Table Top Computer Interface Using A Depth Sensor

Katie Seggerman and Andy Perez
Department of Electrical and Computer Engineering
Tagliatela College of Engineering
Dr. Christopher Martinez

Abstract
Currently popular methods for touch screen development are impractical for large-scale use due to cost and other limitations. The purpose of this project was to create a feasible interface for a large-scale tabletop touch screen. The method used for touch detection was a Kinect depth sensor. This sensor implements a depth camera that is able to distinguish between objects based on their relative distances using trigonometry. A tabletop setup was designed for testing purposes. This included an aluminum-framed table, a projection screen, a Kinect sensor mounted above the tabletop, and a projector reflecting onto the screen. In order to test the viability of the Kinect as a depth-reliant touch screen, various applications were written in a C++ code. This program examined the sensors ability to accurately track touch motion across the screen and explored other gesture dependent applications. The Kinect sensor could recognize where the user was touching the tabletop and the code controlled how the projection responded to the touch.

Introduction

Emerging technologies provide a variety of design options to create devices. A specific device, such as a touchscreen, that enhances the human experience with a computer falls into the category of human computer interaction. This is a multidisciplinary field that encompasses areas of technology, art design, and ergonomics. Currently, the most widely accepted technologies for touchscreens are capacitive, resistive, and infrared. Each design has different properties of durability, accuracy, versatility, and cost efficiency.

Recently, depth cameras have been introduced into the field of human computer interactions. Depth sensor technology introduces the possibility of a more cost efficient touch screen with more versatile, dimension-based gestures. The Kinect sensor has a distinct set of cameras. The depth camera detects and distinguishes between objects based on distance from the sensor using triangulation and trigonometry.

This research explored the accuracy of the Kinect depth sensor as a large-scale touch screen. The goal of this project was to implement a low cost design, which could turn any flat surface into a depth-reliant touch screen. To do this, a C++ program was written to test the accuracy and versatility of the Kinect depth sensor. The program would allow for calibration of a large screen with multiple users. After assessing the functionality of the applications, the feasibility of the Kinect sensor as a mode for touch detection could be evaluated.

Hand Measurements

A study was conducted to determine the optimal hand measurements to use for touch detection using a depth camera. This study also observed how users prefer to interact with a large-scale touch screen.

The experiment involved a tabletop, with dimensions 2ft width, 4ft length, 3ft height, and a static desktop image with icons at specific locations. Participants were asked to touch four icons. The first was in the center, and then the bottom left corner, middle, and far right corner. The participants were also asked to use preferred swipe and zoom gestures.

From each touch, the hand was measured at three points, the distal inter-phalangeal joint (DIP), proximal inter-phalangeal joint (PIP), and metacarpophalangeal joint (MCP). These Joints are shown in Figure 1.

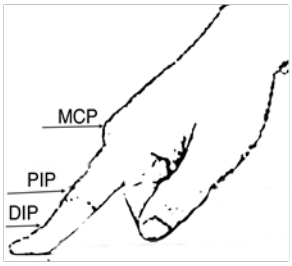


Figure 1: Measurement Location

Table 1: DIP Measurements

	Total (in cm)	Male (in cm)	Female (in cm)
Near μ	1.87	1.95	1.63
Near σ	0.45	0.48	0.19
Center μ	1.84	1.95	1.47
Center σ	0.41	0.39	0.23
Middle μ	1.76	1.91	1.3
Middle σ	0.43	0.37	0.28
Far μ	1.64	1.77	1.23
Far σ	0.45	0.41	0.29

Table 2: MCP Measurements

	Total (in cm)	Male (in cm)	Female (in cm)
Near μ	7.41	7.8	6.38
Near σ	1.3	1.14	1.23
Center μ	7.05	7.25	6.42
Center σ	0.92	0.89	0.71
Middle μ	7.41	7.81	6.15
Middle σ	1.23	0.92	1.3
Far μ	7.14	7.44	6.2
Far σ	1	0.78	1.11

It was observed that most participants used multiple fingers to swipe. All participants used a two finger pinching motion to zoom. The average measurements and gesture observations were used to write the program.

Setup and Design

In order to viably test the Kinect sensor, a table was designed which could mount the camera above the projection. The table was fashioned from 40/40 aluminum bars of dimensions 2ft width, 4ft length, and 3ft height. The depth sensor was mounted 3 ft above the tabletop. The actual tabletop was composed of two sheets of Plexiglas encasing a projection screen. From below the table, a projector was positioned to reflect off a mirror, which angled the projection onto the screen.

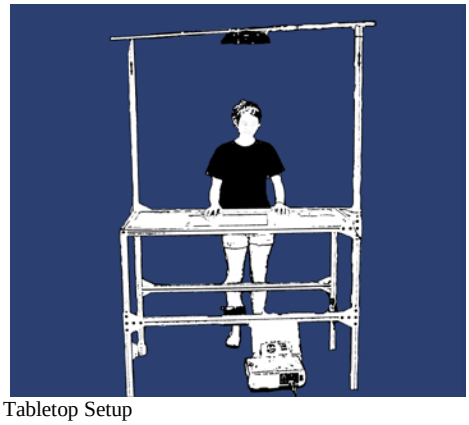


Figure 2: Tabletop Setup

Touch Detection

The first phase in testing the accuracy of the Kinect depth sensor was to assess its ability to track the motion of a person's finger across the table. To do this, multiple thresholds were established using C++ to detect objects within specified distance ranges.

