group meetings which was later followed by SMART Company. With the help of IWB, users were able to write on the whiteboard virtually without using a regular whiteboard marker. It was later introduced to schools for a better interactive learning experience.

Humaidan (2012) conducted a study in Saudi Arabia about positive effects of using IWB in improving students' achievement in private middle schools in Riyadh. The conducted study concludes that there is a statistically significant difference based on average achievement of the control group and the experimental group. 8.666 average achievements were resulted from the control group while the experimental group had 14.166. The result shows that using IWB will give a positive outcome among the students' achievements. The condition of the schools in Riyadh shows that there are already IWBs in use.

In the Philippines however, IWBs are not popular as it is in overseas like in the Unites States of America. GMA Network covered a report in Negros Occidental last December 2013 that 13 public schools in the said area are given the opportunity to experience to use IWBs. It was a project cooperated by the Department of Education (DepEd) and made by a private sector. DepEd Secretary Armin Luistro stated that old mode of teaching will be unreliable if it is still implemented such as the teacher standing in front giving information. He added that the best way to learn is when students discover things on their own.

The conducted project in Negros Occidental was beneficial for the schools, considering how expensive IWBs are. One of the most expensive IWBs in market are sold by the SMART Company called the SMART Board. Due to the expensiveness of the said IWB, many educational institutions are not investing on it.

In the recent years, with the help of advanced technology, an IWB can be made with cost-effective solutions. Many alternative solutions have appeared on the internet, this includes a Kinect for windows, a Nintendo Wii Remote (Wiimote) or even an ordinary webcam which acts as a sensor that tracks hand or object movements. The said solutions were made and developed by different communities volunteering for a better and cheaper IWBs.

An existing capstone project which is a prototype of a cost-effective whiteboard and was created using Raspberry Pi, a webcam (Pi Cam), a projector and a customized Light Emitting Diode (LED) pen that acts as the sensor for the webcam to track gestures that the user does on the board. The proponents had found out that it has more room for improvement and the proponents would likely to enhance it.

Due to lack of available resources and required power for the Raspberry Pi, the proponents decided to make FPS Interactive Whiteboard which uses Kinect as its main technology for motion and color tracking. Tracking will be utilized for manipulating mouse cursor events of the GUI of the Windows OS platform to be used. The output will be displayed on an LCD project on a classroom environment.

Purpose and Description

The FPS Interactive Whiteboard aims to provide a quality interactive learning environment to classrooms by integrating different existing technologies on a typical classroom medium which is the Whiteboard. The beneficiaries of the project are basically the educational institution which consists of students and teachers. The project encourages teachers to use technology for students to bring in a fun and interactive learning environment. In result, this makes students be more active in participating on learning activities that will improve their performance.

The project will be used for wireless multi-user interaction on objects displayed on the whiteboard through an LCD projector. It will use for operating PowerPoint and video presentation, and most especially for educational games and drawing applications the teachers will prepare for his or her students.

To achieve this project, the proponents will use sensors such as Kinect™ as the hardware for capturing the motion of the users. OpenCV, OpenNI, NiTE and Kinect for

Windows SDK 1.8 will the libraries and tools to be used that will be programmed on the C++ language.

Objectives

The main objective of the project is to create an Interactive Whiteboard to be used for computer-interactive education which will have an accurate and precise motion tracking, multi-user functionality and is working on any given lightning conditions with the use a Kinect Sensor and LED pens.

Specifically, the project aims to achieve the following objectives:

- To develop a motion and color tracking system with the use of a Kinect sensor and LED pens.
- To develop an accurate and precise motion tracking for the system in which will be tested on drawing applications.
- To develop a usable project that will work to a typical lighted classroom environment.
- To recognize another user input that will not interfere to the other, which aims to provide a multi-user functionality.

Scope and Limitation

The project covers the following functions and process:

- The system will be utilized for operating simple educational and drawing applications; and PowerPoint and video presentation for a classroom environment.
- The system covers the use of motion and color tracking with the use of Kinect technology to utilize the system cursor.
    - The system cursor movement will react according to the arm movement of the users.

    - The system cursor behavior like move, click and drag will react according to the color input of the LED pens.
- The LED pen will be used to produce 3 specific colors for the input for each user that will be made main through an RGB LED.
    - Another set of predefined color will be used for another LED pen which will be used by another user.
    - The LED Pen will be used generally for customized input in which on this project will focus on mouse cursor events.
- The system will track at most two users at a time providing multi-user functionality.
    - Another user will have a separate cursor to manipulate.
- The system will be developed to have an accurate and precise motion tracking.
- The system will be developed to be usable to a typical lighted classroom environment.

However, the project is limited of the following:

- The motion tracking will cover only x (width) and y (height) movements only.
- The color tracking will cover only 6 colors.
- The LED pens designed only by the proponents will be used.
- Based on the Kinect specification (Table 3.3). The limitations of the Kinect sensor are as follows:
    - Depth sensor range
        - Minimum: 800mm
        - Maximum: 4000mm.
    - Viewing Angle
        - 43° vertical
        - 57° horizontal

CHAPTER 2

REVIEW OF RELATED LITERATURE, STUDIES, AND SYSTEMS

According to Tosuntas, Karadag and Orhan (2014), the IWB was first designed and developed for the business world before it has been used in education. IWB took an important role in integrated technology in education and in many countries nowadays, it is being installed in schools within the context of their education policy.

Competente, Eugenio, Fernandez and Rayos (2015) stated that “The IWB, is a useful technology for instructors all over the world. It helps them focus the attention of their students to the lessons.” IWB is a new generation board that is a helpful technology for the instructors that will enhance the learning experience of the students and they will be motivated to learn. In Istanbul, Turkey, all participants declared that they use a portable IWB such as eBeam and Mimio because of its portability and it is inexpensive. It is widely used in Turkey. (Türel, 2011).

Some IWBs utilize a dry marker pen which commonly shows infrared (IR) signs to movement sensor device. The classroom application as indicated by an article "IWBs Enhance Classroom Instruction and Learning" incorporates media lessons and presentations, community oriented critical thinking, virtual field trips, recorded lessons, and so on. The reason that IWBs are not being totally utilized for training by most schools in the Philippines is its high cost. There are some known brands of IWBs which give great client encounter however evacuates reasonableness because of it being a high cost device.

As indicated by the authority site of SMART, the organization is thought to be the first dealer of IWBs in 1991. It turned into the world's top rated IWB with the expense range from 800 USD up to 20,000 USD. (Source: smarttech.com).

Promethean offers almost the same features of SMART Board, however Promethean is able to make the IWB teacher-centric. An online community is actively helping teachers use the technology by providing hundreds of thousands of lessons and resources which are shared through the community. (Source: prometheanworld.com).

Unlike Promethean, SMART Board, as per Dawson (2010) cook on both business and instructive markets. This organization empowers them to search for conceivable specialized developments and diminishing the educator driven methodology for instructive advantages. Other than the two IWBs specified, Numonics created IWBs which cost around 400 USD to 1700 USD which is indicated on their site. Their IWBs highlight electromagnetic innovation and a media pen that when the board surface is touched, it uses a full scope of cross-curricular exercises and preparing. (Branzburg, 2008).

The official website of eInstruction likewise demonstrates their IWB items. The eInstruction's Interwrite Workspace incorporates devices to make, showcase, arrange, record, and offer educational materials. It meets expectations correspondingly to the next IWB brands and offers their IWB very nearly at the same cost. (Source: einstruction.com).

Also, Resnick (2012) directed an examination that demonstrates that among well-known marked IWBs, SMART Board has pretty nearly 60.3% pieces of the pie in

USA took after by Promethean's ACTIVboard which has 24.1% and different IWBs which got under 6%.

IWB cost too much for schools to afford, many students do not have an experience of the IWB so that they do not know the benefits of the technology for their education. "People from around the world share their projects that emulates an IWB." (Rayos et al, 2014). One example of the emulated IWB was the project of Mattia Avancini. In 2009, Kinect originally known as the “Project Natal” was developed by microsoft to control and interact with the Xbox 360 without the need of the physical controller. Kim, (2009) stated that the Kinect is defined as "the birth of the next-generation of home entertainment." Avancini (2012) introduced the low-cost gesture sensor whiteboard. Since the Kinect sense the gestures of the person, he placed it on the bottom near the computer and the whiteboard. Another example of an emulated IWB was the project of Johnny Chung Lee. As stated by Rayos et al. (2014), “In 2007, millions of Wii game consoles were sold that made Wiimote one of the most common computer input device in the world. Lee (2007) introduced the low-cost multi-touch whiteboard using the Wiimote. Since the Wiimote can track sources of IR light, Lee placed an IR LED in a pen. By pointing a Wiimote at a projection screen or LCD display, you can create very low-cost IWBs or tablet displays.” (p. 11)

“Another great project is the LoCoBoard. The LoCoBoard is composed of a projector, a webcam, an IR pen, and a computer. Soares, Moreira, Torres, and Sobral (2013) used several computer vision algorithms that process captured frames and also detects IR interactions. The application is also able to adapt from certain light conditions which reduces background detection thus it will not affect IR interaction. The coordinates received is transferred to a protocol called Tangible User Interfaces Object (TUIO) that enables the user control the cursor. The coordinates received is transferred to uinput

(User Input) which commands the cursor to move to the given coordinates. The project demonstrates that with an appropriate software running in a Windows personal computer (PC), an IWB system can be produced." (Rayos et al., 2014).

Primary school students learn more with visual representations and it can be more understandable by using IWB. The teacher could present more visual presentation and games to make the class participate more and make an environment that is useful for education. IWB has been first used by the business application before it was used for educational purposes.

According to Fernandez et al. (2014) "Many schools in the United Kingdom are purchasing IWBs because teachers and parents believe that this technology will make great development to a child's learning." The advantages of IWB are the following: a PowerPoint presentation can be used to reduce the time to write the lessons on the whiteboard by the teacher and the teacher could readily use the notes to his or her students and reduce the work to write the lesson on the board. The teacher can be fun with the varieties of creative and work in class. (Beauchamp & Parkinson, 2005)

In recent times, technology have evolved dramatically and one of these technology is the IWB. It is a technology that uses a projector and a whiteboard where the contents are displayed, an LED pen that is used as a pointing device, with a camera that will detect the light and the movement of the pen and a computer where the data will be taken.

Becta (2003) (British Educational Communications and Technology Agency) indicates how IWB improved student's participation in class and its motivation to learn.

(Wall, Higgins and Smith, 2005), more appealing presentations and user friendly. IWB likewise enhances data and communication technology skills. (Cuthell, 2003).

From the opinion of the students, when they have used the IWB, they think that it was fun and enjoyable. Study shows that some students said that it was easier to learn when the lessons where displayed, it helps them to be motivated. One student said that they become independent on textbook. (Akbas & Pektas 2012).

"According to the opinions of the students, using the IWB was enjoyable and fun. Some of the students also claimed that it was quicker and easy to learn when the lessons were displayed, helping them to be motivated. A student also said that it makes students become independent on textbooks. (Akbas & Pektas, 2012, in Rayos, 2014) The result of the study shows that it encourages student participation. Many students in experimental group that uses IWB is easy to comprehend and visualize; it is a different experience from what they normally do in laboratory experiments. According to Beauchamp and Perkinson (2005) cited by Akbas and Pektas, Interactive whiteboard brings collaborative information to the classroom and teaches new learning activities.

Almost all of the classrooms nowadays make use of the Large Monitors, Cameras, dedicated computer projection system and IWB for discussions and lecture instead of using Overhead Projector (OHP). There are still disadvantages of using the IWB, it is when the teacher misuse the technology such as high cost and potential hardware failure. A project stated by Competente et al., (2014) namely "Low Cost Interactive Electronic Whiteboard using Nintendo Wii Remote" by Dalbir Singh, Ridha Omar and Azfar Anuar provides a low cost solution and utilizes the attributes of the OHP and the whiteboard.

The project uses an independent architecture that is consisted of a PC, IR/LED Pen and two Wiimotes that will serve as the camera watching for the IR pen to light. This project would require to be indoors so that the direct sunlight would not affect the performance of the Wiimote. By the use of Bluetooth, the Wiimotes send the movement signals of the IR pen to the PC.

"To test the concept of the low cost IWB using a Wiimote a software prototype called Bluesoleil was developed." (Eugenio et al., 2014) With this software, the user can control the system with the IR Pen with three interactive methods, namely pointing, pressing and releasing the switch button of the IR Pen. It will allow the user to click, drag, double click, etc. just like the functionality of the device called the mouse that is connected in the computer.

The PC must first detect the Wiimote's calibration. The user can calibrate the window using the IR Pen once it is connected. Just like the physical mouse, the calibration performs by pointing and momentarily pressing and releasing the switch button of the IR Pen to the 4 target crosshairs at the edge of the projected screen. When the window desktop appears, the calibration is complete. The user can now use the IR Pen just like the ordinary Windows setting, Icons, Menus, Pointer-based mouse.

“In its testing and evaluation, the subjects who are 15 school teachers are given questionnaires and are being directly observed. The prototype eases and aids them in their teaching process and they are satisfied with the overall functionality. The common complaint is the initial calibration process as it took a longer time than expected. The drag-and-drop feature was also unstable unlike the single and double clicks operations. It is then recommended that in the future the interface design should be improved and the addition of self-calibrating features.” (Fernandez et al., 2014)

The conclusion in a classroom setting is that the low cost IWB is feasible and provide many solutions in term of usage. A better interface would soon be developed in the future and that is the design interface and auto-calibration feature.

In late 2010, another motion sensing device was produced. The Microsoft Kinect is a device that can track motion. It detects human gestures and other body movement. It is interactive with humans. In addition with the Wiimote project done by Johnny Chung Lee, a great technology that can provide emulator for IWB is the Kinect sensor. For improvements in the future, it is suggested to provide a better and stable tool. The technology provided a low-cost IWB which is beneficial for the students according to the survey conducted by the proponents of this project.

Another alternative for a cost-effective will be according to a study entitled "Virtual Board: A low cost Multi Touch Human Computer Interactive System". According to the study, two techniques are applied when creating an IWB, namely active and passive markers. The active markers are used on expensive IWB and it provides better stability and signal while the passive marker are used in a low-cost IWB and it provides an unstable in receiving signals. An LED called the IR Led serves as an indicator of the left mouse click on the physical mouse. Two hand gloves with an IR LED on the tip of the index fingers allows hand gestures beneficial for map and image manipulation. (Lee et al, 2012)

The effects of IWB in classroom may be questioned by many. In the article called "The Art of Science of Teaching /Teaching with Interactive Whiteboards" by Robert J. Marzano, he states that the study that was performed was to see the effect of the IWB in

student achievement. The study conducted was involved by 85 teachers and 170 classrooms. A specific set of lessons was taught by the teacher using IWB in a group of students, and teaching the same lesson without the IWB with another group of students.

According to study, a classroom with IWB, each student would point gain 16% in their student achievement. Using visual aid and graphics to represent the information in class provides a 26% gain in student achievement. One disadvantage would lead to misuse of technology by the teacher. In the end, the teacher must still use the IWB thoughtfully to bring the best out of the students.

The decline of skill in applying students for Computer Science was the problem that they saw. The availability of PC's and game console nowadays replaced the need to learn programming back in earlier computers like the Amigas, BBC Micros, Spectrum ZX, and Commodore 64. The decline of skill in applying students for Computer Science was the problem that they saw. The availability of PC's and game console nowadays replaced the need to learn programming back in earlier computers like the Amigas, BBC Micros, Spectrum ZX, and Commodore 64. A low-cost platform will be used for experiment that will be good for children to learn programming.

The uinput is a Linux Kernel module which can input devices such as the mouse and the keyboard which can also handle them from an application. All devices are initialized as files in a Linux OS. The creation of input devices will make use of the uinput module. A character device file will be created in Linux and will be seen in /dev directory. The device will act as a virtual device and will not act as a physical device.

In 1979, C++ programming language when Bjarne Stroustrup was doing work for his Ph.D. thesis. It is a high-level programming language created by Bjarne Stroustrup at Bell Labs. C++ is the basic programming language for those who want to pursue in the computer field.

“C++ is a mixture of the C language and object-oriented programming. It lets you create your own function, which is useful for when you need your own function most of the time. It also has inheritance like Java. C++ is mainly used for software engineering and graphics.” as stated by Rayos et al. (2014).

In 1972, Dennis Ritchie with Ken Thompson who co-developed, created the C programming language. C programming language is a general purpose which has been closely associated with the operating system, namely UNIX. It has been closely associated for which most of the programs are written in C.

OpenCV was designed for computational efficiency with a strong focus on real time applications. OpenCV will be used in this project. It is the chosen tracker application that will track the LED pen. "The protocol encodes the data obtained from the tracker application and sends it to the client application who will then decode it, the result is then shown in the projected screen." (Rayos et al., 2014).

“From a software point of view the Kinect software technology is developed internally by Rare, a subsidiary of Microsoft Game Studios owned by Microsoft. In December 2010, as reported in [16], PrimeSense released their own open source drivers, along with motion tracking middleware called “Natural Interaction Technology for End-user (NITE)". PrimeSense create with the collaboration of two other companies a not-for-profit organization focused on certifying and improving interoperability of Natural

Interaction (NI) devices called “Open Natural Interaction (OpenNI)". OpenNI is not only an organization but also a cross-platform framework that provides the interface for both physical devices and middleware component.” (Avancini, 2012)

OpenNI or Open Natural Interaction is an open source software project that focuses on certifying and improving interoperability of natural user interfaces for Natural Interaction (NI) devices and applications that use those devices and middleware that facilitates access and use such device. OpenNI framework provides a set of open source API. These API supports voice and voice command recognition, hand gestures and body motion tracking. These are intended to become standard applications to access natural interaction devices. The API framework itself is sometimes referred to the name OpenNI SDK. (Source: http://en.wikipedia.org/wiki/OpenNi)

Windows 8 comes up with different input types. Windows 8 multi-touch screen devices enables users to use multiple fingers to simultaneously produce different interactions such as tapping, dragging or pinching. (Source: http://social.technet.microsoft.com/wiki/contents/articles/6460.simulating-touch-input-in-windows-8-using-touch-injection-api.aspx)

The WM_PAINT message is sent when the framework or another application makes a solicitation to paint a bit of an application's window. The message is sent when the UpdateWindow or RedrawWindow capacity is called, or by the DispatchMessage capacity when the application gets a WM_PAINT message by utilizing the GetMessage or PeekMessage capacity. (Source: https://msdn.microsoft.com/en-us/library/windows/desktop/dd145213(v=vs.85).aspx)

Multi-Touch Vista (UNISoftHID) is a client information administration layer that handles data from different gadgets (touchlib, numerous mice, TUIO and so forth.) and standardizes it against the scale and turn of the objective window. Presently with multi-touch driver for Windows 7. (Source: https://multitouchvista.codeplex.com/)

The Kinect for Windows SDK beta provides hobbyists and researchers with the tools to develop non-commercial applications that run on the Kinect for Xbox 360. “Kinect for Windows brings your apps to life through natural human computing. Find videos, downloads, and docs to help you start developing solutions that respond to a user's gestures and voice.” (Source: https://www.microsoft.com/en-us/kinectforwindows/develop/learn.aspx)

Microsoft Kinect is important in this project. Kinect is codenamed as Project Natal. It is a line of motion sensing input devices by Microsoft Xbox 360 and Xbox One Consoles and windows PC. It is webcam styled with add on peripheral, it enables users to control and interact with their console/computer without the need of game controller. It can detect the motion gestures of the user.

“Microsoft released the Kinect software development kit for Windows 7 on June 16, 2011. This SDK was meant to allow developers to write Kinecting apps in C++/CLI, C#, or Visual Basic .NET.” (Source: http://en.wikipedia.org/wiki/Kinect)

"Decent applications don't draw directly on desktop - it will cause problems, because OS will keep repainting portions of desktop, destroying the image." – Sig Term (Source: http://stackoverflow.com/questions/13492860/how-to-draw-on-the-windows-desktop-using-the-gdi-api)

"You don't own the desktop window, so you'll always have problems with invalidation and repaints." - Adrian McCarthy

(Source: http://stackoverflow.com/questions/13492860/how-to-draw-on-the-windows-desktop-using-the-gdi-api)

Table 2.1 LED Pen Detection and Cursor Control Summary
(Competente et al, 2014, p.41)

Lighting Condition	Distance			
	0.5 Meter	1 Meter	1.5 Meters	2 Meters
Dark Environment	Failed	Pass	Pass	Pass
Low-light Environment	Pass	Pass	Failed	Failed
Regular Classroom	Failed	Failed	Failed	Failed

“There are various lighting conditions which the proponents considered for testing the accuracy of the LED Pen for controlling the cursor. In Table 4.1, the results showed that a dark environment is the best option for the lighting condition when using the LED pen. Further tests regarding with lighting condition is presented in Appendix E from test case numbers 3 up to 27.” (Competente et al., 2014. p.42)

Synthesis

Some schools and businesses corporation make use of the IWBs. The popular whiteboard marker was used for writing in the whiteboard; it can be easily erased by applying friction on the written area. The IWB functionality is almost the same; it's just that the IWB is digital and projects the screen through an LCD projector facing the whiteboard. It is capable of writing and controlling anything on the board virtually. Instead of using a regular computer, the proponents will use a laptop or desktop computer, Kinect sensor and an LED Pen. They will be using the programming language OpenNI/NITE, Kinect SDK for Windows and C++.

