KinectDraw: A Kinect based digital tool to visualize 2D figures and edit them in real time

Rohit Vairamohan
School of Information
University of Michigan
rovairam@umich.edu

ABSTRACT

KinectDraw is an interactive tool that uses Kinect[8] for people to interact with shapes on a 2D screen. The user is able to change the look, feel and the shape of the object he makes on the screen. It uses the location of his right hand on the x-y plane to mark a pointer on the screen and subsequently adjust the coordinates of the edges of the object that the user is interacting with. It is a setup that focuses more towards changing the shape of the object on screen. The coordinate system is relative to the user's body position. I shall make an effort to describe the design of the prototype and the various questions that arose in the design phase as well as future implementations of such technology.

Keywords

NIME, Kinect, visual motion tracking, draw, Bezier curve

1. INTRODUCTION

In today's world artists and even other professionals who deal with interacting with different interfaces to produce work on a two-dimensional surface, feel the necessity to produce a digital copy of the artifact as well. To accomplish this they want to move beyond the regular use of a mouse pointer and being constrained within the WIMP (Window Icon Mouse Pointer) interface. People want to be able to be free from these constraints and move more freely and have all the control that they would have in a real-world drawing pad.

In this project, I am trying to understand the feasibility and the freedom that the user deserves or wants to have when he is editing a picture. I specifically want to see the advantages of using the Kinect sensor and allowing the user all the freedom that he expects from a physical medium. With the use of a prototype the advantages and the drawbacks of the free movement the user is entitled to by using the Kinect Sensor.

I go on to discuss the related work that has been done and the literature I read about the same in this paper. I will also delve into the implementation of my prototype, the design, the working principle and the various challenges and constraints I faced during the design phase. Finally I will go onto discuss the areas where the prototype falls short and the future implementations of the same.

2. Related Works

There has been some interesting research done in the field of art replication from a gesture tracking system using feasible sensing technologies. One such technology is 'Kinect Paint'. Kinect Paint is a mouse and keyboard free drawing application in which the user can simply use his or her natural hand motions to paint a picture on a computer screen (Letendre et Al, 2011). It as a very interesting method they use. The user can simply wear thin gloves on his or her hands. These gloves are designed to have unique color patterns on specific fingertips and the palm of the hand. Then, using the 3D depth sensors and RGB camera on a Microsoft Kinect device, the user's hand motions can be tracked and detected.

Thus in this technology you have to use a physical object (the gloves) to detect you precise positions. This constrains the user in certain ways and he cannot feel one with the freedom that he shares in the case of real painting/drawing on a piece of paper. The mere use of objects like these take the experience away from the user of using a more skeuomorphic (Skeuomorph, 2013) and realistic feel to it. What I mean to say by using the term skeuomorphic is a sensation that the user is close to what the interaction in the real world would be like. With the advent of the new Kinects, RGB detection is no longer necessary and hence body movements can be recognized without the need for any physical object to be strapped onto the individual.

Also this technology allows the users to erase the mistakes that he has made rather than facilitate him to edit the same. There is no way the user can make minor adjustments to the figure that he has drawn. Thus the user can either draw the perfect figure or completely erase it and start afresh. This is a hindrance for the user as even if there is a minor mistake he as to go back and erase the entire picture. Although is it necessary to have a similarity to the physical experience of drawing on a piece of paper, having no edit feature is a bit cumbersome for the user as there is a slight disconnect of not having the piece of paper to draw on. Hence to bridge that disconnect a having a feature like that is imperative.

The system that I have proposed is slightly different than the one described above. My main goal was to target the non-editing feature of this technology. I want users to be able to edit the two-dimensional figure that they have drawn. Secondly I want them to feel as close to the real experience as possible and not be constrained to be using any particular device unless it is absolutely necessary.

3. Design and Implementation

3.1 Design

The main intention of this project was to help users to edit the figures that they had already drawn by using hand gestures. I was particularly interested in what kind of interactions would the user perceive to be able to be executed on the 2D surface using the Kinect Sensor. The other thing worth noting was the possible approaches that might be taken to move the control points of the edge points of the figure around. To answer these questions I used the Kinect to track the users hand movements. I chose to track only the right hand since painters and artist predominantly use one hand to make their masterpiece. A red circle on the screen depicts the position of the hand-controlled pointer. A body-centered frame of reference is used to track the position of the hand. The advantage of doing so is that no matter where in space the user is in he will comfortably be able to move the pointer all across the screen using his hand as there would be no change in the coordinate system with respect to his movement in that space. Please note that that space should be within the sensing ranges of the Kinect.

