Chen, W. et al. (Eds.) (2016). Workshop Proceedings of the 24th International Conference on Computers in Education. India: Asia-Pacific Society for Computers in Education

Virtual Slate: Microsoft Kinect Based Text Input Tool to Improve Handwriting of People

Ashwin T S[a!], Kartik SREENIVASAN[a*], Mohammad Akram RAMEEZ[a%], Anmol VARMA[a^], Vikas MOHANDOSS[a$] and G Ram Mohana REDDY[a#]

[a]*National Institute of Technology Karnataka*
Surathkal, Mangalore, Karnataka, India
[!]ashwindixit9@gmail.com

Abstract: Text input is a mundane activity that is very closely associated with Human Computer Interaction. In this paper, using the object tracking facility of the Microsoft Kinect sensor and Tesseract for Optical Character Recognition (OCR), we made it possible to write the text by moving our finger in the air as though we were writing on a virtual slate. One of the main purposes of this proposed work is to help the children so that they can improve their handwriting without somebody to check and monitor their writing activity continuously.

Keywords: Microsoft Kinect, OCR, Teserract, Virtual Slate, Hand-writing.

1. Introduction

We have been writing on pen and paper for a very long time but now with the huge improvement in tools for Human Computer Interaction, we are constantly trying to improve the means by which we can perform these mundane actions of writing. Several methods have been proposed for text input to a computer including variations of existing tools, like various types of keyboards. In our proposed work, we tried to create a *Virtual Slate* by means of which we would be writing in the air, and the device would convert these gestures into the text that the computer can process (Schick et al., 2012). This has been made possible with the help of the Microsoft Kinect Sensor.

As children everyone learns to write on an actual slate using chalk to write. But in the digital age we are looking for a means to replace this old technique. This problem is important because Human Computer Interaction is a constantly evolving field. We propose a new technique by which children can repeat this same procedure but without chalk or any physical surface. The *Virtual Slate* makes it possible for them writes all their text simply by waving their finger in front of a Kinect sensor.

Another aspect of this technique is that it varies the tolerance for conversion of gesture to text as a simulation of *difficulty level*. By varying this the child in question can get continuous feedback as to whether his/her handwriting is improving or not. This would be a completely unbiased opinion as it is purely based on how close to the actual character shape the drawn character is.

The rest of the paper deals with the various stages of the aforementioned procedure. Section 2 discusses the literature survey. Section 3 describes the proposed methodology. Section 4 explains the implementation details followed by results and analysis discussed in Section 5. Finally, Section 6 concludes the paper with the future work.

2. Literature Survey

As mentioned earlier, not much research work has been done in the area of human computer interaction to recognize the input data from the hand gesture in mid air. Alexander Schick et al. (2012) proposed a hand gesture character recognition system in mid air using Hidden Markov Model (HMM) but it was criticized for its low input speed and physical strain.

Oikonomidis et al. (2011) proposed a robust Particle Swarm Optimization based 3D gesture recognition system which recognizes the hand gesture using Kinect sensor and OCR, but they haven't extended their work for the design of any application. Further, for real time scenario they used GPUs for faster hand gesture recognition.

Jin et al. (2013) proposed Kinect based fingertip recognition tool for deletion of the geometric model and their future work on extending their character set to include English and Chinese Characters. Ayshee et al. (2014) proposed fuzzy rule based hand gesture recognition system but it is used to recognize only Bengali characters. Further, this is a static hand gesture character recognition system. Burgbacher U et al. (2016) proposed stroke based virtual keyboard for mobile devices, as the name suggests giving input requires the physical contact of finger with the device.

There are several works from Feng et al. (2013), Chattopadhyay et al. (2008), Chen et al. (2015), MChen (2016) etc., discusses either static or real-time character recognition from hand gesture using sensors but none of these applications uses character recognition as a game based learning tool for improving the users' handwriting. Table 1 summarizes the merits and limitations of existing works. Hence, in this paper, we propose a real-time Kinect based text input tool for improving the users' handwriting.

Table 1. Summary of Existing Works

Authors	Method	Merits	Limitations
Schick et al. (2012)	HMM based Character recognition system.	High character recognition accuracy.	Low input speed and physical strain.
Xiao-Jie et al. (2013)	Kinect based fingertip recognition system.	Used for geometric models.	English characters are not considered.
Tanzila Ferdous Ayshee et al. (2014)	Fuzzy rule based hand gesture recognition.	Accurate character recognition for Bengali characters.	Character recognition is limited only for Bengali characters.
Mingyu Chen (2016)	Air-Writing Recognition.	Modelling and recognition of both characters and words for mid-air character recognition.	This application was not designed for Game Based Learning or improvise their learning style.