CHAPTER 3

TECHNICAL BACKGROUND

Methodology

The methodology Rapid Prototyping will be used for a faster approach in developing the project as shown on Figure 3.1.

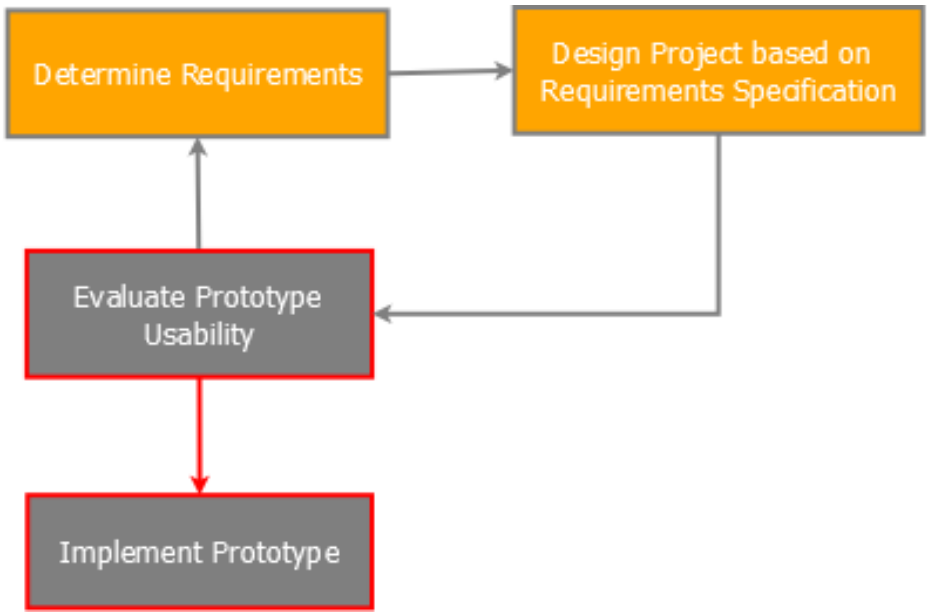

Figure 3.1 Rapid Prototype Modelling

Determine Requirements

In determining requirements, the proponents will be studying Kinect SDK, OpenCV, OpenNI libraries particularly about on Color Tracking, Motion Tracking and Mouse Events that will be implemented on the programming language C++. Hardware components like the Kinect sensor, computer, LED pens and other peripherals will be reviewed to check if it will meet the specific requirements needed to develop the project.

Design Project based on Requirements Specification

First, the LED pens will be designed ergonomically and be made through the use of an RGB LED, resistors and batteries. It will be designed to provide the required colors needed for the color tracking. Second, the computer to be used will be set-upped for coding by installing the required drivers and software. Hardware components will be tested to ensure that devices, especially the Kinect sensor, are working properly. Lastly, codes will be applied and tested on a Windows platform particularly using the Kinect SDK.

Specifically motion detection of the user arm movements will undergo to coding and testing to ensure that it will provide an accurate and precise motion tracking to its x and y movement, in correspond to the UI cursor on the computer. LED Pens will be implemented after with the integration of color tracking. It will be used for the customizable input particularly for Mouse Events action. Multi-user implementation will be next. Providing another cursor input on the UI that will be manipulated by another user's movement and LED pens.

Evaluate Prototype Usability

Evaluation of the overall system integration will be conducted through test-cases to check if the system is working functionally as expected and is usable to any typical classroom lightning conditions. The development of the prototype/s will be continuous until the desired output has been done.

Implement Prototype

Once the desired output of the prototype has been achieved, the functions and processes installed will be implemented permanently and will be polished for better

results and performance. The prototype will be used in certain educational applications to prove its use.

Requirements Analysis

Functional Requirements are as follows:

- A Kinect™ for Xbox 360 device will be used for motion and color tracking for the system. The Kinect™ sensor consists of the following:
    - 3-D depth sensors
    - RGB Camera
    - Multiple Microphones
    - Motorized Tilt
- 2 customized RGB LED are needed for the customizable input. Each individual pens will produce different colors.
    - User 1: Red, Green, Blue
    - User 2: Cyan, Magenta, Yellow
- A LCD projector is required for the display of the output
- A desktop computer or laptop that meets the minimum requirements shown on Table 3.1

Non-Functional Requirements are as follows:

- Kinect™ for Windows SDK 1.8 will be used in developing the system.
- C++ is programming language will be used for the project development.
- OpenCV, OpenNI nad NiTE libraries will be utilized for the motion and color tracking
- Windows OS will be used as a platform for the development of the whole system.

Overview

System Requirements

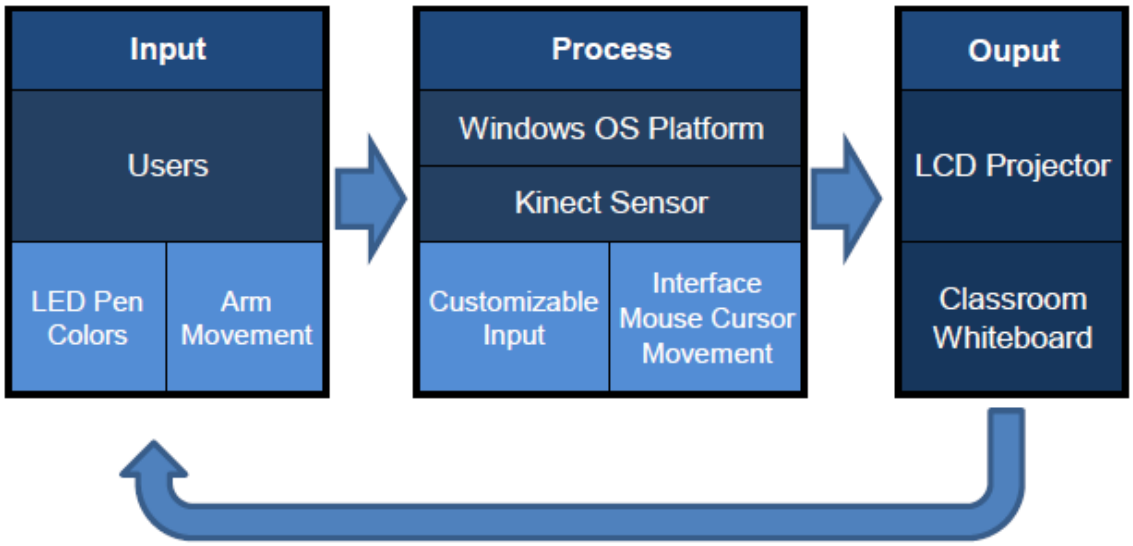

Figure 3.2 Input-Output-Process Chart

As presented on Figure 3.2, the input will be User's LED Pen light colors and its arm movement which corresponds to a chosen mouse cursor behavior and movement respectively. The input will be tracked by the Kinect sensor which is installed to a Windows OS platform. The mouse events will be displayed on the LCD projector in a classroom.

Table 3.1 Kinect for Windows SDK 1.8 Requirements

Kinect for Windows SDK v1.8 Requirements	
Supported Operating System	Windows 7
	Windows 8
	Windows 8.1
	Windows Embedded Standard 7
Hardware Minimum Requirements	32-bit or 64-bit Processor
	Dual-core 2.66-GHz or faster processor
	Dedicated USB 2.0 bus
	2 GB Ram
	Microsoft Kinect for Window Sensor
Software Requirements	Visual Studio 2010 / Visual Studio 2012
	Microsoft Speech Platform SDK v11 (for speech-enabled Kinect Windows Application)

Shown in Table 3.1 are the requirements needed to install the Windows SDK 1.8 It is a package needed to be installed to be able to use Kinect sensor and develop applications on it.

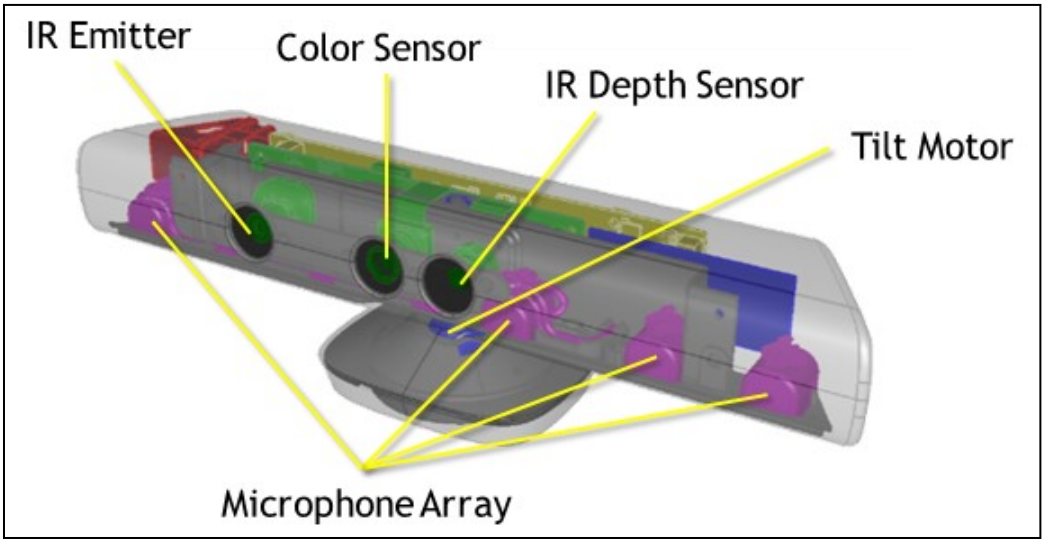

Figure 3.3 Kinect Sensor Components ("Kinect for Windows Sensor," 2015)

Shown on Figure 3.3 are the components of a Kinect Sensor which includes IR Emitter, Color Sensor, IR Depth Sensor, Tilt Motor and Microphone Array. Detailed specification for each sensor and on the Kinect itself is shown on Table 3.2 and Table 3.3 respectively.

Table 3.2 Sensors Specification

Sensors Inside the Kinect		
Sensors	Specific Details	Summary
RGB Camera	Stores three channel data in a 1280x960 resolution	Makes capturing a color image possible.
Infrared Emitter and IR Depth Sensor	The emitter emits infrared light beams and the depth sensor reads the IR beams reflected back to the sensor	Makes capturing a depth image possible.
	The reflected beams are converted into depth information measuring the distance between an object and the sensor.	
A Multi-array Microphone	Contains four microphones for capturing sound.	Record audio as well as find the location of the sound source and the direction of the audio wave.
3-axis Accelerometer	Configured for a 2G range, where G is the acceleration due to gravity.	Determines the current orientation of the Kinect.

Table 3.3 Kinect Specification

Kinect Specification	
Kinect	Array Specification
Viewing angle	43° vertical by 57° horizontal field of view
Vertical tilt range	±27°
Frame rate (depth and color stream)	30 frames per second (FPS)
Audio format	16-kHz, 24-bit mono pulse code modulation (PCM)
Audio input characteristics	A four-microphone array with 24-bit analog-to-digital converter (ADC) and Kinect-resident signal processing including acoustic echo cancellation and noise suppression
Accelerometer characteristics	A 2G/4G/8G accelerometer configured for the 2G range, with a 1° accuracy upper limit.
Kinect depth sensor range	minimum 800mm to maximum 4000mm.

Table 3.4 LED Pen Specification

Specification	LED Pen	
Users	User 1 LED Pen	User 2 LED Pen
Colors Produced	Red, Green, Blue	Magenta, Yellow, Cyan
Materials	2 RGB LED	
	6 Switches	
	4 Coin Cell Battery	
	Wires	
	Resistors	
	Attiny 2313-20pu	
Prototype Image		

Shown on Table 3.4, is the specification of the LED pen to be used for color tracking. Two LED pens will be made producing different colors for first user (Red, Green and Blue) and the second user (Cyan, Magenta, and Yellow). It will be made mainly with the use of an RGB LED.

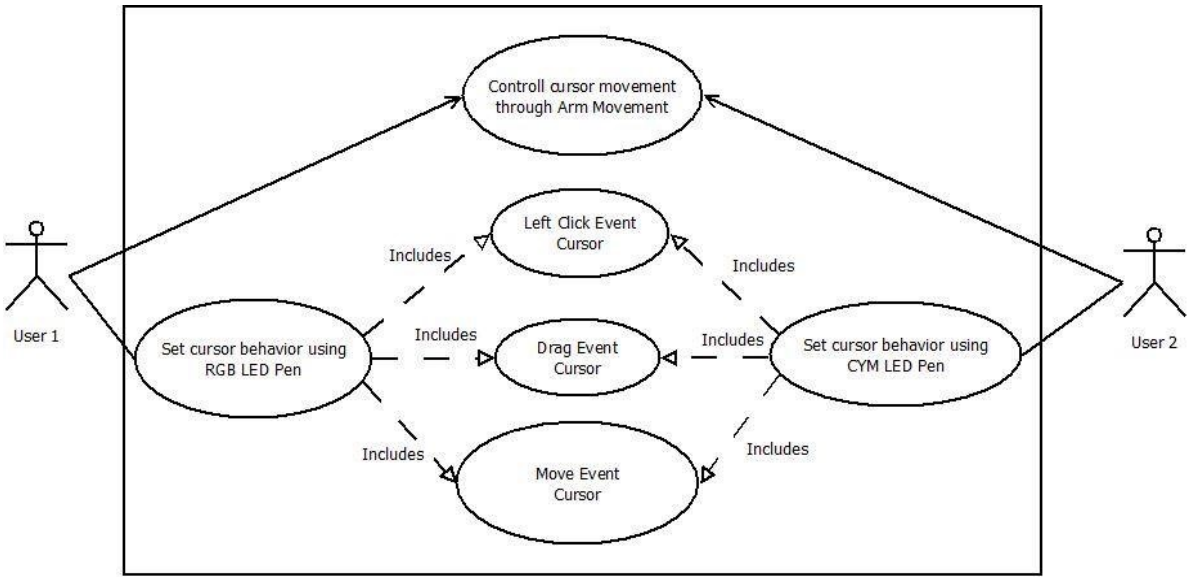

Figure 3.4 FPS Interactive Whiteboard Use Case Diagram

Figure 3.4 Represents 2 users can control two separate cursors simultaneously through arm movement and using two different LED pens. The arm movement is used for tracking the cursor movement. On the LED pens, one user will be using the RGB LED Pen and another is CYM LED pen for a separate customized input which on this project is for different mouse cursor behavior.

Prototype

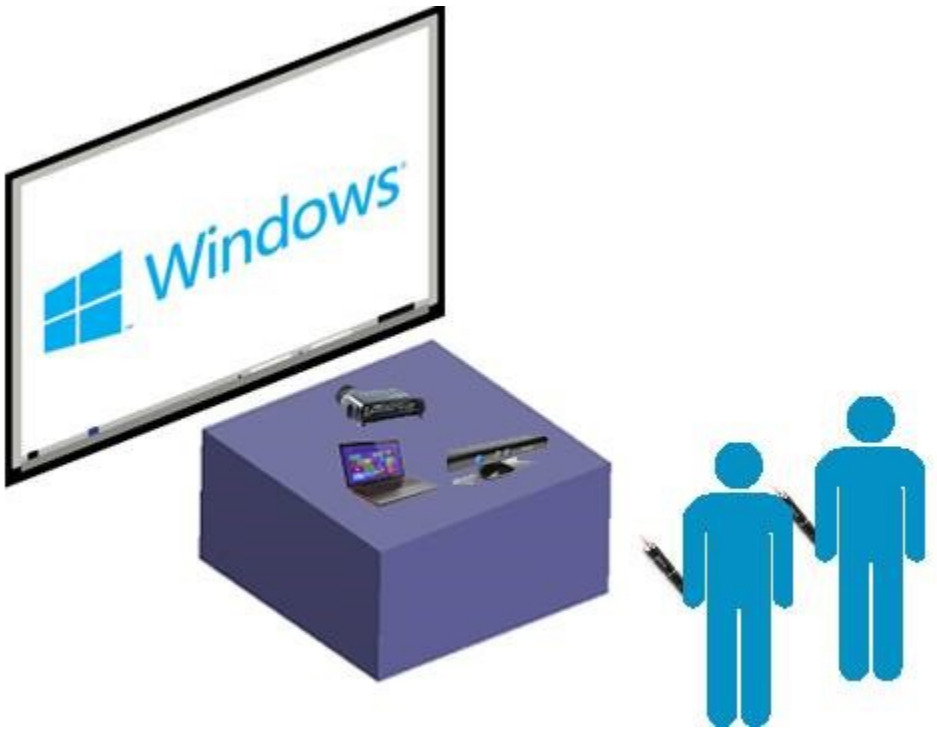

Figure 3.5 Overview of the FPS Interactive Whiteboard

The figure 3.5 shows the overview of the project in which the Kinect sensor facing the users is connected to a Windows OS platform. The Kinect is set-upped and ready to track accurately the user's arm movements and LED pen color to be able to interact on the different applications on a lighted environment.

CHAPTER 4

IMPLEMENTATION, RESULTS, AND DISCUSSION

Requirements Documentation

Design Methodology

The proponents implement the Capstone project to produce a prototype of an Interactive Whiteboard with a Laptop or Desktop with a Windows 8 Operating System installed for testing, configuring, and programming the Kinect that is specifically designed to detect motion and 2 LED pens made and designed by the proponents. The program for motion input used the C++ programming language, along with the OpenCV, OpenNI and NiTE libraries. Each pen has an LED that produces 3 different color, the first pen produces the color of red, green and blue and the second pen produces the color of cyan, magenta and yellow of which function as a move for cursor movement, click and drag, respectively. These lights are detected by the Kinect and the program would then know what the user intends to do. The monitor of the Laptop will show the output of the movement of the arrow cursor.

The Kinect will be connected in the laptop for the program to work and it will detect the users' movements. Once the users are identified you can press a button on the LED pen and the Kinect will identify the color that is pressed and it will do its function (move, click or drag).

Use Case Diagram

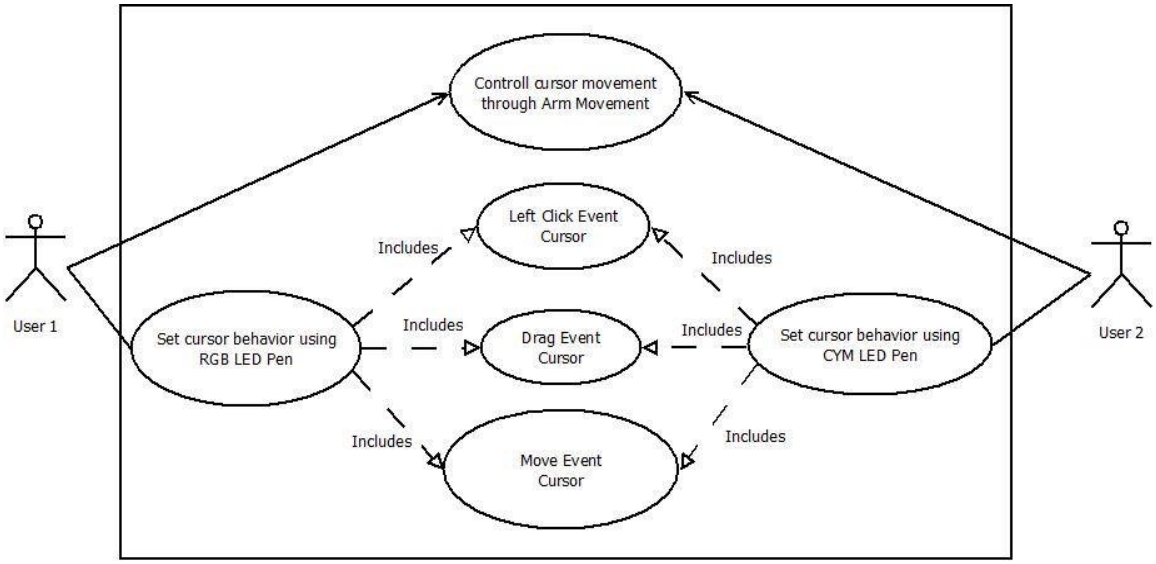

Figure 4.1 FPS Interactive Whiteboard Use Case Diagram

Figure 4.1 Represents 2 users can control two separate cursors simultaneously through arm movement and using two different LED pens. The arm movement is used for tracking the cursor movement. On the LED pens, one user will be using the RGB LED Pen and another is CYM LED pen for a separate customized input which on this project is for different mouse cursor behavior

Hardware Specifications

Figure 4.2 Laptop

Any laptop model that runs with windows 8 O.S with a minimum requirement of 32-bit or 64-bit processor, Dual-core 2.66-Ghz or faster processor, Dedicated USB 2.0 bus and a 2GB Ram. Other Laptops that has lower specifications that the indicated minimum requirement will cause the program to run slow and irresponsive.