To determine the threshold for if a person was touching the table, the initial distance of the depth sensor in relation to the table was recorded using a key press command. The code then checked if the depth sensor found an object within a range between the initial distance and the average measurement for the DIP joint. This range was used as the touch threshold.

The detected pixels within the range were recorded as blobs. The program can specify how many separate blobs to detect and the minimum and maximum pixel size. Using the centroid function, the coordinates for the center of the blob were recorded. Blob detection enabled the program to track the coordinates of the finger on the table.

In order to track more than one finger at a time, the code had to account for the frame updating. The Kinect updates at 30 frames per second and does not consistently detect where the finger is located in each frame. Due to the lapse between frames, the coordinates for finger one and finger two would alternate. This was accounted for in the program by constantly checking the new coordinates for the two fingers against the coordinates from the previous frame.

This same process was used to detect the tops of the hands in a separate threshold determined from the MCP hand measurements. By comparing the centroids of the finger and the hand, it could be determined whether the right or left hand was used.

Figure 3: Blob Centroids

Calibration

Since the coordinate grid from the depth sensor detecting the finger and the coordinate grid of the projection onto the table differed, a calibration function was necessary. The calibration process required the user to touch the top and bottom corners of the projected image in order to create a ratio to write the coordinates detected to where the finger was located above the image.

The calibration process was set to begin when the user pressed a set key, 'o', on the keyboard. This caused two squares to appear, one at the upper left corner of the program and one at the lower right corner. The user was then required to touch the upper corner, wait, and then touch the bottom corner. The coordinates recorded from the two touches were used as a ratio. This relationship between Kinect sensor coordinates to program coordinates was used to locate the touch in compared to the program.

The calibration also excluded any blobs detected outside the range of the projected image to be recorded as coordinates.

The altered coordinates were 1-to-1 with the program. Using these values, many other applications were written to test the versatility of the Kinect depth sensor.

Applications

The applications written to test the Kinect depth sensor were linked in the program using a menu screen with navigation options. This gave the program the illusion of a typical touch screen.

Coloring

The coloring application analyses the ability of the Kinect to continuously track the motion of a touch on the tabletop. Initial attempts to display the motion provided evidence that the tracing was not fluid. To adjust for the discontinuity between points, a line function was written to link the centroids of each respective finger using the slope between the current and previous points. A time elapsed function determined whether a segment should break. This was controlled by the total frames that passed between the recognition of two centroids from the same finger. If more than 10 frames had elapsed, the line would disconnect from the previous point.

To draw, the pixels within a 3x3 range of the centroid coordinate and line function were changed to a designated color. The user could alternate the color from an array. An erase function drew to a 9x9 range of pixels matching the background. Using the multiple colors, eraser, and clear options the user could draw to the screen.

The application verified that the Kinect could be used to accurately track a touch on the tabletop.

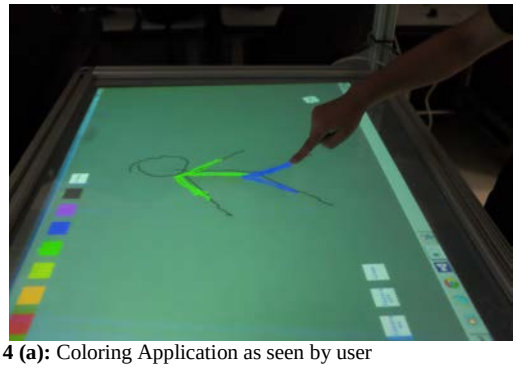


Figure 4 (a): Coloring Application as seen by user

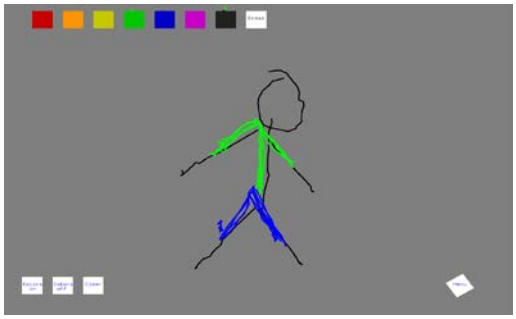


Figure 4 (b): Coloring Application as seen on computer

Puzzle

The puzzle application required the user to arrange the numbers one through nine in numerical order on a predefined grid. This section of the program established that the Kinect could move images using either one or two fingers.

While a touch was implemented on a segment of the puzzle, that piece would redraw continuously to the next coordinates of the finger that was detected. If a piece reached within 50 pixels of its designated grid space, the piece would align itself within the grid.

This application proved to be more accurate when only one finger was touching the board. If multiple touches were detected, there was a higher probability the blob detection would alternate between blobs, thus switching the location of the pieces.