For the purpose of this demonstration, I prepopulated the figure and already kept it ready for the user to edit. The figure is that of a Bezier curve (Bézier curve, 2013). A Bézier curve is a parametric curve frequently used in computer graphics and related fields.

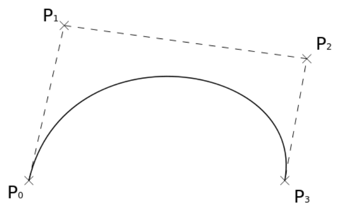

Figure 1 Bezier Curve

Also I avoided using the depth information provided by the Kinect as I wanted to give the user a perception that he was interacting with a two-dimensional surface and not in a three-dimensional environment. So basically the user uses the red pointer on the screen controlled by his right hand to point and decisive points, control points or edge points of the figure. Then an event is initiated by the user to choose that particular control point and based on the movement of his hands the control point is moved across the screen. I use a mouse to initiate this event. The mouse is held in the users right hand, which he uses to control the pointer. Note that the mouse is just used as a source for triggering the event and in no way influences the position of the pointer. This results in a change of shape of the two-dimensional figure on the screen.

3.2 Implementation

Figure 1 gives a clear view of the entire process right from the users input to the output that he sees. The Kinect 360 reads the users movements at first. It is parsed into Synapse, which is then converts this data into skeletal data by tracking the important joints of the body. For synapse to trace skeletal data it is important for the user to strike a T-pose. A T-pose is a position where both the users' arms are extended outwards in parallel to his shoulders and in the sides not in front of him. His arms have to be bent to ninety degrees at the elbow so that his fingertips point towards the ceiling. It is shown in figure 2. Next step is to send this skeletal data into a graphic programming environment to produce the desired output for the various actions of the user. The environment that I chose to use is 'openFrameworks' [7]. Though a lot of support and documentation is not available for openFrameworks it is an excellent environment to work with computer graphics. I send the data using an addon for openFrameworks called Synapse Streamer [9]. You can choose which joint you want to track using Synapse streamer as well. I have chosen to track the right hand for the purpose of this project.

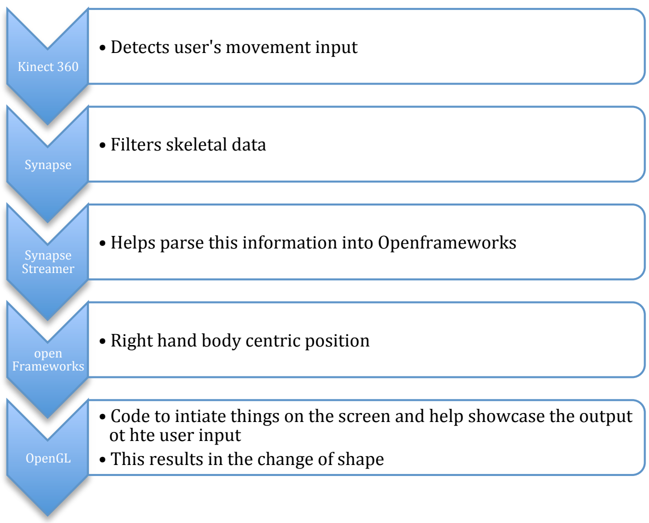

Figure 2 The Kinect 360 detects the users movement, which is then translated into the system in the form of skeletal

NIME'11, 30 May–1 June 2011, Oslo, Norway.

data. Data of the right hand is then extracted from this and using the click to trigger an event the user may move any of the edge points he desires.