Figure 4.3 LCD Projector

An LCD project like in Figure 4.3 is used for the display of the program. Likewise any monitor / TV will also work for output display.

Figure 4.4 Xbox 360 Kinect

Figure 4.5 Xbox 360 AC Adapter / Power Supply

The Xbox 360 AC Adapter / Power Supply shown in Figure 4.5 is used for connecting the Xbox 360 Kinect shown in Figure 4.4 to the computer system.

Main Electronic Parts

Figure 4.6 Diffused RGB LED

The diffused RGB LED show in Figure 4.6 is used to produce different colors in one LED bulb.

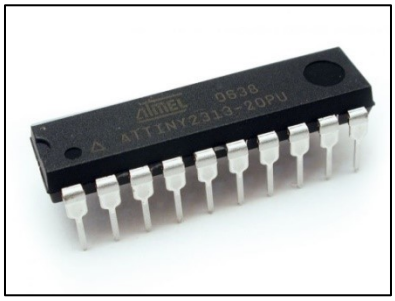

Figure 4.7 Attiny 2313-20pu IC

The Attiny 2313-20pu IC show in figure 4.7 is used to code the program logic for the pen to produce the desired colors.

LED Pen Prototype:

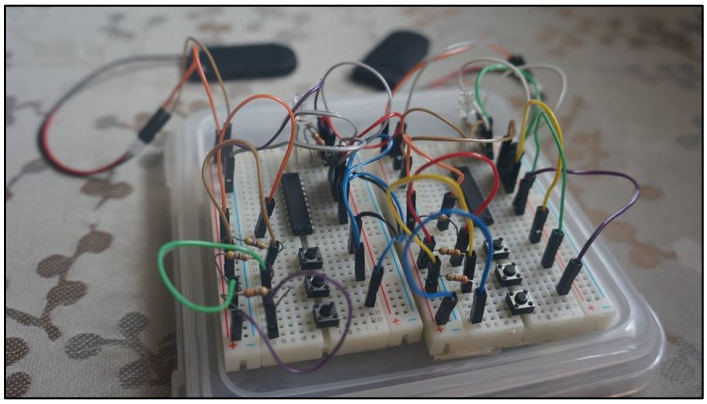

Figure 4.8 Initial Prototype of LED Pen circuit board

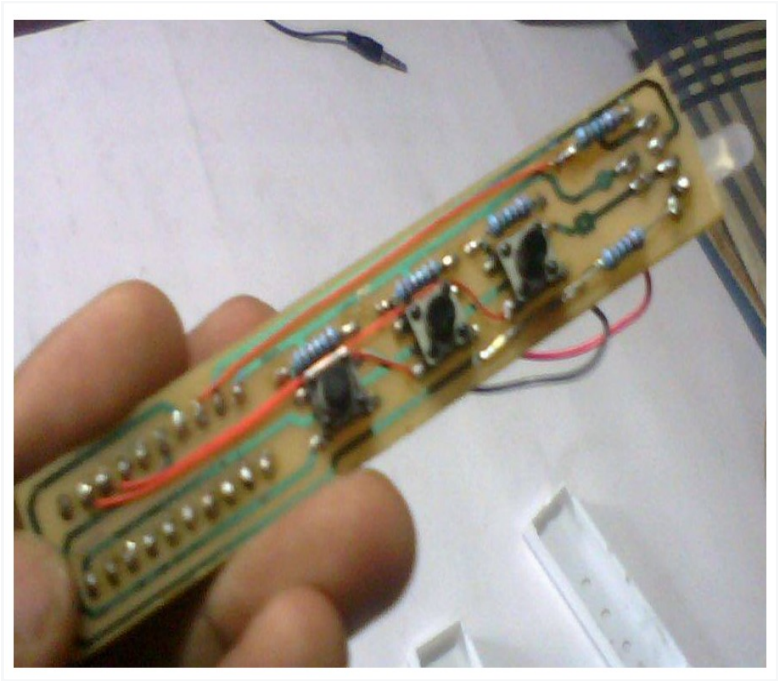

Figure 4.9 Final prototype of the LED Pen circuit board (front view)

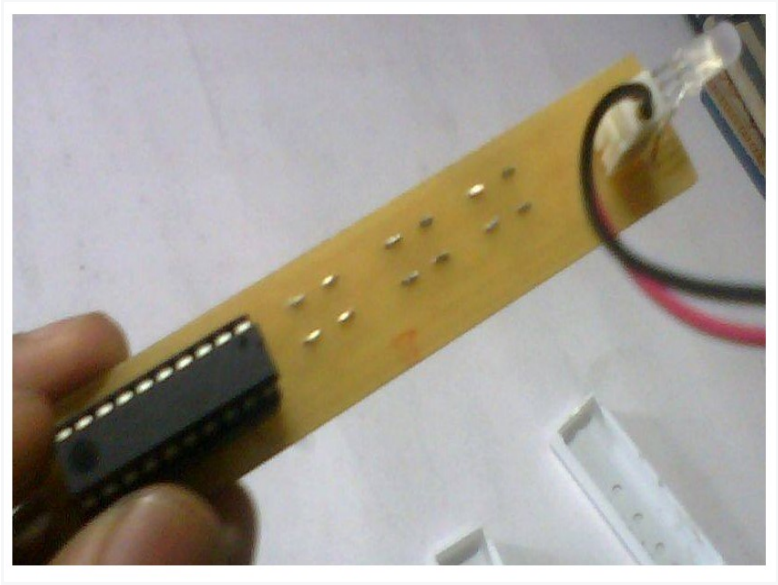

Figure 4.10 Final prototype of the LED Pen circuit board (back view)

.

Shown in figure 4.14 and 4.15, the proponents decided to make a customized circuit board for the LED Pen to have a compact and more organized circuit making it sturdy by preventing the wires from snapping and entangling. It is mainly composed of an Attiny 2313-20pu IC which is shown in figure 4.12, a diffused RGB LED which is shown in Figure 4.11, 6 220 Ohms resistors and 3 push-down switches and is connected to a coin cell battery holder that act as the power supply.

Procedure:

1. Sketch the schematic diagram of the circuit to CadSoft Eagle 6.6.0, a PCB Fabrication software.
2. Design the board manually guided by the auto route feature of the software.
3. Print the board design in black ink to a tracing paper.
4. Peel off the white screen of the pre-sensitized PCB and align/attach the printed design to the green part. The green layer on the copper board is a light sensitive element which can be developed or removed when exposed to light and dumped into a developer solution.
5. Expose the board to a light source, facing the design/copper board for at least 10 minutes.
6. Mix the whole developer granules to 1 liter water to make the solution.
7. Detach the design/tracing paper to the board and soak it to the mixed developer solution.
8. Once the design is already on the board, rinse it with water and etch the board on a ferric chloride.
9. Wipe your board to dry it then drill the holes of the design.
10. Place the components on the board, following the placement on the design shown in figure 4.6
11. Solder the pins and cut the excess parts of the components pin.
12. Place the circuit board to the chassis and do the wiring.

Materials:

Table 4.1 Materials used for the Final Prototype of the LED Pen Circuit Board

Material	Qty. (Per circuit board)
ATTiny2313 Programable Microchip	1
20 pins IC Holder	1
220 Ohm Resistor	5
4x6 Presensitized PCB	1
Diffused RGB LED	1
Soldering Iron	1
At least 5ft. Soldering Lead	1
4 Pin tact Switch	3
2 male and female header pins	1
Coin Cell Battery Holder	1
3v Coin Cell Battery	2

Figure 4.11 LED Pens

The encased LED pens is shown in figure 4.11. It includes the circuit board in figure 4.9 and figure 4.10, a casing, a reflector, a lens cap and a 3v battery holder. The blue LED pen produces the colors red, green and blue while the pink LED pen produces the colors cyan, yellow and magenta.

Other Peripherals:

Figure 4.12 3V Coin Cell Battery

Figure 4.12 shows a pair of 3V Coin cell battery which acts as power supply for the LED Pens shown in Figure 4.11.

Figure 4.13 GizDuino Mega

The GizDuino Mega shown in figure 4.13 is the hardware used by the proponents to program the Attiny 2313-20pu IC which is placed inside the LED pens in figure 4.16 to produce the desired color of the LED bulb.

Software Requirements

Table 4.2 Software Requirement for the Kinect Program

For Kinect Program	
IDE	Visual Studio 2010 Express C++
Programming language	C++
Open Source Libraries	Open CV 2.4.11
	OpenNi 2.2
	Nite 2.2
Windows Utility / Library Used for C++	Touch injection API on windows 8
	WM_Paint Message
Drivers Used	Driver part of Kinect SDK 1.8 (Official Microsoft SDK)
	UnisoftHID

Table 4.3 Software Requirement for Attiny2313 program uploading

For ATTINY2313 Coding	
IDE	Arduino IDE
Programming Language	C
Drivers Used	PL2303 Prolific Driver
Other Peripherals	Win AVR
	GizDuino

Design of Software, Systems, Product and/or Processes

Flow Chart

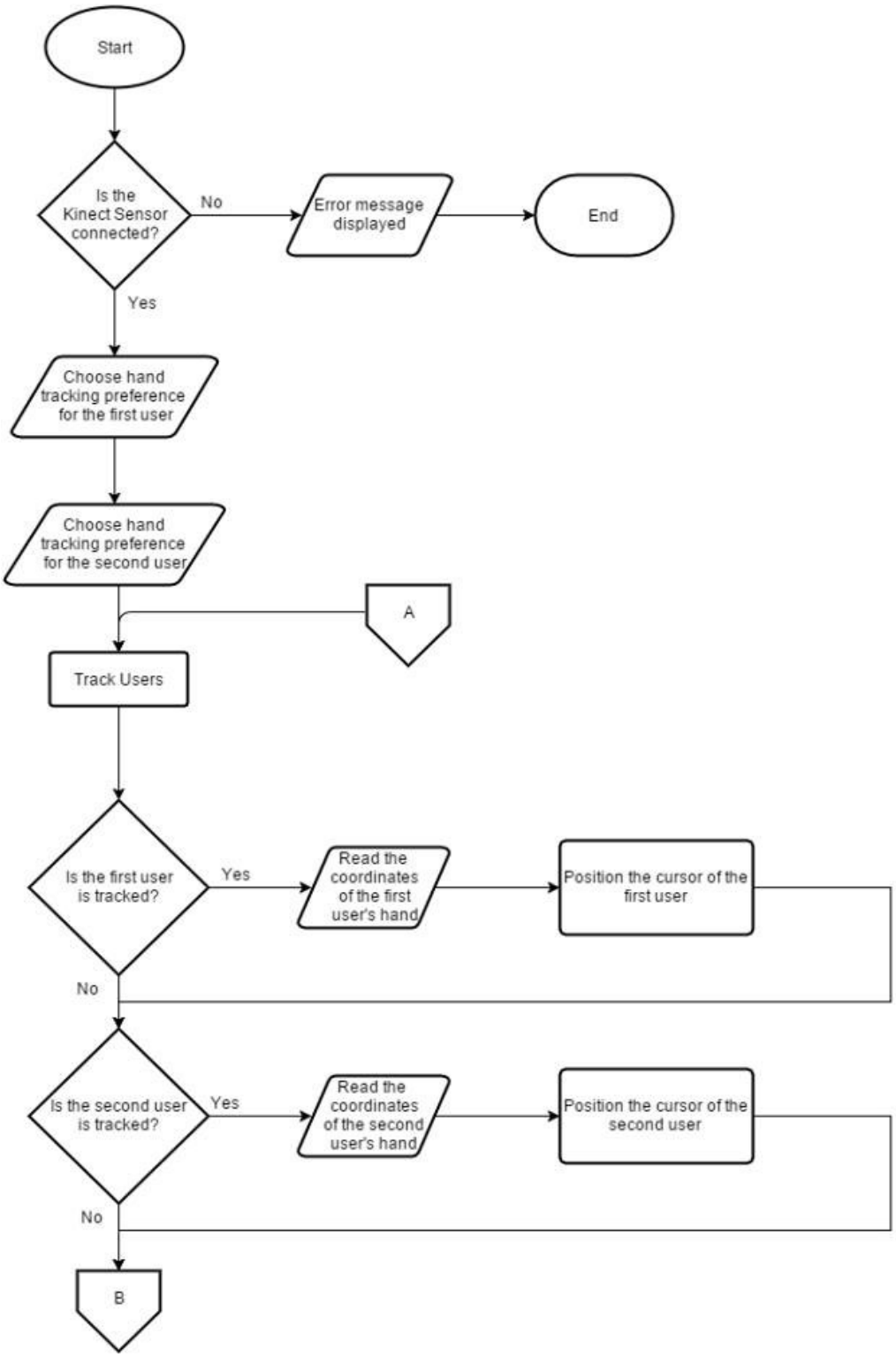

Figure 4.14 Initial flow for launching the program

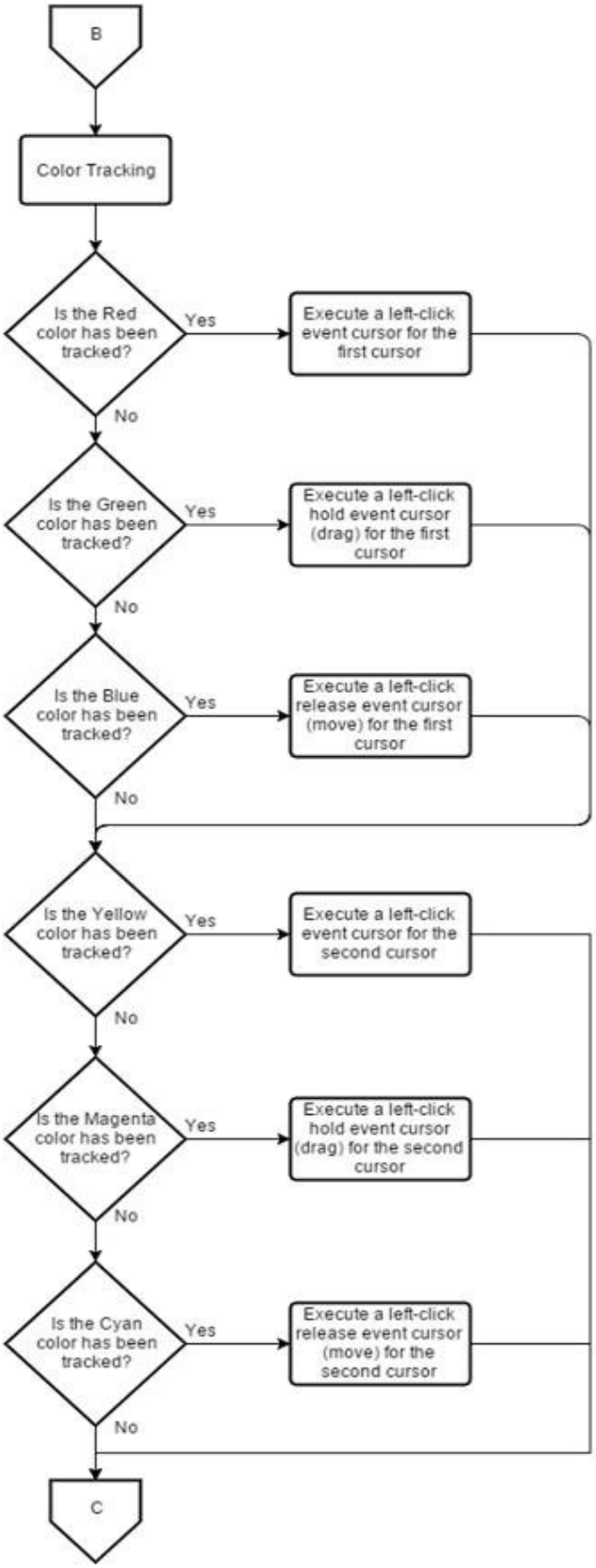

Figure 4.15 Flow chart for color tracking and function

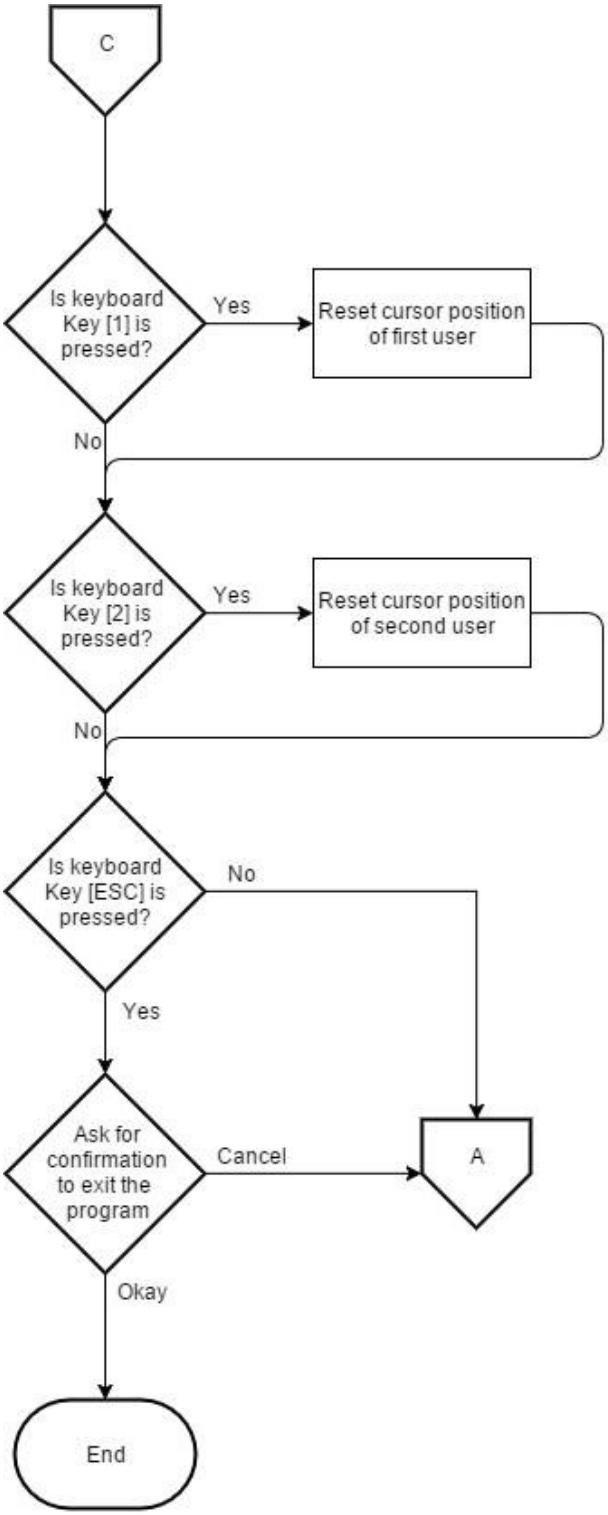

Figure 4.16 Flow chart for resetting the user position and closing the program

Block Diagram

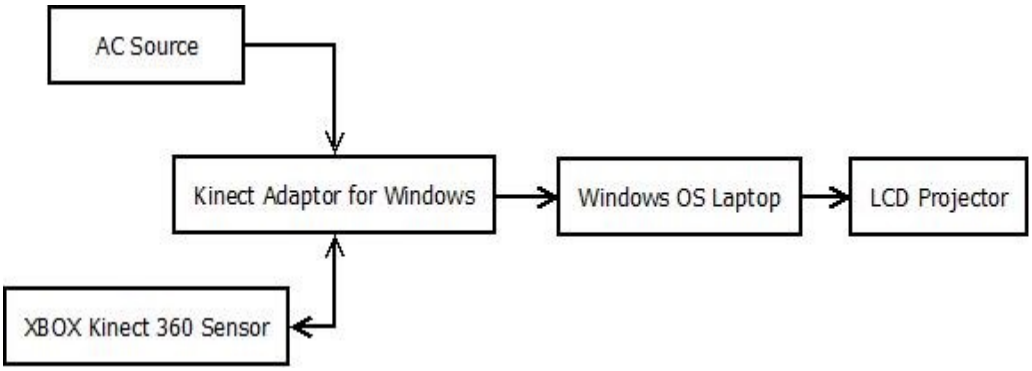

Figure 4.17 Kinect Block Diagram

Figure 4.17 The figure shown above shows that the Kinect is connected to the laptop through the Kinect Adapter, in which is needed to provide sufficient power source for the Kinect that the USB power output can't supply. The display will be output using an LCD projector into the classroom whiteboard.