3. Proposed Methodology

The proposed methodology consists of hand tracking using Microsoft Kinect sensor (Figure 1), storing the tracked coordinates into bitmap image, then recognizing the character and finally giving the feedback to the user for the improvement of his(her) hand writing (Figure 2).

3.1 Hand Tracking

Object tracking is the main feature of the Microsoft Kinect sensor as shown in Figure 1 and it is able to do this with very high accuracy. When a person is present in front of the Microsoft Kinect, the Skeleton Frame is captured for each person. But in our application we rely on tracking on a single person's hand joints. The minimum distance between the Microsoft Kinect and person to be tracked is 6 feet. The Person who is near to the Microsoft Kinect is taken as the person to be tracked depending on the z- axis value. It is taken as Primary Skeleton and up to twenty joints of the body of the person are tracked. From the Primary Skeleton, we obtained the wrist and hand joint points of the skeleton hand. For our application we retrieve the coordinates of the left and right hand joints (Mori et al., 1992). Using this data, we decide whether the writing is yet to start or ongoing. When the user's left hand is raised then the application assumes that the drawing has begun. Based on the relative z-coordinate of the right hand with respect to the person's head the application begins tracking the x- and y- coordinates of the right hand when it is thrust forward. It is in this stage that the drawing is recorded. When the user's left hand is put back down then the application stores the tracked coordinates in the form a bitmap image.

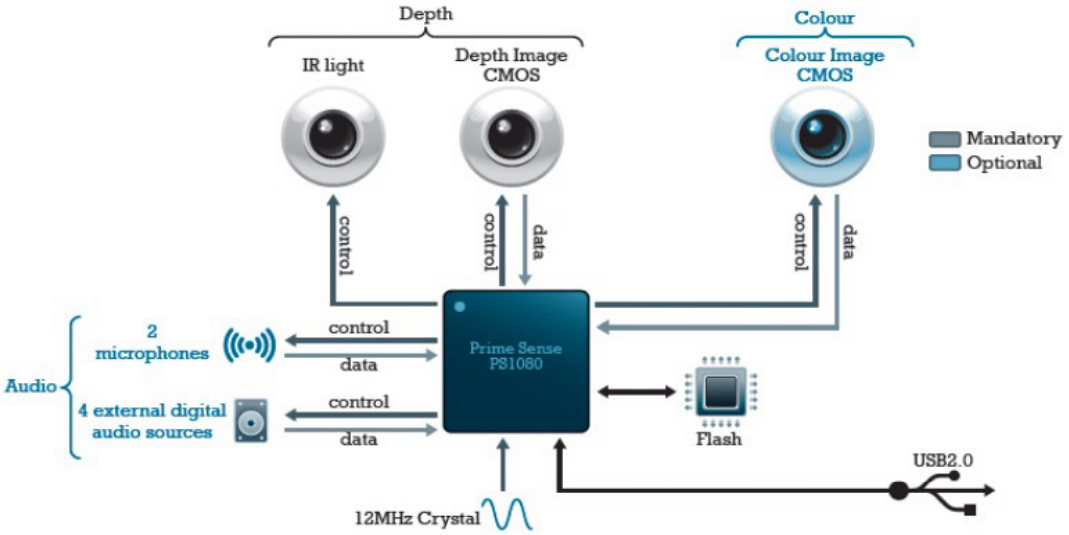

Figure 1. The Kinect Architecture

3.2 Optical Character Recognition (OCR)

Optical Character Recognition (Oikonomidis et al., 2011) is the next step after storing the tracked coordinates. The bitmap image that was generated after the previous stage is passed to the portion of the application that handles OCR. To aid with OCR, we used the open source library Tesseract sponsored by Google. Tesseract is mainly written in C and C++ and hence the APIs to use it in C++ are fairly intuitive. It has proved itself to be the best open source OCR engine at the moment and the accuracy achieved during our application is reasonably high. This is mainly because of its excellent character recognition kernel. The application in its current stage only uses the OCR one character at a time but it can be extended to apply OCR to words or even sentences as a whole at a time. Tesseract is capable of both these features and hence should scale with the application if we add these features.