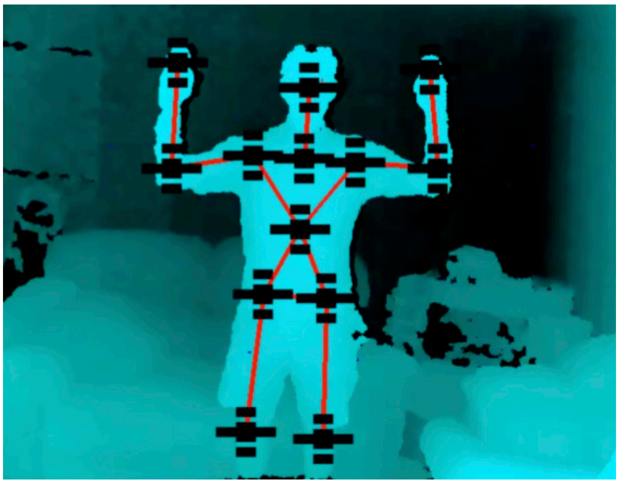

Figure 3 T-pose

Once the data is sent into openFrameworks begins the math. The first step is to convert the coordinates received into body-centered coordinates. This is done because irrespective of the user's physical location he should be able reach out to any of the corners of the drawing window. Once the system gets the body-centered coordinates, the code I have written then translates them into the coordinate system used by OpenGL [10]. OpenGL (Open Graphics Library) is a cross-language, multi-platform application programming interface (API) for rendering 2D and 3D computer graphics. Once this is done, I formulated a pointer out of the information about the right hand that is sent into the program. I declare conditions to find out how close the pointer is to any of the edge points. The user can visually see square markers for each of the edge points. Whenever the user aligns the pointer inside those markers and clicks using the mouse in his hand, the program detects that the click was inside a particular marker and allows to users to move the marker while the mouse button is still pressed. Hence the user performs a grab and move sort of action. The identification as to whether the pointer is inside a certain marker is done using a technique called edge detection. The edge of each marker is detected and it is checked whether the pointer lies inside these edges. If true then when a certain event is triggered (click), the user can move the marker to any desired region on the screen. The users may choose to move round as many markers around as he decides to. When the user releases the mouse button the marker is left at that position.

As far as the use of gestures is concerned I wanted the program to be fairly simple. I mapped the use of gestures using the system to that when the user would be literally drawing on a physical surface. The movement occurs only with respect to one hand, the dominant one. For the purpose of this project I limited that to the right hand. The use of any other gesture of takes it away from the real experience of drawing on a physical surface.

The mouse click could have been initiated with the other arm but I chose to stick with interactions with a single arm. The sense I am trying to invoke is that the user navigates to an edge point, grabs it and then moves it. It also could have easily been a trigger using a keyboard but I wanted the user's dominant hand to be completely mobile and he should be free to interact with the interface the way he interacts with the real physical surface.

Figure 4 My demonstration of editing a Bezier Curve. On the left hand side am I using hand gestures to control the red pointer on the right hand side.

To summarize, there are two key gestures involved in the whole interaction. Firstly, the movement of the hand controls the pointer. This is dictated by the x and y values of the hand in the three dimensional environment. The coordinate in the z-plane is ignored. Secondly, the click and move gesture that moves the markers that dictate the shape of the figure. When the user clicks the mouse when the pointer is inside the marker, the marker then moves to the users desired position till the moue is pressed.

4. Discussion

During the course of the programming and implementation of the system I ran over several issues and interesting challenges. To start of, my earliest vision was to actually use a physical two-dimensional surface like a dry erase. But due to technical constraints I could not do the same. The technical constraint being that I could Synapse does not interpret skeletal data from the back. It skews up the skeleton and hence the coordinates.

Also initially I wanted to let the users draw figures and edit them at their will but due to the timeline of the project this feature was left unimplemented. Had this feature been implemented, this application would have been a complete drawing tool supporting both drawing and editing of the images. Moreover, with the use of sophisticated technology like the Leap Motion it would have been possible to do away with the use of the mouse and use more fine grain gestures like pinching and dragging. Since synapse did not provide the ability to track these minute gestures it was not possible for me to implement the same and I had to rely on the mouse click triggered event for the same. Another approach to remove the mouse from the equation is to use timers in the program. If the pointer is inside a marker for a certain amount of time, the marker can be selected and the user can move the marker as he desires. The one issue with this approach is the fact that after he has attained the desired position for the marker there should be another event to release the pointer from the marker. This would either involved a physical press of some button or involve gestures by using the other hand. I was completely against this hypothesis as I wanted to the user to relate the experience to the real world and hence I had to let go of this approach.

Furthermore, since the pace of the pointer (tracked by the hand) is really fast it sometimes escapes the boundaries of the markers when the user tries to move it. Using a filter that contains the pointer inside the markers can alleviate this problem. Adjusting the frame rate at which the hand position is sampled could solve this problem.

Also some users might consider striking the T-pose as an unnecessary gesture for the user to make. I tried to consider this during the design phase and doing away with the same but unfortunately due to technical constraints it was not possible to do the same. I had to stick to the T-pose gesture as that was the only way that I could use Synapse, which was the only software that I could get to run along with openFrameworks using my limited technological experience in the domain.