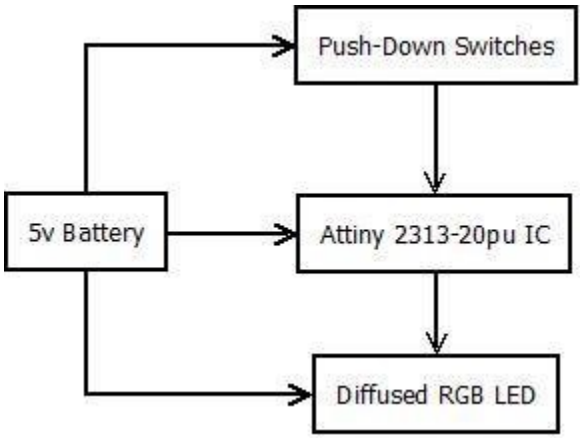

Figure 4.18 LED Pen Block Diagram

Figure 4.18 The figure shown above shows the connection between the individual electronic parts. The switches act as an input while the Diffused RGB

LED will provide the desired color output. In between them is the IC that will provide the program logic that will process which color will be displayed depending on what switch is pressed. And all individual electronic parts are powered by 5v battery source.

Schematic Diagram

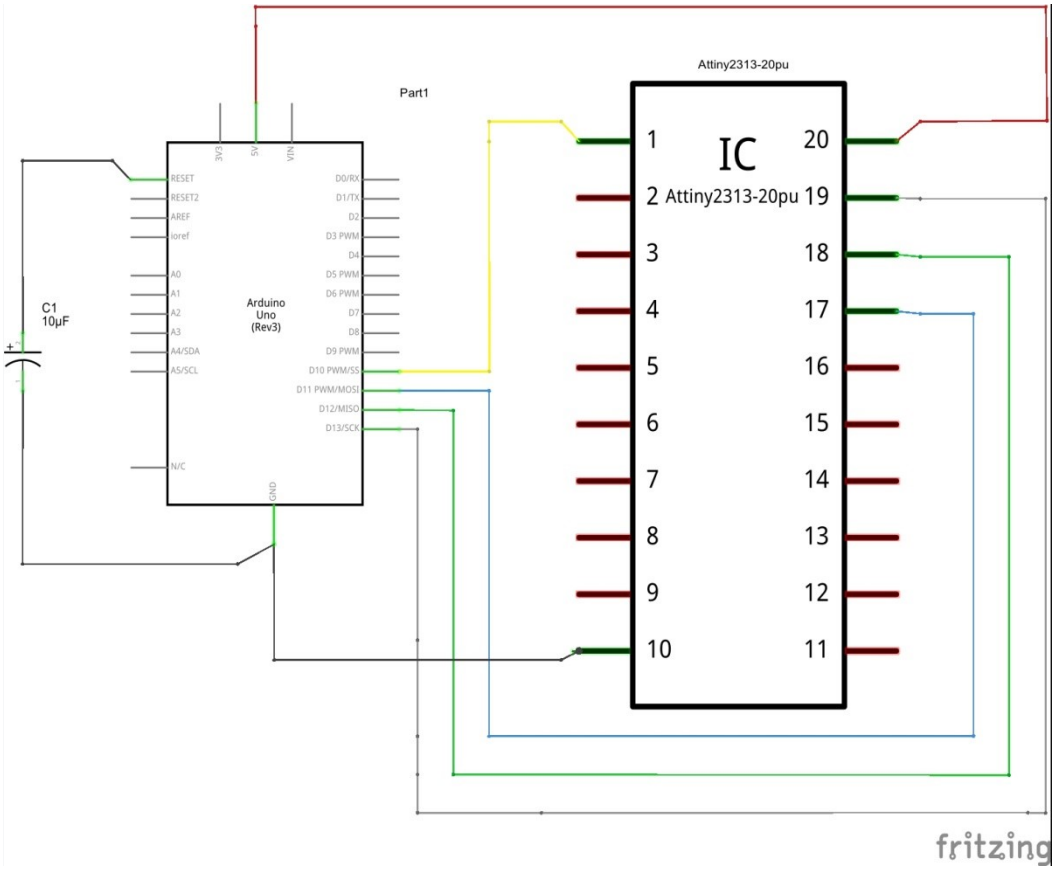

Figure 4.19 Schematic Diagram on uploading the program to the Attiny2313-20pu

Figure 4.19 The figure shown above is the circuit for uploading the Attinty2313-20pu using the Arduino/Gizduino Mega. It also requires 10 microfarad capacitor during the program uploading process.

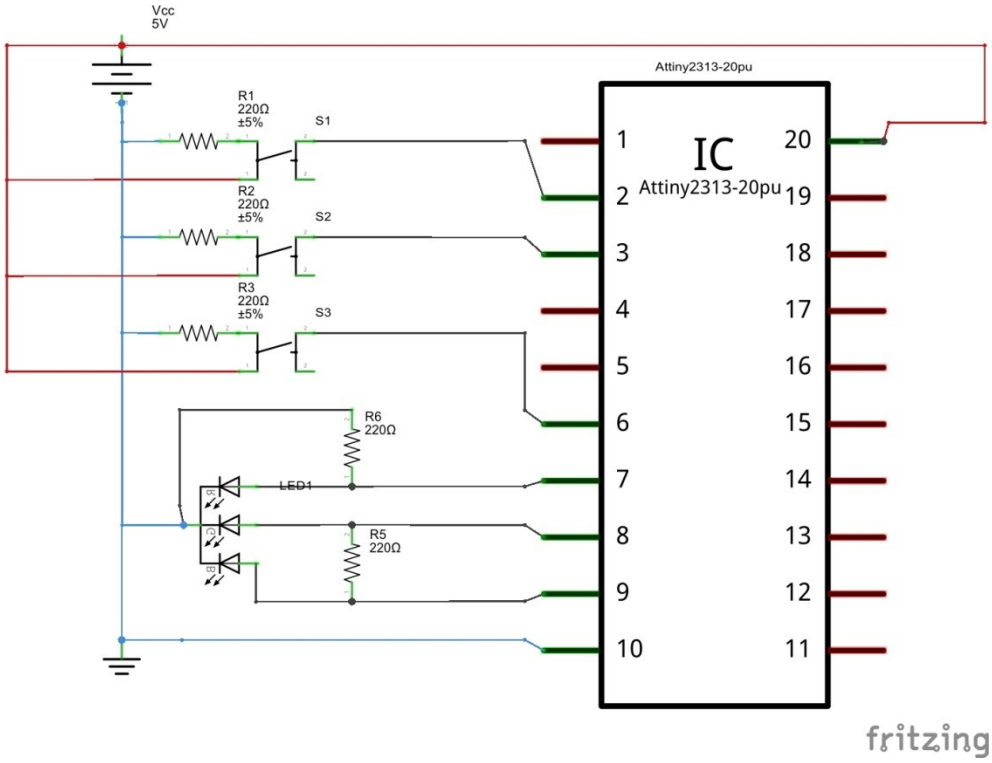

Figure 4.20 LED Pen Schematic Diagram

Figure 4.20 The figure shown above is the circuit of the LED Pen. It comprises of 3 switches, 5 220 Ohms resistor, 5v dc source, an Attiny2313-20pu IC which provides the program logic and a diffused RGB Led. Both LED pens had the same circuit but has different program logic stored on the IC. The first pen provides three colors red, blue and green while the second pen provides colors yellow, magenta and cyan for each switch respectively.

Circuit Board Diagram

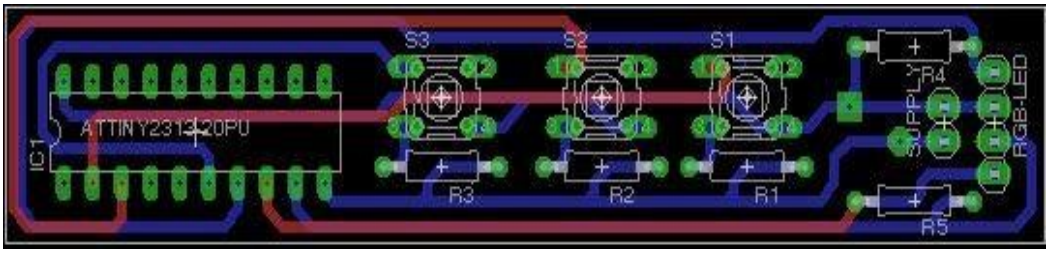

Figure 4.21 LED Pen Circuit Board

Figure 4.21 The figure shown above is the LED Pen Circuit Board design which is used the layout the schematic diagram into the circuit board in Figure 4.9 and Figure 4.10 inside the LED pens casing in Figure 4.11.

Development and Testing

The prototype FPS interactive whiteboard development is focused mainly on the efficiency of the LED pens for cursor control behavior. Other module includes the flow of the main program and its interaction to the Kinect's tracking process. The test cases produced by the proponents can be found in Appendix C.

Table 4.4 Color Tracking Test Summary tested without the presence of sunlight

Color Tracking Test Summary				
Lightning Condition	Distance			
	0.5 Meter	1 Meter	1.5 Meter	2 Meter
Dark Environment	Pass	Fail	Fail	Fail
Low-Light Environment	Pass	Pass	Pass	Fail
Regular Classroom	Fail	Pass	Pass	Fail
LED pens extended with 4 diameter aluminum reflector				
Dark Environment	Fail	Fail	Fail	Fail
Low-Light Environment	Pass	Pass	Pass	Pass
Regular Classroom	Fail	Pass	Pass	Pass

Table 4.5 Color Tracking Test Summary with the presence of sunlight

Color Tracking Test Summary with the presence sunlight				
Special Condition	Affected by the sunlight coming from the windows and glass doors			
Lightning Condition	Distance			
	0.5 Meter	1 Meter	1.5 Meter	2 Meter
Dark Environment	Pass	Pass	Fail	Fail
Low-Light Environment	Pass	Pass	Fail	Fail
Regular Classroom	Pass	Fail	Fail	Fail
LED pens extended with 4 diameter aluminum reflector				
Dark Environment	Pass	Pass	Pass	Fail
Low-Light Environment	Pass	Pass	Pass	Fail
Regular Classroom	Pass	Pass	Fail	Fail

There are various lightning conditions which the proponents considered for testing the accuracy of the LED Pen for controlling the cursor behavior. In Table 4.4, shows the tests conducted without the presence of sunlight. While in Table 4.5 shows the tests conducted with the presence of sunlight coming from the windows and glass doors. The test proves that a larger reflector provides better distance for using the system. Setting aside all the conditions, low-light environment is best option for using the system. This is a summary of test cases C01 to C68.

Table 4.6 Tracking Time Test Summary

Tracking Time Test Summary	
User	Ave. Time
First User	4.21 sec
Second User	7.06 sec

The table 4.6 shows the summary of the tracking time for the users to be able to move the individual cursor position. Time is started to be measured once the touch cursors appear. We measured the time using a stopwatch. This is a summary of test case T05.

Table 4.7 Special Locations and Conditions Test

Testing the system on a special locations and conditions				
Scenario				Results
Location	Lightning Condition	Area Condition	Other Conditions	
ICS Lab 2 (University of Santo Tomas, Roque Ruaño BLDG, 3F)	24 Fluorescent lamp (all switched-on)	1.9m by 7.5m	Students 2m away	Excessive light on the computer laboratory weakens the color tracking range of the system
Dev Room (University of Santo Tomas, Roque Ruaño BLDG, 2F)	3 Fluorescent lamp	8.5m by 9.5m	Professors 2m away	The walls of the room bounces the infrared light scattered by the Kinect's IR Emitter making the user tracking inaccurate.
				The yellow walls of the room is very closed that it also detected by the color tracking

Seen on Table 4.7 is the test result of test case S01, showing the results when the system was tested on different locations and its conditions, other than the typical classroom setting.

Table 4.8 Usage of WM_Paint Message to the Windows Desktop Test

Usage of WM_Paint Message to the Windows Desktop Test			
Scenario:	Check the graphics interaction of the WM_Paint message to the Windows OS ability to refresh the desktop image.		
Graphics Category	Description	Problems Occurred	conclusion
Testing Animated Graphics	Graphics that moves in real time	Smudges the desktop image	Not usable
		Slows down the program	
Testing Still Graphics	Still graphics shown on the entire run of the program	Graphic flickers on dynamic frames opened on the desktop	Can be used only on static window frames like MSpaint
		Still graphics must be ran to a separate executable program or else it will slow down the main program	
Testing Pop-up Graphics	Single occurrence once per user action	N/A	Usable

On the earlier versions of our program, we used WM_Paint message to provide information and border through drawing on the top layer of the Windows desktop. Upon research and testing the proponents have concluded that it is not advisable to draw on the top of desktop particularly animated and still graphics (borders, instructions etc.), since the windows OS constantly refreshes the desktop image. It causes smudges and flickers on the screen and also slows downs the tracking of the program which is crucial since the program run on real time. Results are from test case S03, shown on Table 4.8.

On the final version of the program, WM_Paint message is still used to provide graphic feedback for user and color tracking information, since it only occur in a certain user event and doesn't affect the usability and speed of the program.

Table 4.9 Program Speed Test

Program Speed Test				
Scenario	Program Date Version	Trial	Ave sec. per iteration	Total Average (sec)
Testing the program speed	4/27/2016	1	0.093	0.095
		2	0.098	
		3	0.094	
Testing the program speed without color tracking	4/27/2016	1	0.037	0.035
		2	0.033	
		3	0.035	
Testing the program speed with an algorithm that minimize the load of the color tracking by using timers	4/29/2016	1	0.037	0.037
		2	0.036	
		3	0.039	

Shown Table 4.9 is the results test case S02. This shows the speed difference of the program in various tests scenario conducted. Program speed was a critical value for the performance of the system as this affect the calibration and motion tracking of the system. As shown on the results, the algorithm added to the program logic was successful in enhancing the speed for thee calibration and motion tracking process, therefore enhancing the overall performance.

Description of the Prototype

For the program to work the Xbox 360 Kinect must be connected in the Laptop or Desktop Computer with a Windows 8 operating system via the Xbox 360 AC adapter/power supply to power the said adapter is also used for powering on the Xbox 360 Kinect.

There are certain installers that new users must have on their computer for the whole program to work; one of it is the Kinect SDK 1.8 which includes the device driver of the Xbox360 Kinect sensor. The next are the open source installers:

- OpenCV 2.2 - It includes the computer vision library which is utilized for the color tracking process..
- OpenNi 2.2 - It includes the neutral interface library which is utilized to get the tracked information from the Kinect sensor.
- NiTE 2.2 – it includes the modules of OpenNI providing gesture and skeleton tracking functions.

Once everything is set and installed, the user should just run the program and open any simple drawing applications to test and see how the program works. As said earlier in the early chapters, there are two LED pens used and the program detects two users, the user will just press a button based on what the user wants to do, the LED pens has labels in it so that the users would not be confused of the light's functions and just point the light that emits in the pen to the Xbox 360 Kinect and it will do the function based on what the user pressed.

Screenshots

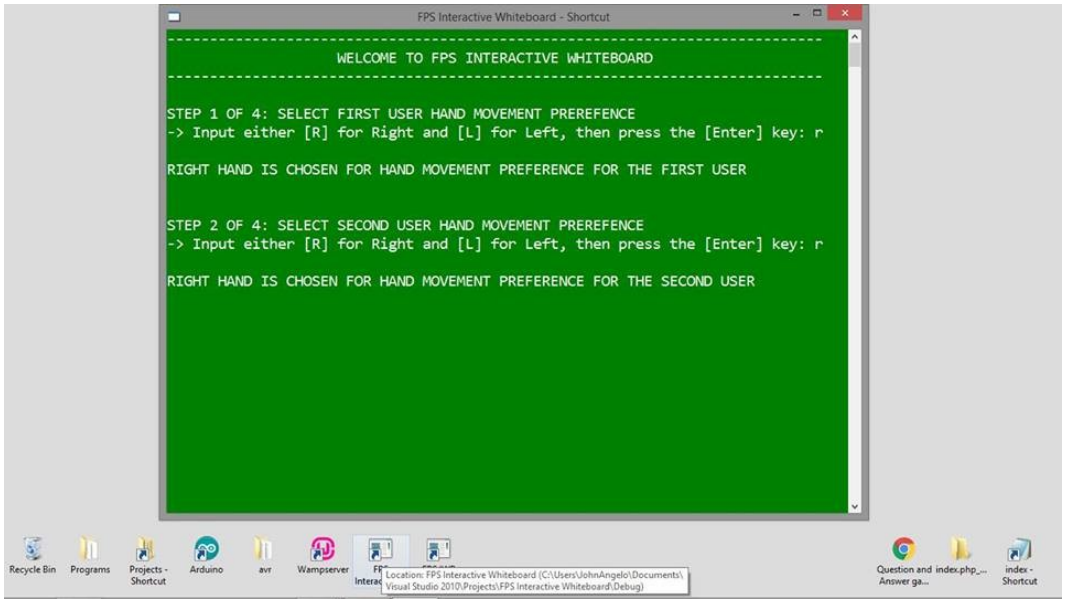

Figure 4.22 Program Start-up (Part 1)

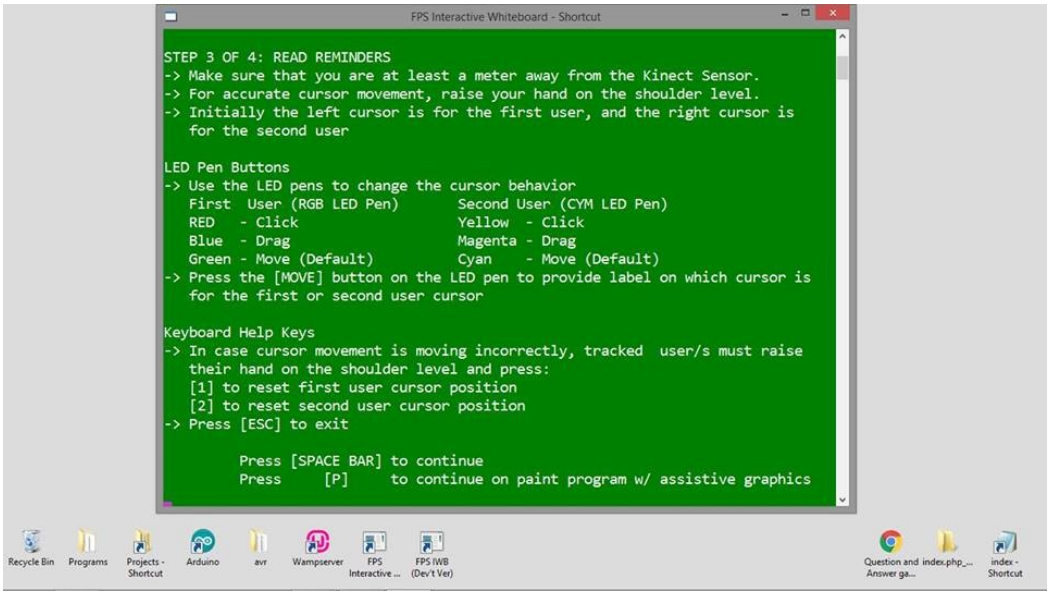

Figure 4.23 Program Start-up (part 2)