Zoom

The zoom was written to assess the ability to resize images with the Kinect. To do this, an image was set in the center of the screen with options to switch the designated image. This function exploits the separate entities created for multiple finger detection. The Kinect registered the coordinates of the two blobs and determined the difference in both the x and y distances. The image was continually resized to the updated dimensions relative to the differences calculated.

The application confirmed that a two finger pinching gesture could be used for zooming in and out. If only one finger were detected, the image would not resize.

Swipe

This application was used to alternate between images using two methods.

The first method implemented a two finger swiping gesture to move an image, numbered one through four, to the edge of the screen. If the image reached a specified limit at either edge, the next respective image would appear. If only one finger was detected, the image would remain stationary.

The second method moved the images in the same order using a hovering motion. Creating a third threshold above the screen did this. This method was used to show the multiple advantages of using a depth sensor as a touch screen.

The swipe established that the common gesture of a two-finger swipe could be used. This application also experimented with non-touching gestures.

Rotate

This application was written to display the ability to rotate 3-dimensional objects. A 3-D box was anchored to the center of the screen. If a blob was detected on the screen it was recorded as the starting point as if it were the center of the box. The user was able to move their finger in any direction and the center of the box would rotate respectively.

Conclusion

Overall, the design established a feasible interface for a touch screen. The Kinect sensor was able to accurately detect and track the user's finger on the tabletop. The methods discussed for calibration proved to precisely scale the coordinates for the touch to a 1-to-1 ratio with the location of the projected image. This allowed the numerous applications to function properly.

Each individual function evaluated a distinct aspect of the depth sensors versatility. Through testing and observation of the program, conclusions were drawn as to the capabilities of the Kinect. The investigation of these processes construed the capacity of the Kinect to recognize touch within specified threshold regions. Through this research, the ability to track motion with multiple blob detection is displayed and the program allowed for further examination of gesture recognition. The variety of applications demonstrated the flexibility of the Kinect and ensured it was a viable option for touch detection.

Due to the testing process, the properties of the Kinect depth sensor versus alternate modes of touch detection could be evaluated. Compared to a typical capacitive touch screen, the Kinect method proved to have a larger delay and less continuous detection. However, the depth sensor technique allowed for multiple users and a more cost efficient large-scale design. Using depth sensor technology also enabled the use of non-touching gestures, as demonstrated in the swipe application. This capability of a depth sensor to recognize motion within numerous distance thresholds is an advantage over the other options for touch recognition. These properties discovered in the research proved the Kinect depth sensor to be a versatile option for a multi-user touch screen.

This research could be continued to further explore the capabilities of the Kinect and non-touching gestures. The field of human computer interaction is never static. With new technologies emerging, the accuracy and precision of depth cameras has improved and will continue to progress, enabling more innovative models to be designed for touch detection.

References

1. Wilson, Andrew D. Using a Depth Camera as a Touch Sensor, ITS 2012: Devices & Algorithms
2. Kim, J., Park, J., and Lee, H. HCI(Human Computer Interaction) Using Multi-touch Tabletop Display (Gwangju, Korea)
3. Zhang, H. EVALUATING FINGER ORIENTATION FOR POSITION AWARENESS ON MULTI-TOUCH TABLETOP SYSTEMS (University of Manitoba Winnipeg, Manitoba, Canada)

Acknowledgements

We would like to thank our research advisor and mentor, Dr. Christopher Martinez. His knowledge and guidance throughout the design and implementation process enabled us to create a successful program. We would also like to thank Mark Morton and John Kelly for their assistance in designing and constructing the set up for our project. And of course, we would like to thank everyone involved in the SURF program and the donors for granting us this opportunity. The contributions made and support given to us throughout this project helped us realize our vision for a depth sensor touch screen.

Biographies

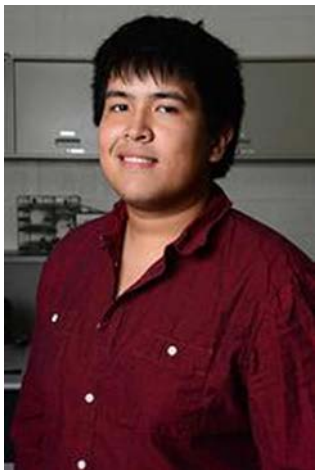
Katie Seggerman

Katie Seggerman is a third-year electrical engineering student at the University of New Haven with plans to pursue a minor in computer engineering. She is the vice president of the society of women engineers on campus and secretary of the game-making club. Currently she works as a teaching assistant for digital systems I. After receiving her bachelor's degree she hopes to get a job on the digital design side of electrical engineering, then pursue a further degree.

Katie Seggerman

Andy Perez

Andy Perez is a third-year electrical engineering student at the University of New Haven with intentions to minor in computer engineering. He is the Vice president of a game-making club and is a member of multiple music programs such as the UNH chargers marching band, concert band and wind ensemble. He currently works as an enhanced visitors guide for the Tagliatela College of Engineering. After receiving his bachelor's degree, Andy intends on obtaining a job in the field of green energy before continuing his studies to pursue a master's degree.



Andy Perez