Finally another aspect of the program that I would have loved to have some changes implemented in was the way the click and drag feature was implemented. Since my initial thoughts was that the user would navigate to the marker, grab it and then move the same I would have loved to substitute the clicking interaction with something that was more synonymous to grabbing. For instance a stress ball, could be used to give the user a sense of grabbing the marker as he pressed the ball and then moving it around. But due to the lack to technical expertise and the availability of such technology at my disposal I was unable to implement the same.

5. Conclusion and Future Work

In this paper, I described my prototype that helps users to edit the two-dimensional figures that they have drawn using hand gestures and sensing these hand gestures using Kinect 360 sensor.

I have also addressed the various problem and challenges that I encountered during the course of this project. The sensing that I use is limited to the right hand to give the user a realistic feel of interacting with a two-dimensional physical surface. Please feel free to view the demo at

As I was researching further in this domain, I came across various papers on art therapy. This an interesting domain that applications like this can cater to. Drawing is often singled out by art therapists as a common intervention for it is a less threatening way for people to express themselves (Malchiodi, 2006). By drawing, people can enhance self-expression and self-esteem (Rees, 1998). These papers confirm that art therapy is indeed something that rejuvenates individuals. Creating different shapes on a two-dimensional surface though does not reflect art in its true form but if implied and used in the right direction can make a huge impact. Users can tap onto their creative selves to come up with meaningful and beautiful inscriptions that could help them express themselves (Chang Hao Wang, 2012).

Another possible application is again from the field of medical science. Applications like this can be used at rehabilitation centers to help patients make a speedy recovery from their injuries. The patients can try to make improvements on the various figures that they are able to make and change the shape of. Medical practitioners would be able to gauge as to how well a certain patients is doing at any given point of time based on their accuracy at drawing and depicting certain shapes and changing these shapes into something else.

Applications similar to these can be used at various interactive installations to help people interact with the two-dimensional surface without being in contact with it. There can be an amalgamation of the artist's thoughts and the onlookers thoughts to create some interesting visuals in these installations. For instance if the artist had a square in the installation and the onlooker thinks that a circle would be more meaningful in that particular installation, it would be interesting to see the onlooker make the changes he desires and have a look at it. The next person who walks by the installation does not just see the artist's representation but also of all the people who interacted with the installation before the current onlooker. Thus is a way you can say that it would be a interesting tool to explore this domain of collaborative art creating.

Finally, this can be a great tool for education of the young minds in elementary schools. Small children can use this to create and understand different kinds of shapes and teachers and professors can help nurture their skills and know exactly what the level of understanding of the particular child is.

6. ACKNOWLEDGMENTS

I would like to thank Professor Maura Sile O'Modhrain who provided me with guidance on the various design decisions I took and the direction I was headed with the project. I would also like to extend my regards to Conor Barry who helped me set up the programming environment.

7. REFERENCES

1. Baxter, B., Scheib, V., Lin, M.C., and Manocha, D. DAB: Interactive Haptic Painting with 3D Virtual Brushes. *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, ACM (2001), 461–468.
2. Rees, M. (1998). *Drawing on difference: art therapy with people who have learning difficulties*. Psychology Press.
3. Malchiodi, C. A. (2006). *The art therapy sourcebook*. McGraw-Hill Professional.
4. Chang Hao, Wang (2012). A Whole Body, Kinaesthetic Digital Drawing Tool for Art Therapy
5. Bézier curve. (2013, November 23). In *Wikipedia, The Free Encyclopedia*. Retrieved 10:33, December 17, 2013, from http://en.wikipedia.org/w/index.php?title=B%C3%A9zier_curve&oldid=582912160
6. Skeuomorph. (2013, November 7). In *Wikipedia, The Free Encyclopedia*. Retrieved 17:58, December 17, 2013, from http://en.wikipedia.org/w/index.php?title=Skeuomorph&oldid=580666238
7. *openFrameworks*. (n.d.). Retrieved from http://www.openframeworks.cc/
8. *Xbox Home Page | Games and Entertainment | Microsoft - Xbox.com*. (n.d.). Retrieved from http://www.xbox.com/en-US/#fbid=lULnegtG-3l
9. *Synapse Streamer*. (n.d.). Retrieved from http://www.nickgillian.com/software/synapsestreamer/
10. *Modern OpenGL Series Archives « Tom Dalling Tom Dalling*. (n.d.). Retrieved from http://tomdalling.com/blog/modern-opengl/