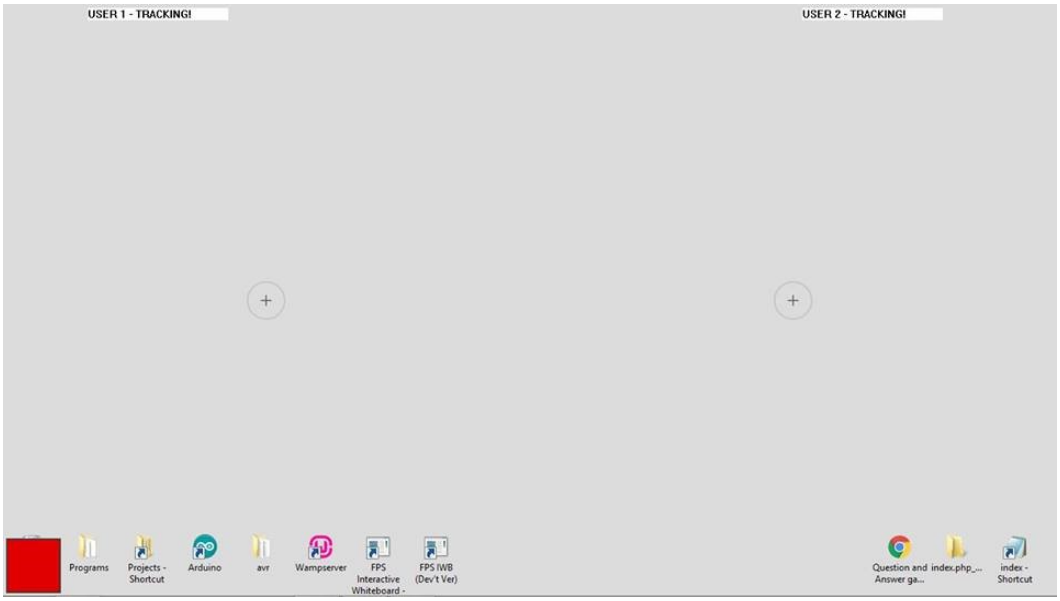

Figure 4.24 Program running: red color was detected / user 1 click-event cursor

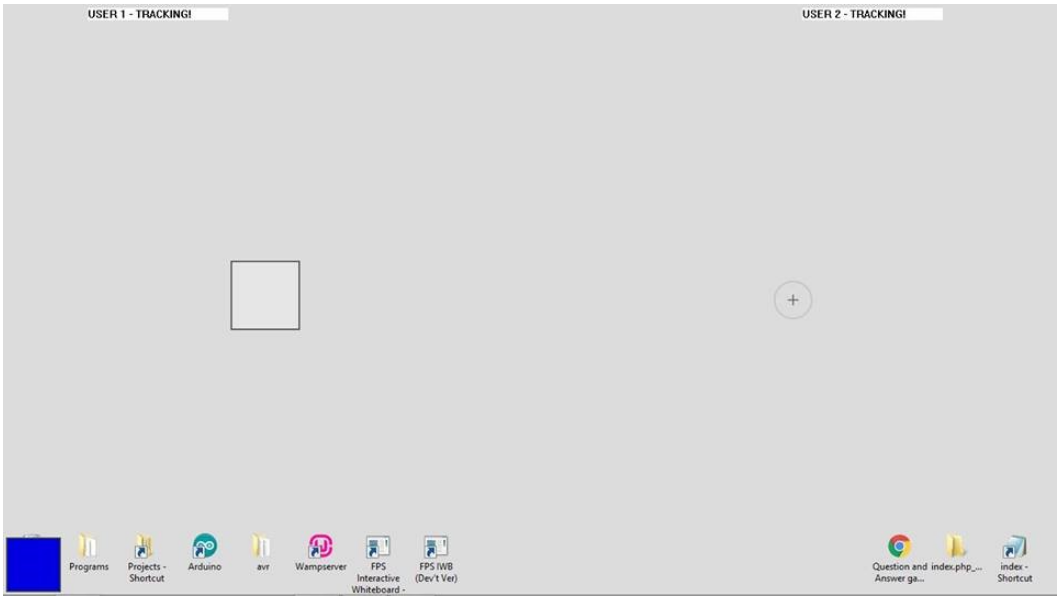

Figure 4.25 Program running: blue color was detected / user 1 drag-event cursor

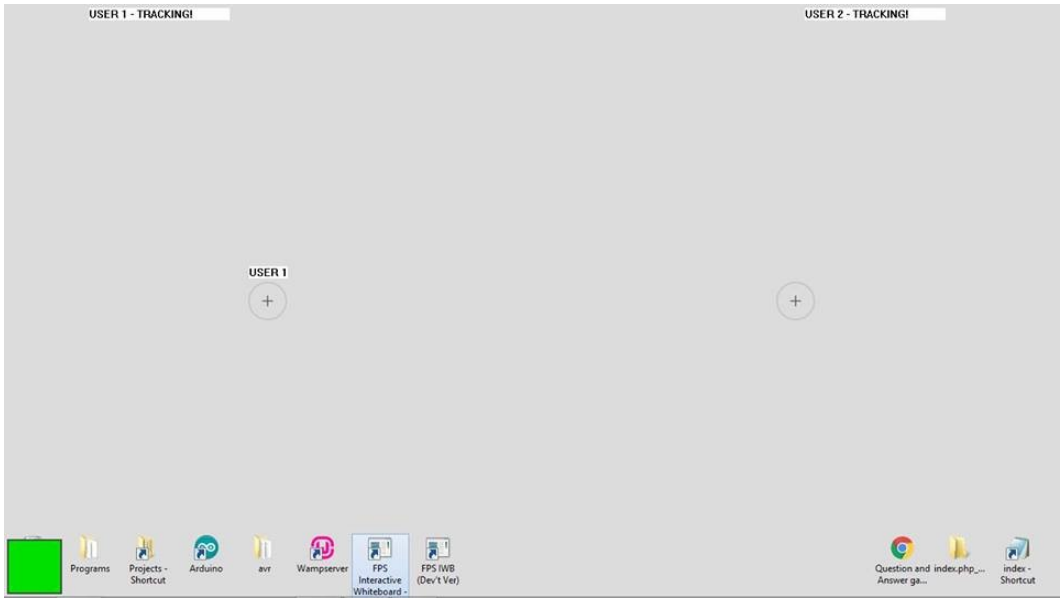

Figure 4.26 Program running: green color was detected / user 1 move-event cursor

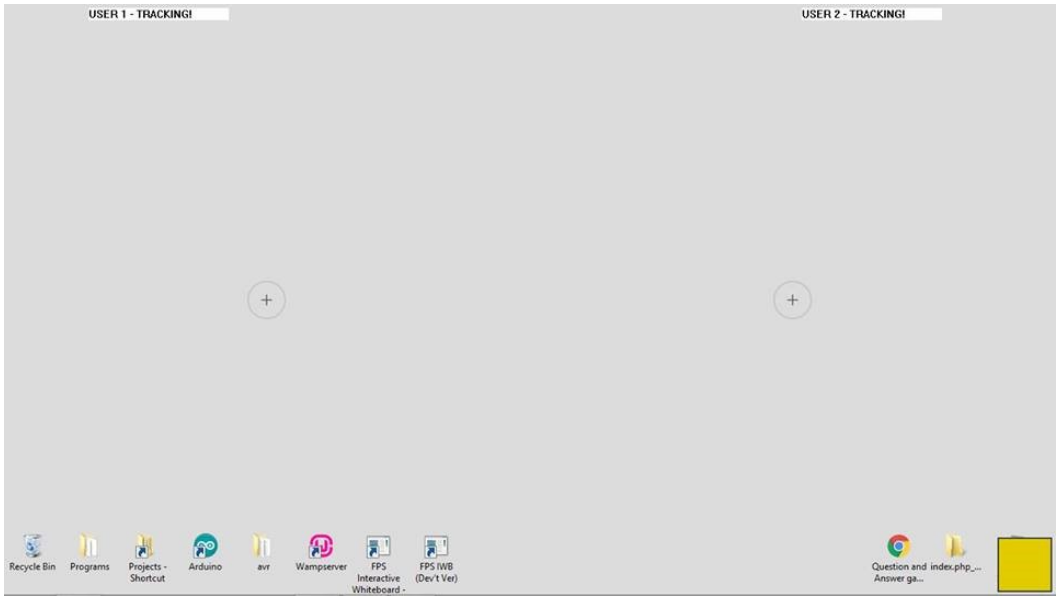

Figure 4.27 Program running: yellow color was detected / user 2 click-event cursor

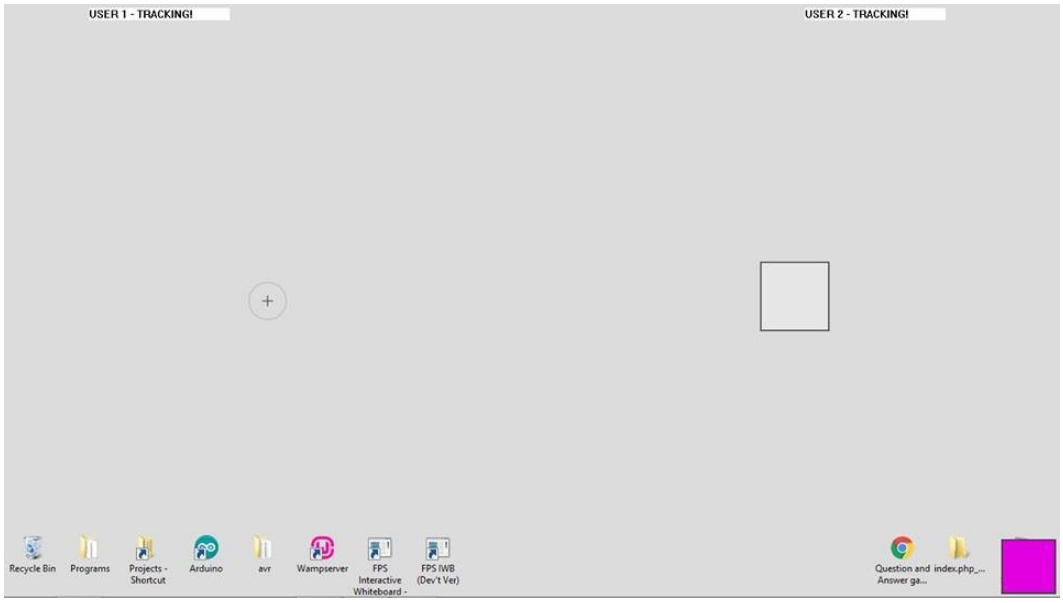

Figure 4.28 Program running: magenta color was detected / user 2 drag-event cursor

Figure 4.29 Program running: cyan color was detected / user 2 move-event cursor

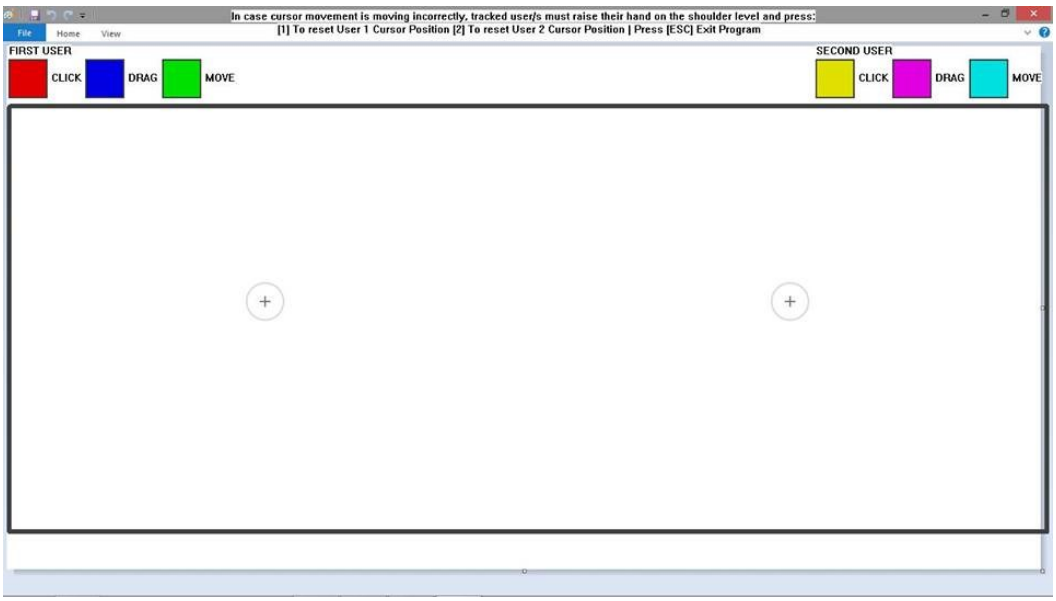

Figure 4.30 Program running: On demo paint program opened with assistive graphics

Pictures

Figure 4.31 User 1 LED Pen producing the red color

Figure 4.32 User 1 LED Pen producing the green color

Figure 4.33 User 1 LED Pen producing the blue color

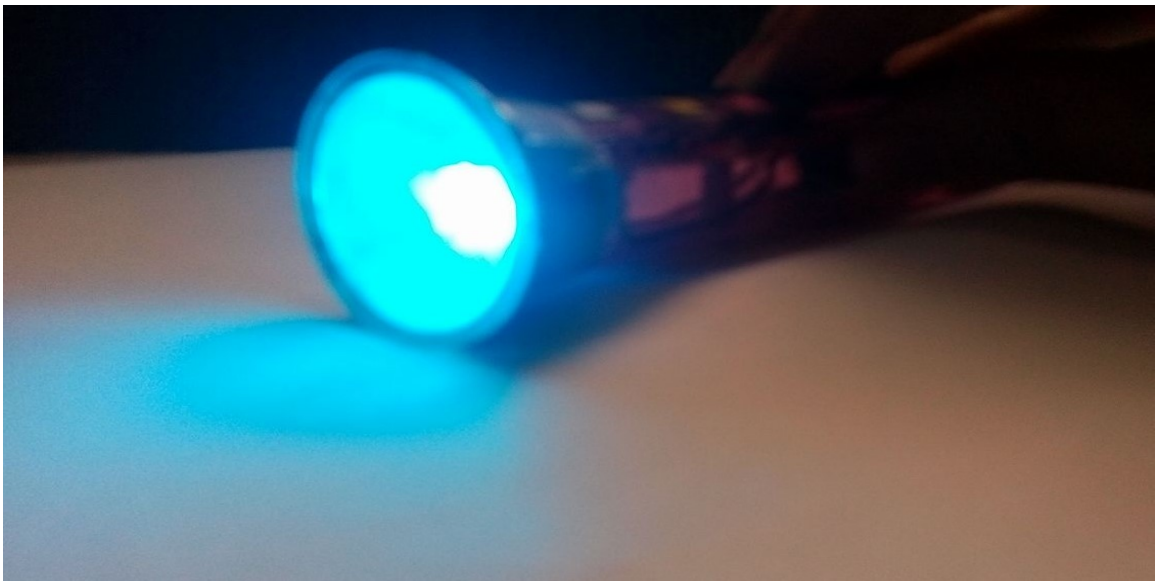

Figure 4.34 User 2 LED Pen producing the cyan color

Figure 4.35 User 2 LED Pen producing the color yellow

Figure 4.36 User 2 LED Pen producing the color magenta

CHAPTER 5

RECOMMENDATIONS

Through extensive testing and collaboration on the prototype, the proponents have gathered some recommendations that can further improve the capstone project:

- Provide a better color source that produces a hue similar to what can be seen on typical LED screens of a smart phone as it give more accuracy and ease for the color tracking.
- Improve the LED pen's durability by using materials such as fiber glass for the casing.
- Improve the LED pen's effectivity by providing a larger flashlight reflector that will fit the casing
- Make use of the Xbox one Kinect this would allow them to have access to better hardware and API to make the project as efficient as it can be.

Some recommendation not included in the prototype as it was not part of the scope but can be included in future iterations:

- The integration of gestures available in the NITE Library or Kinect API. This would allow more inputs for the users, maximizing the Kinects potential.
- The inclusion of voice commands or a text to speech component. This would allow the system to be completely hands off but would also require heavy system requirements on the machine running the system

BIBLIOGRAPHY

Akbas, O., & Pektas, H. M. (2012, December 27). *The effects of using an interactive whiteboard on the academic achievement of university students*. Retrieved February 26, 2014, from http://www.ied.edu.hk/apfslt/v12_issue2/akbas/

Avancini, M. (2012). *Using Kinect to an Interactive Whiteboard*. Retrieved from http://latemar.science.unitn.it/segue_userFiles/LITSA/Avancini_Mattia_138793_Thesis.pdf

Barton, M. (n.d.). *The History of Interactive Whiteboards*. eHow. Retrieved February 18, 2014, from http://www.ehow.com/facts_6976419_history-interactive-whiteboards.htm

Branzburg, J. (2008, April 22). *The Whiteboard Revolution.* Retrieved from www.techlearning.com/product-reviews/0072/the-whiteboard-revolution/45024

Competente, J., Eugenio, K., Fernandez, G., Rayos, H. (2014, October) *Interactive Whiteboard using Raspberry Pi.*

Dawson, C. (2010, July 21). *SMART vs. Promethean* [Web log post]. Retrieved from http://www.zdnet.com/blog/education/smart-vs-promethean/4049 Retrieved February 22, 2014, from http://www.interactivewhiteboards.com/

Humaidan, I. (2012). *The Effect of Using the Smart Board on Students' Achievement in Social Studies Curriculum*. Retrieved from http://www.cluteinstitute.com/proceedings/2012%20LV%20Papers/Article%20210.pdf

Marzano, R. (2009, November). *Educational Leadership: Multiple Measures: Teaching with Interactive Whiteboards*. Retrieved from http://www.ascd.org/publications/educational-leadership/nov09/vol67/num03/Teaching-with-Interactive-Whiteboards.aspx

McEntyre, M. (2006). *The Effects Interactive Whiteboards Have on Student Motivation*. Retrieved February 22, 2014, from http://mandymc.myweb.uga.edu/iwb%20synthesis.pdf

Lee, J. (2007). *Low-Cost Multi-Point Interactive Whiteboards Using the Wiimote.* Retrieved February 22, 2014, from http://johnnylee.net/projects/wii/

Lee, W. C., Merino-Schettino, D. A., López-Martínez, I., Posada Gómez, R., & Juárez-Martínez, U. (2012). *Virtual Board: A low cost Multi Touch Human Computer Interaction System.* Procedia Technology, 3, 178-186.

Preston, C., & Mowbray, M. (2008). *Use of SMART Boards for teaching, learning and assessment in kindergarten science*. Teaching Science, 54(2), 1-2. Retrieved from http://smartboardita.pbworks.com/f/smartboard+with+kindergartener.pdf

Ronchetti, M., & Avancini, M. (2011). *Using Kinect to emulate an Interactive Whiteboard.* MS in Computer Science, University of Trento.

Schneider, L. (n.d.). *C++ Programming Language.* Retrieved from http://jobsearchtech. 55 about.com/od/techcareersskills/p/CPPProgramming.htm

Singh, D., Omar, R., & Anuar, A. (2010). *Low Cost Interactive Electronic Whiteboard Using Nintendo Wii Remote*. American Journal of Applied Sciences, 7(11).

Soares, C., Torres, R., & Sobral, P. (2013). *LoCoBoard: Low-Cost Interactive Whiteboard Using Computer Vision Algorithms*. ISRN Machine Vision, 2013, 13. doi:10.1155/2013/252406

Stourstrup, B. (1979) C++ Programming. Retrieved from http://www.cplusplus.com/info/history/

Türel, Y. (2011). *An interactive whiteboard student survey: Development, validity and reliability.* Retrieved from http://www.cluteinstitute.com/proceedings/2012%20 LV%20Papers/Article%20210.pdf

About." (n.d.). OpenCV. Retrieved February 27, 2014, from http://opencv.org/about.html

"About SMART Boards." (n.d.). eHow. Retrieved March 3, 2014 from http://www.ehow.com/about_5154748_smart-boards.html

"Interactive Whiteboards Enhance Classroom Instruction and Learning" (n.d.). Retrieved

from February 18, 2014, from http://www.neamb.com/professionalresources/ benefits of- interactive-whiteboards.htm

"Kinect for Windows" (November 2010) Retrieved from https://www.microsoft.com/enus/kinectforwindows/develop/learn.aspx

*"Kinect for Windows Sensor Components and Specifications "(*2015) Retrieved from https://msdn.microsoft.com/en-us/library/jj131033.aspx

"Microsoft Kinect" (November 2010) Retrieved from http://en.wikipedia.org/wiki/Kinect

"Negros public schools to be given interactive whiteboards, other educational tech." (2013, December 14). Retrieved from http://www.gmanetwork.com/news/story/ 56 339795/scitech/technology/negros-public-schools-to-be-given-interactivewhiteboardsother-educational-tech

"OpenNI" (n.d.) Retrieved November 2010, from http://en.wikipedia.org/wiki/OpenNi

"SMART Interactive Solutions for Education, Business and Government -SMART Technologies." (n.d.). Retrieved February 22, 2014, from http://smarttech.com/

"Welcome to eInstruction - Simple Solutions. Real Results." (n.d.). eInstruction. Retrieved from http://www.einstruction.com/

"SMART's commitment to education - SMART Technologies." (n.d.). Retrieved February 19, 2014, from https://smarttech.com/About+SMART/About+SMART/ Commitment/Commitment+ To+Education

"The C++ Programming Language. (n.d.)." Retrieved February 27, 2014, from http:// groups.engin.umd.umich.edu/CIS/course.des/cis400/cpp/cpp.html

"*The C Programming Language*." (n.d.). Retrieved from http://www.cprogramming.com/begin.html

"Biography of Dennis Ritchie." (n.d.) Retrieved from http://www.computerhistory.org/fellowawards/hall/bios/Dennis,Ritchie/

"Simulating Touch Input in Windows 8 Using Touch Injection API" (n.d.) Retrieved from

http://social.technet.microsoft.com/wiki/contents/articles/6460.simulating-touch-input-in-windows-8-using-touch-injection-api.aspx

"WM_Paint Message" (n.d.) Retrieved from https://msdn.microsoft.com/en-us/library/windows/desktop/dd145213(v=vs.85).aspx

"Multi-touch Vista" (n.d.) Retrieved from https://multitouchvista.codeplex.com/

"How to draw on the windows desktop using the GDI API?" (n.d.) Retrieved from http://stackoverflow.com/questions/13492860/how-to-draw-on-the-windows-desktop-using-the-gdi-api

APPENDIX A

RELEVANT SOURCE CODE

Main Kinect Program
main.cpp

```
//---------------------------------MAIN---------
-------------------------//
int WINAPI WinMain
(
	_In_ HINSTANCE hInstance,
	_In_opt_ HINSTANCE
hPrevInstance,
	_In_ LPSTR lpCmdLine,
	_In_ int nShowCmd
)
{
#pragma region Initialization
	// OpenCV Video Stream
	cv::Mat mImageBGR;
	cv::Mat mImageHSV;
	//  Desktop Resolution
	RECT desktopResolution;
	GetWindowRect(hDesktop,
&desktopResolution);

	// Cursor Speed
	int cursorSpeedDivisor = 3;
	// Confidence Level
	float nConfidence = .5;
	// Resolution Ratio
	double rRatio = 0.17;

	// DISPLAY USER TEXT
	double time_counter0 = 0;
  clock_t this_time0 = clock();
  clock_t last_time0 = this_time0;
	double time_counter1 = 0;
  clock_t this_time1 = clock();
  clock_t last_time1 = this_time1;
	bool displayUserNTextLock0 =
false;
	bool displayUserNTextLock1 =
false;

	// user profile variable (0 = user1,
1 = user2)
	int n[2] = {0,1};
	// Cursor Coordinates Reference
	float startPosX[2] = { -1,-1 };
	float startPosY[2] = { -1,-1 };
	// color track counter
```

```
        bool wasDrag1Tracked = false;
        bool wasDrag2Tracked = false;
        // Stream Capturing Perspective
        bool bMirror = false;
        // Dragged Cursor Status
        bool isDragged[2] = {false,false};
        // User slot status
        bool isUserSlot1Occupied =
false;
        bool isUserSlot2Occupied =
false;
        // Cursor Coordinates
        int curX[2] =
{desktopResolution.right * 1/4,
desktopResolution.right * 3/4};
        int curY[2] =
{desktopResolution.bottom/2,
desktopResolution.bottom/2};
        // Cursor Limitations
        int cursorLimitLeft =
desktopResolution.left;
        int cursorLimitRight =
desktopResolution.right - 5;
        int cursorLimitTop =
desktopResolution.top;
        int cursorLimitBottom =
desktopResolution.bottom - 5;

        // COLOR INDICATORS
        int     indicatorBoxSize = 70;
        int indicatorGap = 5;
        int indicatorTotalSize =
indicatorBoxSize + indicatorGap;
        // INSTRUCTIONS
        int instructionGap = 10;
        int instructionLabel = 20;
        int instructionBorderSize = 2;
        int instructionBoxSize = 50;
        int instructionPosition = (int)
desktopResolution.bottom * rRatio;
        // BORDER
        int borderTopMax = (int)
desktopResolution.bottom * rRatio;
        int borderBottomMax =
desktopResolution.bottom -
indicatorTotalSize - 10;
        int borderGap = indicatorGap +
3;

#pragma region Auto Adjust Resolution
```

```
        if(desktopResolution.right ==
1920)
                rRatio = 0.14;
        else if(desktopResolution.right
== 1366)
                rRatio = 0.17;
        else
                rRatio = 0.17;

        borderTopMax = (int)
desktopResolution.bottom * rRatio;
        instructionPosition = (int)
desktopResolution.bottom * rRatio;
#pragma endregion

#pragma region Pop Text Coordinates
        yPopText = instructionPosition -
instructionBoxSize - instructionGap -
instructionLabel;
        xPopText1 = borderGap + 80;
        xPopText2 =
desktopResolution.right - borderGap -
300 + 99;
#pragma endregion

#pragma region Graphics
        HDC dcRed = GetDC(NULL);
        HDC dcGreen = GetDC(NULL);
        HDC dcBlue = GetDC(NULL);
        HDC dcCyan = GetDC(NULL);
        HDC dcYellow = GetDC(NULL);
        HDC dcMagenta =
GetDC(NULL);
        HDC dcText = GetDC(NULL);

        HBRUSH hRedBrush =
CreateSolidBrush(RGB(224,0,0));
        HBRUSH hGreenBrush =
CreateSolidBrush(RGB(0,224,0));
        HBRUSH hBlueBrush =
CreateSolidBrush(RGB(0,0,224));
        HBRUSH hCyanBrush =
CreateSolidBrush(RGB(0,224,224));
        HBRUSH hYellowBrush =
CreateSolidBrush(RGB(224,202,0));
        HBRUSH hMagentaBrush =
CreateSolidBrush(RGB(224,0,224));

        HPEN hGrayPen =
CreatePen(PS_SOLID,
instructionBorderSize, RGB(64, 64, 64));
```

```
        // Instruction Graphics
        SelectObject(dcRed,hRedBrush)
;
        SelectObject(dcGreen,hGreenBr
ush);
        SelectObject(dcBlue,hBlueBrush
);
        SelectObject(dcCyan,hCyanBrus
h);
        SelectObject(dcYellow,hYellowB
rush);
        SelectObject(dcMagenta,hMage
ntaBrush);

        SelectObject(dcRed,hGrayPen);
        SelectObject(dcGreen,hGrayPen
);
        SelectObject(dcBlue,hGrayPen);
        SelectObject(dcCyan,hGrayPen)
;
        SelectObject(dcYellow,hGrayPe
n);
        SelectObject(dcMagenta,hGrayP
en);
#pragma endregion

#pragma region Touch Injector
        InitializeTouchInjection(2,
TOUCH_FEEDBACK_INDIRECT);

        for (int i = 0; i < 2; i++)
        {

        POINTER_TOUCH_INFO
&contact = contact_[i];
                memset(&contact, 0,
sizeof(POINTER_TOUCH_INFO));

        contact.pointerInfo.pointerType =
PT_TOUCH;

        contact.pointerInfo.pointerId =
i;

        contact.pointerInfo.ptPixelLocati
on.y = 0;

        contact.pointerInfo.ptPixelLocati
on.x = 0;

                contact.touchFlags =
TOUCH_FLAG_NONE;
```

```
                contact.touchMask =
TOUCH_MASK_CONTACTAREA |
TOUCH_MASK_ORIENTATION |
TOUCH_MASK_PRESSURE;
                contact.orientation = 90;
                contact.pressure =
32000;

                // defining contact area (I
have taken area of 4 x 4 pixel)
                contact.rcContact.top =
contact.pointerInfo.ptPixelLocation.y - 2;
                contact.rcContact.bottom
= contact.pointerInfo.ptPixelLocation.y +
2;
                contact.rcContact.left =
contact.pointerInfo.ptPixelLocation.x - 2;
                contact.rcContact.right =
contact.pointerInfo.ptPixelLocation.x +
2;

        contact.pointerInfo.pointerFlags
= POINTER_FLAG_UPDATE |
POINTER_FLAG_INRANGE;//|
POINTER_FLAG_INCONTACT;
        }
#pragma endregion

#pragma region OpenNI Part
        // Initial OpenNI
        if (OpenNI::initialize() !=
openni::STATUS_OK)
        {
                cerr << "OpenNI Initial
Error: " << OpenNI::getExtendedError()
<< endl;

        MessageBox(NULL,(LPCWSTR)
L"Please connect/reconnect Kinect
device!",(LPCWSTR)L"FPS Interactive
Whiteboard",MB_ICONERROR |
MB_OK | MB_DEFBUTTON1);
                return -1;
        }
        // Open Device
        Device devDevice;
        if
(devDevice.open(ANY_DEVICE) !=
openni::STATUS_OK)
        {
```

```
                cerr << "Can't Open
Device: " <<
OpenNI::getExtendedError() << endl;

        MessageBox(NULL,(LPCWSTR)
L"Please connect/reconnect Kinect
device!",(LPCWSTR)L"FPS Interactive
Whiteboard",MB_ICONERROR |
MB_OK | MB_DEFBUTTON1);
                return -1;
        }
        // create depth stream
        VideoStream vsDepthStream;
        if
(vsDepthStream.create(devDevice,
SENSOR_DEPTH) ==
openni::STATUS_OK)
        {
                // set video mode
                VideoMode mMode;

        mMode.setResolution(640,480);
                mMode.setFps(30);

        mMode.setPixelFormat(PIXEL_F
ORMAT_DEPTH_1_MM);
                if
(vsDepthStream.setVideoMode(mMode)
!= openni::STATUS_OK)
                {
                        cout << "Can't
apply VideoMode: " <<
OpenNI::getExtendedError() << endl;
                }

        vsDepthStream.setMirroringEna
bled(bMirror);
        }

        // Create color stream
        VideoStream vsColorStream;
        if
(vsColorStream.create(devDevice,
SENSOR_COLOR) ==
openni::STATUS_OK)
        {
                // set video mode
                VideoMode mMode;

        mMode.setResolution(FRAME_
WIDTH, FRAME_HEIGHT);
                mMode.setFps(30);
```

```
        mMode.setPixelFormat(PIXEL_F
ORMAT_RGB888);

                if
(vsColorStream.setVideoMode(mMode)
!= openni::STATUS_OK)
                {
                        cout << "Can't
apply VideoMode: " <<
OpenNI::getExtendedError() << endl;
                }

                // image registration
                if
(devDevice.isImageRegistrationModeSu
pported(IMAGE_REGISTRATION_DEP
TH_TO_COLOR))
                {

        devDevice.setImageRegistration
Mode(IMAGE_REGISTRATION_DEPT
H_TO_COLOR);
                }

        vsColorStream.setMirroringEnabl
ed(bMirror);
        }
        else
        {
                cerr << "Can't create
color stream on device: " <<
OpenNI::getExtendedError() << endl;

        MessageBox(NULL,(LPCWSTR)
L"Please connect/reconnect Kinect
device!",(LPCWSTR)L"FPS Interactive
Whiteboard",MB_ICONERROR |
MB_OK | MB_DEFBUTTON1);
                return -1;
        }
#pragma endregion

#pragma region NiTE Part
        // Initial NiTE
        if (NiTE::initialize() !=
nite::STATUS_OK)
        {
                cerr << "NiTE initial error"
<< endl;
```

```
        MessageBox(NULL,(LPCWSTR)
L"Please connect/reconnect Kinect
device!",(LPCWSTR)L"FPS Interactive
Whiteboard",MB_ICONERROR |
MB_OK | MB_DEFBUTTON1);
                return -1;
        }
        // create user tracker
        UserTracker mUserTracker;
        if
(mUserTracker.create(&devDevice) !=
nite::STATUS_OK)
        {
                cerr << "Can't create user
tracker" << endl;

        MessageBox(NULL,(LPCWSTR)
L"Please connect/reconnect Kinect
device!",(LPCWSTR)L"FPS Interactive
Whiteboard",MB_ICONERROR |
MB_OK | MB_DEFBUTTON1);
                return -1;
        }
#pragma endregion
#pragma endregion
        //---------------------------------
START---------------------------------//
        RedirectIOToConsole();

        vsColorStream.start();
        vsDepthStream.start();

        promptMenu();

        if(conToPaint)
        {
                cursorLimitLeft =
borderGap + 30;
                cursorLimitRight =
desktopResolution.right - borderGap -
30;
                cursorLimitTop
=  borderTopMax + 30;
                cursorLimitBottom =
borderBottomMax - 30;
                system("start
mspaint.exe");
```

```
                system("start
C:\\FPSIWBGraphics3\\Debug\\FPSIWB
Graphics3.exe");
                //system("start
C:\\Users\\Angelo\\Desktop\\FPSIWBGr
aphicsVer3\\Debug\\FPSIWBGraphicsV
er3.exe");
        }
        else if(!conToPaint)
        {
                cursorLimitLeft = 10;
                cursorLimitRight =
desktopResolution.right - 10;
                cursorLimitTop =  10;
                cursorLimitBottom =
borderBottomMax - 10;

                desktopMaxWidth =
desktopResolution.right;

                xPopText1 = 160;
                xPopText2 =
desktopResolution.right - 280;
                yPopText = 5;
        }
        nite::Status status =
nite::STATUS_OK;
        //---------------------------------
LOOP START---------------------------------
//
        while (true)
        {

        mouseMotion(contact_[0],curX[0]
,curY[0],isDragged[0]);

        mouseMotion(contact_[1],curX[1]
,curY[1],isDragged[1]);
                InjectTouchInput(2,
contact_);

                #pragma region Display
User N Text Clock Timer

        if(displayUserNTextLock0)
                {
                        this_time0 =
clock();
                        time_counter0 +=
(double)(this_time0 - last_time0);
```

```
                        last_time0 =
this_time0;

                        if(time_counter0 >
(double)(NUM_SECONDS2 *
CLOCKS_PER_SEC))
                        {

        time_counter0 = 0;

        displayUserNTextLock0 = false;
                        }
                }

        if(displayUserNTextLock1)
                {
                        this_time1 =
clock();
                        time_counter1 +=
(double)(this_time1 - last_time1);
                        last_time1 =
this_time1;

                        if(time_counter1 >
(double)(NUM_SECONDS2 *
CLOCKS_PER_SEC))
                        {

        time_counter1 = 0;

        displayUserNTextLock1 = false;
                        }
                }
                #pragma endregion

                // get user frame
                UserTrackerFrameRef
        mUserFrame;
                if
(mUserTracker.readFrame(&mUserFra
me) == nite::STATUS_OK)
                {
                        // get color frame
                        VideoFrameRef
vfColorFrame;
                        if
(vsColorStream.readFrame(&vfColorFra
me) == openni::STATUS_OK)
                        {
                                // convert
data to OpenCV format
```

```
                                const
cv::Mat
mImageRGB(vfColorFrame.getHeight(),
vfColorFrame.getWidth(), CV_8UC3,
const_cast<void*>(vfColorFrame.getDat
a()));
                                // convert
form RGB to BGR

        cv::cvtColor(mImageRGB,
mImageBGR, CV_RGB2BGR);
                                //Convert
the captured frame from BGR to HSV

        cvtColor(mImageBGR,
mImageHSV, COLOR_BGR2HSV);

                                //----------
----------COLOR TRACKING-----
----------//
                                #pragma
region Color Tracking Part
                                // First
User (RGB)
```

```
        if(trackFilteredObject(cRed,
mImageHSV))
                                {

        Rectangle(dcRed,indicatorGap,d
esktopResolution.bottom - indicatorGap,
indicatorTotalSize,desktopResolution.bo
ttom -  indicatorTotalSize);


        if(wasDrag1Tracked)
                                {

        mouseReleased(contact_[0],
curX[0], curY[0]);

        InjectTouchInput(2, contact_);

        Sleep(20);
                                }

        mousePressed(contact_[0],
curX[0], curY[0]);

        InjectTouchInput(2, contact_);
```

```
        Sleep(20);

        mouseReleased(contact_[0],
curX[0], curY[0]);

        InjectTouchInput(2, contact_);

        Sleep(20);



        isDragged[0] = false;

        wasDragg1Tracked = false;

                                }
                                else
if(trackFilteredObject(cBlue,
mImageHSV))
                                {

        Rectangle(dcBlue,indicatorGap,d
esktopResolution.bottom - indicatorGap,
indicatorTotalSize,desktopResolution.bo
ttom -  indicatorTotalSize);

        if(!wasDrag1Tracked)
                                                {

        mousePressed(contact_[0],
curX[0], curY[0]);

        InjectTouchInput(2, contact_);

        Sleep(20);



        isDragged[0] = true;

        wasDrag1Tracked = true;
                                                }
                                }
                                else
if(trackFilteredObject(cGreen,
mImageHSV))
                                {

        Rectangle(dcGreen,indicatorGap
,desktopResolution.bottom -
indicatorGap,
```

```
indicatorTotalSize,desktopResolution.bo
ttom -  indicatorTotalSize);

        if(!displayUserNTextLock0)
                                        {

        displayUserNText(curX[0],curY[0
],0,cursorLimitLeft,cursorLimitTop);

        displayUserNTextLock0 = true;
                                        }

        if(wasDrag1Tracked)
                                        {

        mouseReleased(contact_[0],
curX[0], curY[0]);

        InjectTouchInput(2, contact_);

        Sleep(20);
                                        }

        isDragged[0] = false;

        wasDrag1Tracked = false;
                                        }

                                        // Second
User (CYM)

        if(trackFilteredObject(cYellow,
mImageHSV))
                                        {

        Rectangle(dcYellow,desktopRes
olution.right -
indicatorGap,desktopResolution.bottom
- indicatorGap ,desktopResolution.right -
  indicatorTotalSize,desktopResolution.b
ottom -  indicatorTotalSize);

        if(wasDrag2Tracked)
                                                {

        mouseReleased(contact_[1],
curX[1], curY[1]);

        InjectTouchInput(2, contact_);

        Sleep(20);
                                                }
```

```
        mousePressed(contact_[1],
curX[1], curY[1]);

        InjectTouchInput(2, contact_);

        Sleep(20);

        mouseReleased(contact_[1],
curX[1], curY[1]);

        InjectTouchInput(2, contact_);

        Sleep(20);


        wasDrag2Tracked = false;

        isDragged[1] = false;
                                }
                                else
if(trackFilteredObject(cMagenta,
mImageHSV))
                                {

        Rectangle(dcMagenta,desktopR
esolution.right -
indicatorGap,desktopResolution.bottom
- indicatorGap ,desktopResolution.right -
  indicatorTotalSize,desktopResolution.b
ottom -  indicatorTotalSize);

                if(!wasDrag2Tracked)
                                                {

                mousePressed(contact_[1],
curX[1], curY[1]);

                InjectTouchInput(2, contact_);

                Sleep(20);


                isDragged[1] = true;

                wasDrag2Tracked = true;
                                                }
                                        }
                                        else
if(trackFilteredObject(cCyan,
mImageHSV))
                                        {
```

```
        Rectangle(dcCyan,desktopResol
ution.right -
indicatorGap,desktopResolution.bottom
- indicatorGap ,desktopResolution.right -
  indicatorTotalSize,desktopResolution.b
ottom -  indicatorTotalSize);

        if(!displayUserNTextLock1)
                                        {

        displayUserNText(curX[1],curY[1
],1,cursorLimitLeft,cursorLimitTop);

        displayUserNTextLock1 = true;
                                        }

        if(wasDrag2Tracked)
                                        {

        mouseReleased(contact_[1],
curX[1], curY[1]);

        InjectTouchInput(2, contact_);
                Sleep(20);
                                                }

                isDragged[1] = false;

                wasDrag2Tracked = false;
                                        }

                                        #pragma
        endregion

                vfColorFrame.release();
                                }
                                else
                                {
                                        cerr <<
        "Can't get color frame" << endl;
                                }

                mouseMotion(contact_[0],curX[0]
        ,curY[0],isDragged[0]);
```

```
        mouseMotion(contact_[1],curX[1]
,curY[1],isDragged[1]);
                        //----------------------
----------TRACK JOINTS and
CURSOR----------------------------------//
                        #pragma region
Track Joints and Cursor Part

                        const
nite::Array<nite::UserData>& mUsers =
mUserFrame.getUsers();

                        for (int i = 0; i <
mUsers.getSize() && i < MAX_USERS;
++i)
                        {
                                const
nite::UserData& user = mUsers[i];
                                const
nite::SkeletonJoint& nHandR =
user.getSkeleton().getJoint(nite::JOINT_
RIGHT_HAND);
                                const
nite::SkeletonJoint& nHandL =
user.getSkeleton().getJoint(nite::JOINT_
LEFT_HAND);

                                uIDDisplay
= n[i] + 1;

                        updateUserState(user);

                                if
(user.isNew())
                                {

                        mUserTracker.startSkeletonTrac
king(user.getId());

                        //if(uIDDisplay == 2)

                        //printf("\t\t\t\t");

                        //printf("----- REGISTER USER
%d -----\n",n[i] + 1);

                        mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);
```

```
        #pragma region retain user
profile algo - ADD

        if(profileAlgo)
                                        {

        if(!isUserSlot2Occupied)
                // check if slot 2 is not
occupied / else is full

        {

                if(!isUserSlot1Occupied)
                // check if slot 1 is not
occupied

                {

                        n[0] = 0;
                                // slot 1 will
accomodate the user 1


        isUserSlot1Occupied = true;
                                                }

                                                else

                                                {

                                                        if(n[0] == 0)
                                                        // slot 1 is
occupied for user 1 / slot 2 will will retain
user 2 profile

                                                                n[1] = 1;

                                                        else
                                                                // slot 1 is
occupied for user 2 / slot 2 will will retain
user 1 profile

                                                                n[1] = 0;
```

```
                                        (nHandR.getPositionConfidence() >
                                        nConfidence)
        isUserSlot2Occupied = true;
                                                {
                }
                                                    if (!(startPosX[n[i]] >= 0
        }                                       && startPosY[n[i]] >= 0))
                                    }
                                                    {

        #pragma endregion
                                                        mUserTracker.convertJointCoord
                                    }               inatesToDepth(
                                    else if
(user.getSkeleton().getState() ==
nite::SKELETON_TRACKED)                             nHandR.getPosition().x,
                                    {

        #pragma region For User 1/Right             nHandR.getPosition().y,
Hand

        if(n[i] == 0 && rHand1)                     nHandR.getPosition().z,
                                    {

        if                                          &startPosX[n[i]],
                                                    &startPosY[n[i]]);
```

```
                                        mUserTracker.convertJointCoord
inatesToDepth(
                        //printf("Starting
Cursor X: %4f | Cursor Y: %4f\n",
startPosX[n[i]], startPosY[n[i]]);                      nHandR.getPosition().x,



                                                        nHandR.getPosition().y,
        mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);
                                                        nHandR.getPosition().z,
                }

                else if (startPosX[n[i]] >=             &posX[n[i]], &posY[n[i]]);
0 && startPosY[n[i]] >= 0)

                {                                               if
                                                        (abs(int(posX[n[i]] - startPosX[n[i]])) >
                        float                           10)
posX[MAX_USERS],
posY[MAX_USERS];                                                curX[n[i]]
                                                        += int(((posX[n[i]] - startPosX[n[i]]) - 10)
                                                        / cursorSpeedDivisor);

                                                                if
```

```
(abs(int(posY[n[i]] - startPosY[n[i]])) >
10)
                                                    mouseMotion(contact_[n[i]],curX[
                        curY[n[i]]          n[i]],curY[n[i]],isDragged[n[i]]);
+= int(((posY[n[i]] - startPosY[n[i]]) - 10)
/ cursorSpeedDivisor);                              }

                    curX[n[i]] =                    else
min(curX[n[i]], cursorLimitRight);

                    curX[n[i]] =                    mouseMotion(contact_[n[i]],curX[
max(curX[n[i]], cursorLimitLeft);                   n[i]],curY[n[i]],isDragged[n[i]]);

                    curY[n[i]] =                    }
min(curY[n[i]], cursorLimitBottom);
                                                    else
                    curY[n[i]] =
max(curY[n[i]], cursorLimitTop);
                                                    mouseMotion(contact_[n[i]],curX[
                                                    n[i]],curY[n[i]],isDragged[n[i]]);
                    //printf("User %d |                                         }
Cursor X: %4d | Cursor Y: %4d\n",
user.getId(), curX[n[i]], curY[n[i]]);              #pragma endregion

                                                    #pragma region For User 1/Left
                                                    Hand
```

```
        else if(n[i] == 0 && !rHand1)
                                        {

        if
(nHandL.getPositionConfidence() >
nConfidence)

        {

                if (!(startPosX[n[i]] >= 0
&& startPosY[n[i]] >= 0))

                {


        mUserTracker.convertJointCoord
inatesToDepth(

        nHandL.getPosition().x,

        nHandL.getPosition().y,
                nHandL.getPosition().z,

                &startPosX[n[i]],
&startPosY[n[i]]);

                        //printf("Starting
Cursor X: %4f | Cursor Y: %4f\n",
startPosX[n[i]], startPosY[n[i]]);

                mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);

                }

                else if (startPosX[n[i]] >=
0 && startPosY[n[i]] >= 0)

                {

                        float
```

```
posX[MAX_USERS],
posY[MAX_USERS];

        mUserTracker.convertJointCoord
inatesToDepth(
        nHandL.getPosition().x,
        nHandL.getPosition().y,
        nHandL.getPosition().z,
        &posX[n[i]], &posY[n[i]]);

                if
(abs(int(posX[n[i]] - startPosX[n[i]])) >
10)
                                curX[n[i]]
+= int(((posX[n[i]] - startPosX[n[i]]) - 10)
/ cursorSpeedDivisor);

                                if
(abs(int(posY[n[i]] - startPosY[n[i]])) >
10)
                                curY[n[i]]
+= int(((posY[n[i]] - startPosY[n[i]]) - 10)
/ cursorSpeedDivisor);

                                curX[n[i]] =
min(curX[n[i]], cursorLimitRight);

                                curX[n[i]] =
max(curX[n[i]], cursorLimitLeft);

                                curY[n[i]] =
min(curY[n[i]], cursorLimitBottom);

                                curY[n[i]] =
max(curY[n[i]], cursorLimitTop);
```

```
                                                                }
                        //printf("User %d |
Cursor X: %4d | Cursor Y: %4d\n",
user.getId(), curX[n[i]], curY[n[i]]);

        mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);
                }
                else

        mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);
        }
        else

        mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);
        #pragma endregion

        #pragma region For User 2/Right
Hand

        else if(n[i] == 1 && rHand2)
                                                                {
        if
(nHandR.getPositionConfidence() >
nConfidence)
        {
                if (!(startPosX[n[i]] >= 0
&& startPosY[i] >= 0))
                {

        mUserTracker.convertJointCoord
inatesToDepth(
```

```
                                                    else if (startPosX[n[i]] >=
        nHandR.getPosition().x,                     0 && startPosY[n[i]] >= 0)

                                                    {
        nHandR.getPosition().y,
                                                            float
                                                    posX[MAX_USERS],
        nHandR.getPosition().z,                     posY[MAX_USERS];


        &startPosX[n[i]],
&startPosY[n[i]]);
                                                            mUserTracker.convertJointCoord
                                                    inatesToDepth(
                        //printf("Starting
Cursor X: %4f | Cursor Y: %4f\n",
startPosX[n[i]], startPosY[n[i]]);                          nHandR.getPosition().x,


                                                            nHandR.getPosition().y,
        mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);
                                                            nHandR.getPosition().z,
                }
```

```
        &posX[n[i]], &posY[n[i]]);

                        if
(abs(int(posX[n[i]] - startPosX[n[i]])) >
10)
                            curX[n[i]]
+= int(((posX[n[i]] - startPosX[n[i]]) - 10)
/ cursorSpeedDivisor);

                        if
(abs(int(posY[n[i]] - startPosY[n[i]])) >
10)
                            curY[n[i]]
+= int(((posY[n[i]] - startPosY[n[i]]) - 10)
/ cursorSpeedDivisor);

                        curX[n[i]] =
min(curX[n[i]], cursorLimitRight);

                        curX[n[i]] =
max(curX[n[i]], cursorLimitLeft);

                                curY[n[i]] =
min(curY[n[i]], cursorLimitBottom);

                                curY[n[i]] =
max(curY[n[i]], cursorLimitTop);

                                //printf("User %d |
Cursor X: %4d | Cursor Y: %4d\n",
user.getId(), curX[n[i]], curY[n[i]]);

                    mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);

                }

                else

                    mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);
```

```
        }

        else

        mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);
                                    }

        #pragma endregion

        #pragma region For User 2/Right
Hand

        else if(n[i] == 1 && !rHand2)
                                    {

        if
(nHandL.getPositionConfidence() >
nConfidence)

        {

                if (!(startPosX[n[i]] >= 0
&& startPosY[n[i]] >= 0))
                                        {

                        mUserTracker.convertJointCoord
inatesToDepth(
                        nHandL.getPosition().x,
                        nHandL.getPosition().y,
                        nHandL.getPosition().z,
                        &startPosX[n[i]],
&startPosY[n[i]]);

                                //printf("Starting
Cursor X: %4f | Cursor Y: %4f\n",
startPosX[n[i]], startPosY[n[i]]);
```

```
        mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);

                }

                else if (startPosX[n[i]] >=
0 && startPosY[n[i]] >= 0)

                {

                        float
posX[MAX_USERS],
posY[MAX_USERS];


        mUserTracker.convertJointCoord
inatesToDepth(

        nHandL.getPosition().x,
                nHandL.getPosition().y,

                nHandL.getPosition().z,

                &posX[n[i]], &posY[n[i]]);

                        if
(abs(int(posX[n[i]] - startPosX[n[i]])) >
10)

                                curX[n[i]]
+= int(((posX[n[i]] - startPosX[n[i]]) - 10)
/ cursorSpeedDivisor);

                        if
(abs(int(posY[n[i]] - startPosY[n[i]])) >
10)

                                curY[n[i]]
+= int(((posY[n[i]] - startPosY[n[i]]) - 10)
/ cursorSpeedDivisor);
```

```
                curX[n[i]] =
min(curX[n[i]], cursorLimitRight);

                curX[n[i]] =
max(curX[n[i]], cursorLimitLeft);

                curY[n[i]] =
min(curY[n[i]], cursorLimitBottom);

                curY[n[i]] =
max(curY[n[i]], cursorLimitTop);

                //printf("User %d |
Cursor X: %4d | Cursor Y: %4d\n",
user.getId(), curX[n[i]], curY[n[i]]);

        mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);

            }
                else

        mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);

        }

        else

        mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);

        #pragma endregion

        else

        mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);
                }
                else
if(user.isLost())
                {
```

```
        mUserTracker.stopSkeletonTrac
king(user.getId());

        //if(uIDDisplay == 2)

        //printf("\t\t\t\t");

        //printf("----- DROPPED USER
%d -----\n",n[i] + 1);


        #pragma region retain user
profile algo - REMOVE


        if(profileAlgo)
                                {

        if(isUserSlot1Occupied)

        {

                if(isUserSlot2Occupied)

                {
                        if(i == 0)

                        {

                                n[0] = n[1];
                        // slot 2 is dropped
/ slot 1 will fill for user in slot 2

                                if(n[0] ==
0)                      // slot 1 was
occupied by user 1

                                {


                        n[1] = 1;
                        // slot 2 will be occupied user 2

                                }

                                else
                                // slot 1
was occupied by user 2

                                {
```

```
                                                                {

n[1] = 0;                                                               curX[n[i]] =
// slot 2 will be occupied user 1                       desktopResolution.right * 1/4;
                                                                }
                        }
                                                                else
                }                                               {

                                                                curX[n[i]] =
isUserSlot2Occupied = false;                            desktopResolution.right * 3/4;
                                                                }
        }
                                                                curY[n[i]] =
        else                                            desktopResolution.bottom/2;

                                                                startPosX[n[i]] = -1;
isUserSlot1Occupied = false;
                                                                startPosY[n[i]] = -1;
}
                                }
                                                                mouseMotion(contact_[n[i]],curX[
#pragma endregion                                       n[i]],curY[n[i]],isDragged[n[i]]);
                                                                        }
                                                                        else
if(n[i] == 0)
```

```
                mouseMotion(contact_[n[i]],curX[
n[i]],curY[n[i]],isDragged[n[i]]);
                        }

        InjectTouchInput(2, contact_);
                        #pragma
endregion


        mUserFrame.release();
                }
                else
                {
                        cerr << "Can't get
user frame" << endl;
                }
                //-------------------------------
---KEYBOARD KEYS ------------------------
-----------//
                if
(GetAsyncKeyState(0x31)) //Key '1'
                {
                        startPosX[0] = -1;
                        startPosY[0] = -1;
                                curX[0] =
desktopResolution.right * 1/4;
                                curY[0] =
desktopResolution.bottom/2;
                        }
                        if
(GetAsyncKeyState(0x32)) //Key '2'
                        {
                                startPosX[1] = -1;
                                startPosY[1] = -1;
                                curX[1] =
desktopResolution.right * 3/4;
                                curY[1] =
desktopResolution.bottom/2;
                        }
                        if
(GetAsyncKeyState(VK_ESCAPE)) //
Key 'ESC'
                {
                                int msgboxID =
MessageBox
                                (
                                        NULL,
                (LPCWSTR)L"Exit the
program?",
```

```
        (LPCWSTR)L"FPS Interactive
Whiteboard",

        MB_ICONQUESTION |
MB_YESNO| MB_DEFBUTTON1 |
MB_SYSTEMMODAL
                        );
                        if(msgboxID ==
IDYES)
                        {

        if(conToPaint)
                                {

        system("taskkill /im
mspaint.exe");

        system("taskkill /f /im
FPSIWBGraphics3.exe");
                                }
                                std::cout
<< "..Exiting the program" << std::endl;
                                break;
                        }
    }
                }
                //---------------------------------
LOOP END---------------------------------//
                DeleteObject(hRedBrush);
                DeleteObject(hGreenBrush);
                DeleteObject(hBlueBrush);
                DeleteObject(hCyanBrush);
                DeleteObject(hYellowBrush);
                DeleteObject(hMagentaBrush);
                DeleteObject(hGrayPen);

                DeleteDC(dcText);
                DeleteDC(dcPopText);
                DeleteDC(dcRed);
                DeleteDC(dcGreen);
                DeleteDC(dcBlue);
                DeleteDC(dcCyan);
                DeleteDC(dcYellow);
                DeleteDC(dcMagenta);
                DeleteDC(dcPopText);
                DeleteObject(hDesktop);

                vsColorStream.stop();
                cout << "Color Stream Stopped"
<< endl;
                Sleep(200);
```

```
        vsDepthStream.stop();
        cout << "Depth Stream Stopped"
<< endl;
        Sleep(200);
        mUserTracker.destroy();
        cout << "User Tracker Stopped"
<< endl;
        Sleep(200);
        vsColorStream.destroy();
        cout << "Color Stream
Destroyed" << endl;
        Sleep(200);
        vsDepthStream.destroy();
        cout << "Depth Stream
Destroyed" << endl;
        Sleep(200);
        devDevice.close();
        cout << "Device Closed" << endl;
        Sleep(200);
        NiTE::shutdown();
        cout << "NiTE Shutdown" <<
endl;
        Sleep(200);
        OpenNI::shutdown();
        cout << "OpenNI Shutdown" <<
endl;
        return 0
}
```

RGB LED Pen

main.c

```
#include <avr/io.h>

int main(void)
{
// output pin (RGB LED)
DDRD |= (1 << PD3); // RED_LED
DDRD |= (1 << PD4); // GREEN_LED
DDRD |= (1 << PD5); // BLUE_LED

// input pin (PUSH BUTTONS)
DDRD &= ~(1 << PD0); // SWITCH_1
PORTD |= (1 << PD0); // enable pull-up
resistor
DDRD &= ~(1 << PD1); // SWITCH_2
PORTD |= (1 << PD1); // enable pull-up
resistor
DDRD &= ~(1 << PD2); // SWITCH_3
PORTD |= (1 << PD2); // enable pull-up
resistor
```

```
for(;;)
{
//RED
if (PIND & (1 << PD0)) { // is SWITCH_1
open?
PORTD &= ~(1 << PD3); // switch is
open, switch RED LED off
}
else
{
PORTD |= (1 << PD3); // switch is
closed, switch RED LED on
}

//GREEN AND YELLOW
if (PIND & (1 << PD1)) { // is SWITCH_2
open?
PORTD &= ~(1 << PD4); // switch is
closed, switch GREEN LED on
}
else
{
PORTD |= (1 << PD4); // switch is open,
switch GREEN LED off
}

//BLUE AND CYAN
if (PIND & (1 << PD2)) { // is SWITCH_2
open?
PORTD &= ~(1 << PD5); // switch is
closed, switch BLUE LED on
}
else
{
PORTD |= (1 << PD5); // switch is open,
switch BLUE LED off
}
}
return 0
```

CYM LED Pen

main.c

```
#include <avr/io.h>

int main(void)
{
// output pin (RGB LED)
DDRD |= (1 << PD3); // RED_LED
DDRD |= (1 << PD4); // GREEN_LED
DDRD |= (1 << PD5); // BLUE_LED

// input pin (PUSH BUTTONS)
```

```
DDRD &= ~(1 << PD0); // SWITCH_1
PORTD |= (1 << PD0); // enable pull-up
resistor
DDRD &= ~(1 << PD1); // SWITCH_2
PORTD |= (1 << PD1); // enable pull-up
resistor
DDRD &= ~(1 << PD2); // SWITCH_3
PORTD |= (1 << PD2); // enable pull-up
resistor

for(;;)
{
//MAGENTA
if (PIND & (1 << PD0)) { // is SWITCH_1
open?
PORTD &= ~(1 << PD3); // switch is
open, switch RED LED off
PORTD &= ~(1 << PD5); // switch is
open, switch BLUE LED off
}
else
{
PORTD |= (1 << PD3); // switch is
closed, switch RED LED on
PORTD |= (1 << PD5); // switch is
closed, switch BLUE LED on
}

//YELLOW
if (PIND & (1 << PD1)) { // is SWITCH_2
open?
PORTD &= ~(1 << PD4); // switch is
closed, switch GREEN LED on
PORTD &= ~(1 << PD3); // switch is
closed, switch RED LED on
}
else
{
PORTD |= (1 << PD4); // switch is open,
switch GREEN LED off
PORTD |= (1 << PD3); // switch is open,
switch RED LED off
}

//CYAN
if (PIND & (1 << PD2)) { // is SWITCH_2
open?
PORTD &= ~(1 << PD5); // switch is
closed, switch BLUE LED on
PORTD &= ~(1 << PD4); // switch is
```

```
closed, switch GREEN LED on
}
else
{
PORTD |= (1 << PD5); // switch is open,
switch BLUE LED off
PORTD |= (1 << PD4); // switch is open,
switch GREEN LED off
}
}
return 0
```

;

APPENDIX B

USER MANUAL

A. Turn-on all necessary hardware components

- Turn-on the Laptop
- Turn-on the LED projector
- Switch-on the LED Pens

B. Connect all necessary peripherals

- Connect the Kinect cable port to the Kinect Adaptor
- Plug the Kinect Adaptor's USB port to the laptop and electric plug to an AC electric source.
- Connect the LCD projector VGA/HDMI cable to the laptop.

C. Start the program

- Open "FPS Interactive Whiteboard.exe"
- Input the hand preference for user 1 and user 2.
- Read reminders and instruction of the program displayed on the screen as well is written on each of the LED pens.
- Start tracking.

D. User Tracking

- Make sure that the users are at least a meter away from the Kinect Sensor.
- Raise your hand at the shoulder level before entering the tracking area
- The first user must go into the tracking area and calibrate its position until is tracked.
- The second user must go into the tracking area and calibrate its position until is tracked

- The tracked users can now move their respective cursor on the desktop. Initially the left cursor is for the first user, and the right cursor is for the second user.

E. Using the LED pens

- To change the cursor behavior of each user, face the LED pen into the Kinect sensor and press the corresponding button for each mouse action per user.
- For the first user, the RGB pen will be used:
    - Red is for user 1 click event
    - Blue is for user 1 drag event
    - Green is for user 1 move event (default)
- For the second user, the CYM pen will be used:
    - Yellow is for user 2 click event
    - Magenta is for user 2 drag event
    - Cyan is for user 2 move event (default)
- Press the [MOVE] button on the LED Pen to provide label on which cursor is for the first or second user cursor.

F. Utility Keys

- In case cursor movement is moving incorrectly , tracked users must raise their hand on the shoulder level then press:
    - To reset second user cursor position.
    - To reset second user cursor position.
- Press [ESC] to exit the program.

APPENDIX C

TEST CASES

Check LED Status

Test Case No:	T01		
Module Name:	Check the LED status		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button of the RGB LED Pen	A red color will light up on the LED	A red color will light up on the LED	Successful
2.) Press green button of the RGB LED Pen	A green color will light up on the LED	A green color will light up on the LED	Successful
3.) Press blue button of the RGB LED pen	A blue color will light up on the LED	A blue color will light up on the LED	Successful
4.) Press cyan button of the CYM LED Pen	A cyan color will light up on the LED	A cyan color will light up on the LED	Successful
5.) Press yellow button of the CYM LED Pen	A yellow color will light up on the LED	A yellow color will light up on the LED	Successful
6.) Press magenta button of the CYM LED Pen	A magenta color will light up on the LED	A magenta color will light up on the LED	Successful

Program Start-up

Test Case No:	T02		
Module Name:	Program Start-up		
Scenario	Expected Result	Actual Result	Remarks
1. Open the program "FPS Interactive Whiteboard.exe"	If the Kinect Sensor is connected, a command prompt will open.	If the Kinect Sensor is connected, a command prompt will open.	Successful
	If the Kinect Sensor is not connected, an error message box will be displayed.	If the Kinect Sensor is not connected, an error message box will be displayed.	Successful
2. The program will ask for the hand preference of user 1. The user will either enter letter 'R' for right or 'L' for left.	The chosen hand preference of the user 1 will be used on the hand tracking.	The chosen hand preference of the user 1 will be used on the hand tracking.	Successful
	If the input is neither of the choices, it will ask again for an input.	If the input is neither of the choices, it will ask again for an input.	Successful

3. The program will ask for the hand preference of user 2. The user will either enter letter 'R' for right or 'L' for left.	The chosen hand preference of the user 2 will be used on the hand tracking.	The chosen hand preference of the user 2 will be used on the hand tracking.	Successful
	If the input is neither of the choices, it will ask again for an input.	If the input is neither of the choices, it will ask again for an input.	Successful
4. Directions will be displayed and will ask the user to press the [SPACE BAR] to continue to the program	The program will open a paint program, display the help graphics and will now start tracking motion and color.	The program will open a paint program, display the help graphics and will now start tracking motion and color.	Successful

Cursor Manipulation

Test Case No:	T03		
Module Name:	Test if the color manipulation changes the cursor behavior of the first user.		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button of the RGB LED Pen	If the red color was successfully found, the cursor of the first user will generate a click event	When the red color was successfully found, the cursor of the first user will generate a click event	Successful
2.) Press blue button of the RGB LED Pen	If the blue color was successfully found, the cursor of the first user will generate a left-click on-hold (drag) event	When the blue color was successfully found, the cursor of the first user will generate a left-click on-hold (drag) event	Successful
3.) Press green button of the RGB LED pen	If the green color was successfully found, the cursor of the first user will generate a left-click released (move) event	When the green color was successfully found, the cursor of the first user will generate a left-click released (move) event	Successful

Test Case No:	T04		
Module Name:	Test if the color manipulation changes the cursor behavior of the second user.		
Scenario	Expected Result	Actual Result	Remarks
3.) Press yellow button of the CYM LED Pen	If the yellow color was successfully found, the cursor of the second user will generate a click event	When the yellow color was successfully found, the cursor of the second user will generate a click event	Successful

3.) Press magenta button of the CYM LED Pen	If the magenta color was successfully found, the cursor of the second user will generate a left-click on-hold (drag) event	When the magenta color was successfully found, the cursor of the second user will generate a left-click on-hold (drag) event	Successful
3.) Press cyan button of the CYM LED Pen	If the cyan color was successfully found, the cursor of the second user will generate a left-click released (move) event	When the cyan color was successfully found, the cursor of the second user will generate a left-click released (move) event	Successful

Tracking Response Time

Test Case No:	T05		
Module Name:	Checks the average time the user will be tracked		
Scenario	Trial	Time (sec)	Average (sec)
First User	1	4.18	4.216
	2	4.14	
	3	4.33	
	4	4.25	
	5	4.18	
Second User	1	6.22	7.058
	2	6.35	
	3	7.64	
	4	8.47	
	5	6.61	

Cursor Response to Hand Tracking

Test Case No:	T06		
Module Name:	Cursor response to hand tracking for user 1		
Scenario	Expected Result	Actual Result	Remarks
Tracked user moves its right hand to change the cursor positon. (Hand preference chosen is the right, initialized on the the start-up)	Cursor is moved on the desired direction	Cursor is moved on the desired direction	Successful
Tracked user moves its left hand to change the cursor positon. (Hand preference chosen is the left, initialized on the the start-up)	Cursor is moved on the desired direction	Cursor is moved on the desired direction	Successful

Test Case No:	T07		
Module Name:	Cursor response to hand tracking for user 2		
Scenario	Expected Result	Actual Result	Remarks
Tracked user moves its right hand to change the cursor positon. (Hand preference chosen is the right, initialized on the the start-up)	Cursor is moved on the desired direction	Cursor is moved on the desired direction	Successful
Tracked user moves its left hand to change the cursor positon. (Hand preference chosen is the left, initialized on the the start-up)	Cursor is moved on the desired direction	Cursor is moved on the desired direction	Successful

Utility Keys

Test Case No:	T08		
Module Name:	Program Utility Keys		
Scenario	Expected Result	Actual Result	Remarks
1.Press [1] to reset cursor position and reference position of the user 1	The first user's cursor will be placed into its default position and a new reference position is taken.	The first user's cursor will be placed into its default position and a new reference position is taken.	Successful

1.Press [2] to reset cursor position and reference position of the user 2	The second user's cursor will be placed into its default position and a new reference position is taken.	The second user's cursor will be placed into its default position and a new reference position is taken.	Successful
3. Press [ESC] to exit the program	A message box will appear to ask if the user/s will exit the program. If pressed [okay] the program will continue to exit.	A message box will appear to ask if the user/s will exit the program. If pressed [okay] the program will continue to exit.	Successful
	A message box will appear to ask if the user/s will exit the program. If pressed [cancel] the program will continue to execute.	A message box will appear to ask if the user/s will exit the program. If pressed [cancel] the program will continue to execute.	Successful

Special Location and Condition Test

Test Case No:	S01			
Module Name:	Testing the system on a special location and condition			
Scenario				Results
Location	Lightning Condition	Area Condition	Other Conditions	
ICS Lab 2 (University of Santo Tomas, Roque Ruaño BLDG, 3F)	24 Fluorescent lamp (all switched-on)	1.9m by 7.5m	Students 2m away	Excessive light on the computer laboratory weakens the color tracking range of the system
Dev Room (University of Santo Tomas, Roque Ruaño BLDG, 2F)	3 Fluorescent lamp	8.5m by 9.5m	Professors 2m away	The wall of the room bounces the infrared light scattered by the Kinect's IR Emitter making the user tracking inaccurate.
				The yellow walls of the room is very closed that it also detected by the color tracking

Program Speed Test

Test Case No:	S02			
Module Name:	esting the speed of the running program per iterationT			
Scenario	Program Date Version	Trial	Ave sec. per iteration	Total Average (sec)
Testing the program speed	4/27/2016	1	0.093	0.095
		2	0.098	
		3	0.094	
Testing the program speed without color tracking	4/27/2016	1	0.037	0.037
		2	0.036	
		3	0.039	
Testing the program speed with an algorithm that minimize the load of the color tracking by using timers	4/29/2016	1	0.037	0.035
		2	0.033	
		3	0.035	

Graphics Test

Test Case No:	S03		
Module Name:	Test Summary of WM_Paint Message on the Windows Desktop		
Scenario:	Check the graphics interaction of the WM_Paint message to the Windows OS ability to refresh the desktop image.		
Graphics Category	Description	Problems Occured	In conclusion
Testing Animated Graphics	Graphics that moves in real time	Smudges the desktop image	Not usable
		Slows down the program	
Testing Still Graphics	Still graphics shown on the entire run of the program	Graphic flickers on dynamic pages of the system	Can be used only on static window frames of the
		Slows down the program	
Tseting Pop-up Graphics	Single occurence once per user action	N/A	Usable

Lighting Conditions for LED Pen Detection

Dark Environment - only light reflected from the LCD projector on the whiteboard is light

Test Case No:	C01		
Module Name:	Test if the program can detect the colors from RGB LED at 0.5m in a Dark Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C02		
Module Name:	Test if the program can detect the colors from RGB LED at 1m in a Dark Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program		Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful

4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C03		
Module Name:	Test if the program can detect the colors from RGB LED at 1.5m in a Dark Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	No color detected	Failed
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C04		
Module Name:	Test if the program can detect the colors from RGB LED at 2m in a Dark Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful

3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	No color detected	Fail
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	No color detected	Fail

Low-light Environment (80 watts of regular long fluorescent lamps) - In this Scenario, half of the lights of the classroom turned off

Test Case No:	C05		
Module Name:	Test if the program can detect the colors from RGB LED at 0.5m in a Low-Light Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C06
Module Name:	Test if the program can detect the colors from RGB LED at 1m in a Low-Light Environment

Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C07		
Module Name:	Test if the program can detect the colors from RGB LED at 1.5m in a Low-Light Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C08		
Module Name:	Test if the program can detect the colors from RGB LED at 2m in a Low-Light Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No color detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	No color detected	Failed
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	No color detected	Failed
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	No color detected	Failed
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	No color detected	Failed
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	No color detected	Failed

Regular Classroom Environment (480 watts regular long fluorescent lamps) - In this scenario, all of the lights in the classroom are turned on

Test Case No:	C09		
Module Name:	Test if the program can detect the colors from RGB LED at 0.5m in a Classroom Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No color detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	No color detected	Failed
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	No color detected	Failed
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C10		
Module Name:	Test if the program can detect the colors from RGB LED at 1m in a Classroom Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C11		
Module Name:	Test if the program can detect the colors from RGB LED at 1.5m in a Classroom Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful

6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C12		
Module Name:	Test if the program can detect the colors from RGB LED at 2m in a Classroom Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No color detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	No color detected	Failed
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	No color detected	Failed

Dark Environment - only light reflected from the LCD projector on the whiteboard is light

Test Case No:	C13		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 0.5m in a Dark Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	No Color Detected	Failed
3.) Press blue button	Only the blue color will	Only the blue color will	Successful

and move the pen from left to right	be tracked by the program	be tracked by the program	
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C14		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 1m in a Dark Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No Color Detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C15		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 1.5m in a Dark Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to	Only the red color will be tracked by the program	No Color Detected	Failed

right			
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	No color detected	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C16		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 2m in a Dark Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No Color Detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Low-light Environment (80 watts of regular long fluorescent lamps) - In this Scenario, half of the lights of the classroom turned off

Test Case No:	C17		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 0.5m in a Low-Light Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C18		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 1m in a Low-Light Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from	Only the cyan color will be tracked by the	Only the cyan color will be tracked by the	Successful

left to right	program	program	
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C19		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 1.5m in a Low-Light Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Regular Classroom Environment (480 watts regular long fluorescent lamps) - In this scenario, all of the lights in the classroom are turned on

Test Case No:	C20		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 2m in a Low-Light Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left	Only the red color will be tracked by the program	Only the red color will be tracked by the	Successful

to right		program	
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	T21		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 0.5m in a Classroom Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No Color Detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	No Color Detected	Failed
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	No Color Detected	Failed
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	T22		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflectorat 1m in a Classroom Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	T23		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 1.5m in a Classroom Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful

6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	T24		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 2m in a Classroom Environment		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Dark Environment - only light reflected from the LCD projector on the whiteboard is light

Test Case No:	C25		
Module Name:	Test if the program can detect the colors from RGB LED at 0.5m in a Dark Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful

3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C26		
Module Name:	Test if the program can detect the colors from RGB LED at 1m in a Dark Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C27		
Module Name:	Test if the program can detect the colors from RGB LED at 1.5m in a Dark Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	No color detected	Failed

Test Case No:	C28		
Module Name:	Test if the program can detect the colors from RGB LED at 2m in a Dark Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No color detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	No color detected	Failed
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	No color detected	Failed

6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	No color detected	Failed

Low-light Environment (80 watts of regular long fluorescent lamps) - In this Scenario, half of the lights of the classroom turned off

Test Case No:	C29		
Module Name:	Test if the program can detect the colors from RGB LED at 0.5m in a Low-Light Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C30		
Module Name:	Test if the program can detect the colors from RGB LED at 1m in a Low-Light Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful

3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C31		
Module Name:	Test if the program can detect the colors from RGB LED at 1.5m in a Low-Light Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No color detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	No color detected	Failed
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	No color detected	Failed
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	No color detected	Failed
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	No color detected	Failed
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	No color detected	Failed

Test Case No:	C32		
Module Name:	Test if the program can detect the colors from RGB LED at 2m in a Low-Light Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks

1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No color detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	No color detected	Failed
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	No color detected	Failed
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	No color detected	Failed
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	No color detected	Failed
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	No color detected	Failed

Regular Classroom Environment (480 watts regular long fluorescent lamps) - In this scenario, all of the lights in the classroom are turned on

Test Case No:	C33		
Module Name:	Test if the program can detect the colors from RGB LED at 0.5m in a Classroom Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the	Only the magenta color will be tracked by the	Only the magenta color will be tracked by the	Successful

pen from left to right	program	program	

Test Case No:	C34		
Module Name:	Test if the program can detect the colors from RGB LED at 1m in a Classroom Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No color detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	No color detected	Failed
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	No color detected	Failed
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	No color detected	Failed
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	No color detected	Failed
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	No color detected	Failed

Test Case No:	C35		
Module Name:	Test if the program can detect the colors from RGB LED at 1.5m in a Classroom Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No color detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	No color detected	Failed
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	No color detected	Failed
4.) Press cyan button and move the pen	Only the cyan color will be tracked by the program	No color detected	Failed

from left to right			
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	No color detected	Failed
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	No color detected	Failed

Test Case No:	C36		
Module Name:	Test if the program can detect the colors from RGB LED at 2m in a Classroom Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No color detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	No color detected	Failed
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	No color detected	Failed
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	No color detected	Failed
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	No color detected	Failed
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	No color detected	Failed

Dark Environment - only light reflected from the LCD projector on the whiteboard is light

Test Case No:	C37		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 0.5m in a Dark Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful

from left to right			
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C38		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 1m in a Dark Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful

right			
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C39		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 1.5m in a Dark Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C40		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 2m in a Dark Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Failed

2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	No color detected	Failed
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	No color detected	Failed
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	No color detected	Failed

Low-light Environment (80 watts of regular long fluorescent lamps) - In this Scenario, half of the lights of the classroom turned off

Test Case No:	C41		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 0.5m in a Low-Light Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful

5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C42		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 1m in a Low-Light Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C43
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 1.5m in a Low-Light Environment affected by the sunlight coming from the windows and glass doors

Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C44		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 2m in a Low-Light Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No color detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	No color detected	Failed
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	No color detected	Failed
4.) Press cyan button and move the pen from left to	Only the cyan color will be tracked by the program	No color detected	Failed

right			
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	No color detected	Failed
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	No color detected	Failed

Regular Classroom Environment (480 watts regular long fluorescent lamps) - In this scenario, all of the lights in the classroom are turned on

Test Case No:	C45		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 0.5m in a Classroom Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C46		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 1m in a Classroom Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	Only the red color will be tracked by the program	Successful
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	Only the green color will be tracked by the program	Successful
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	Only the blue color will be tracked by the program	Successful
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	Only the cyan color will be tracked by the program	Successful
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	Only the yellow color will be tracked by the program	Successful
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	Only the magenta color will be tracked by the program	Successful

Test Case No:	C47		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 1.5m in a Classroom Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No color detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	No color detected	Failed
3.) Press blue	Only the blue color will be tracked by	No color	Failed

button and move the pen from left to right	the program	detected	
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	No color detected	Failed
5.) Press yellow button and move the pen from left to right	Only the yellow color will be tracked by the program	No color detected	Failed
6.) Press magenta button and move the pen from left to right	Only the magenta color will be tracked by the program	No color detected	Failed

Test Case No:	C48		
Module Name:	Test if the program can detect the colors from RGB LED extended with 4 diameter aluminum reflector at 2m in a Classroom Environment affected by the sunlight coming from the windows and glass doors		
Scenario	Expected Result	Actual Result	Remarks
1.) Press red button and move the pen from left to right	Only the red color will be tracked by the program	No color detected	Failed
2.) Press green button and move the pen from left to right	Only the green color will be tracked by the program	No color detected	Failed
3.) Press blue button and move the pen from left to right	Only the blue color will be tracked by the program	No color detected	Failed
4.) Press cyan button and move the pen from left to right	Only the cyan color will be tracked by the program	No color detected	Failed

APPENDIX D

CURRICULUM VITAE

XAVIER GIRARD M. FAJARDO

Address: Eastwood Residences P1 B36 L24
Rodriguez,Rizal
Mobile No.: (+63) 9166925906
e-mail: xavier_fajardo@yahoo.com

PERSONAL INFORMATION

Gender: Male
Birth place: Quzon City
Civil Status: Single

EDUCATIONAL BACKGROUND

Degree	School	Date
Bachelor of Science in Information Technology	University of Santo Tomas España Boulevard, Sampaloc, Manila	June 2011 - present
Secondary Education	University of Santo Tomas High School España Boulevard, Sampaloc, Manila	June 2007 - March 2011
Primary Education	St. Rita Collge Manila Plaza Del Carmen, Quiapo Manila	June 2000 – March 2011

TRAININGS AND CERTIFICATION

- Philippine National IT Standards(PhilNITS)Level 1 Certified (2013)
- DB2 Fundamentals (2013)
- CCNA Routing and Switching: Introduction to Networks (2014)
- CCNA Routing and Switching: Routing and Switching Essentials (2014)
- CCNA Routing and Switching: Scaling Networks (2015)
- CCNA Routing and Switching: Connecting Networks (2015)
- Intern at Wilcon Builders Depot (2015)
- Intern at University of Santo Tomas Hospital (2015)

TECHNICAL SKILLS

- Languages : sql,xml,html,css, basic c/c++ , basic java , unix commands
- Platforms : windows xp/7/8/10
- Concepts : Database management system. web designing, knowledgeable in configuring various cisco devices, computer troubleshooting and system administrating
- Software : ibm clp, notepad , command prompt
- IDE : notepad++, dev-c++,netbeans

Julian Patric Joshua G. Paman

Contact #: 09179230796 / 9550286
Email: Joshpaman@gmail.com
#6 Major Dizon Ave. IVS Marikina

OBJECTIVES

I am looking for a suitable entry level position that I may have an opportunity to apply my academic experiences in the industry. I am also expecting to apply the teachings from the excellent program provided by the University of the Philippines during my 6 month internship with them. I promise that I will fulfill all responsibilities given to me with enthusiasm and to the best of my abilities.

EDUCATIONAL BACKGROUND

Tertiary Education
- University of Santo Tomas
- Sampaloc, Manila 1008
- BS Information Technology
- 2010 – Present

Secondary Education
- Immaculate Conception Cathedral School
- Lantana St. Quezon City
- 2006 – 2010

SKILLS

Oriented in Microsoft office applications
- Microsoft Word
- Microsoft Power Point
- Microsoft Access
- Microsoft Excel

Familiar with many different operating systems
- Windows
- UNIX
- LINUX

Capable of most media editing with the use of
- Adobe Photoshop
- Adobe After Effects
- Adobe Illustrator

Basic knowledge in Cisco Networking
- Routing
- Switching

Hardware and software installation Oriented in many programming languages
- DB2
- PHP, HTML, CSS
- Java, JavaScript
- C++
- Assembly
- LARAVEL – PHP Framework

JOHN ANGELO D.C. PARAYNO

Address: 9 Almaciga Street, Project 3, Quezon City
Mobile No.: 0917-551-6965
Landline No.: 374-9904
E-mail Address: j.angelo.parayno@gmail.com

PROFILE

Skills and Strengths:	excellent programming and problem solving skills, proficient in graphic design theory, creative in design
Qualities and Traits:	organized, detail-oriented, good listener, objective, passionate in learning and discovery
Career Ambitions:	Systems Designer, Game Developer, Innovation Leader, Software Engineer, Network Engineer, Network Sys. Admin

EDUCATIONAL BACKGROUND

Degree	School	Date
Bachelor of Science in Information Technology	University of Santo Tomas España Boulevard, Sampaloc, Manila City	2010 – present
Secondary Education	Claret School of Quezon City, Mahinhin Street, U.P. Village, Diliman, Quezon City	2006 - 2010
Primary Education	Claret School of Quezon City, Mahinhin Street, U.P. Village, Diliman, Quezon City	1999 - 2006

TECHNICAL EXPERIENCE

Languages	:	C/C++, Java, Assembly
Layouts/Scripting	:	HTML, CSS, JavaScript, JSP
Platforms	:	Windows, Linux, Cisco IOS
Software	:	Adobe Photoshop, Adobe Premiere, MS Office, DB2, MySQL, Audacity, Apache, NetBeans, Eclipse
Others	:	Computer Assembling and Maintenance, Troubleshooting, Cisco Networking, Database Management

Karlene C. Siy

Address: 806 Severino Reyes Sta. Cruz
Mobile No.: (+63) 9274677090
E-mail: karlene.siy@gmail.com

OBJECTIVES

- To enhance my skills as an Information Technology graduate By means of experience and training
- To be a reliable teammate for future projects

PERSONAL INFORMATION

Sex: Female
Birth date: March 31, 1993
Birth palce: Metro Manila
Citizenship: Filipino
Civil Status: Single

EDUCATIONAL BACKGROUND

Degree	School	Date
Bachelor of Science in Information Technology	University of Santo Tomas España Boulevard, Sampaloc, Manila	June 2011 – May 2016
Secondary Education	Hope Christian High School 1242 Benavidez St, Manila	June 2007 - March 2011
Primary Education	Uno High School 1440 Alvarado Extension, Manila	June 2000 – March 2007

TRAININGS AND CERITIFICATIONS

- Project Management in Hewlett-Packard University of Santo Tomas
- Wilcon Builder's Depot Information Technology Department Internship
- Metropolitan Medical Center Information Technology Department Internship

WORK EXPERIENCE

Wilcon Builder's Depot
Office Staff
1274 EDSA Balintawak 1106
April 2011 to June 2011

TECHNICAL SKILLS

Languages : C/C++, Java, Assembly
Platform : Windows XP/7/8, Linux
Concepts : System Quality Assurance, Web Design & Development, Interface Design and Web Technologies, Operating Systems, System Analysis and Design, Data Communications and Networking
Software : Microsoft Office, Adobe Photoshop, Cisco Packet Tracer