3.3 Tesseract

Tesseract assumes that its input is a binary image with optional polygonal text regions defined. Processing follows a traditional step-by-step pipeline. The first step is a connected component analysis in which outlines of the components are stored. This was a computationally expensive design decision at the time, but had a significant advantage: by inspection of the nesting of outlines, and the number of child and grandchild outlines, it is simple to detect inverse text and recognize it as easily as black-on-white text. Tesseract was probably the first OCR engine able to handle white-on-black text so trivially. At this stage, outlines are gathered together, purely by nesting, into Blobs. Blobs are organized into text lines, and the lines and regions are analyzed for fixed pitch or proportional text. Text lines are broken into words differently according to the kind of character spacing. Fixed pitch text is chopped immediately by character cells. Proportional text is broken into words using definite spaces and fuzzy spaces. Recognition then proceeds as a two-pass process. In the first pass, an attempt is made to recognize each word in turn. Each word that is satisfactory is passed to an adaptive classifier as training data. The adaptive classifier then gets a chance to more accurately recognize text lower down the page. Since the adaptive classifier may have learned something useful too late to make a contribution near the top of the page, a second pass is run over the page, in which words that were not recognized well enough are recognized again. A final phase resolves fuzzy spaces, and checks alternative hypotheses for the x-height to locate small-cap text (Smith, 2007).

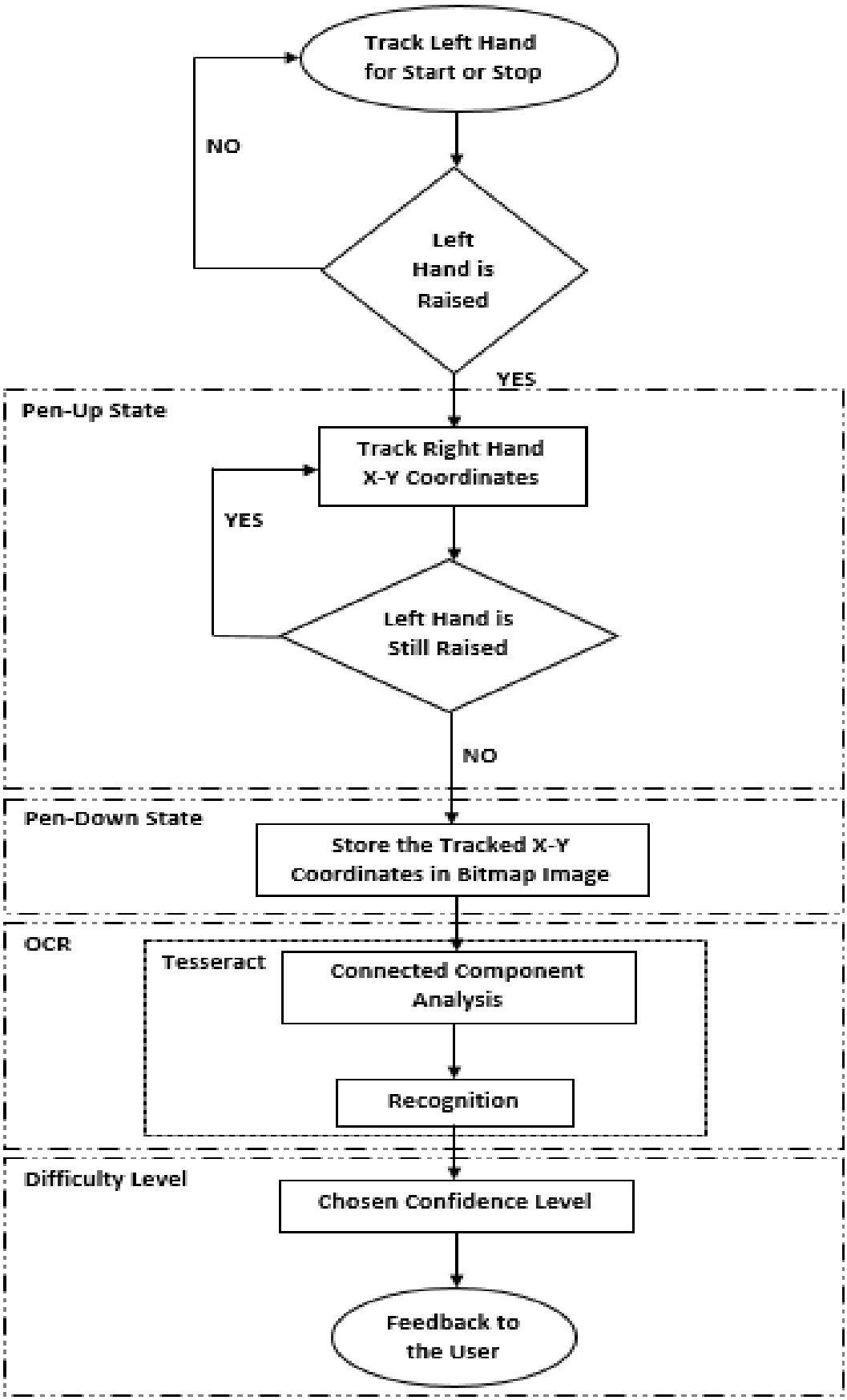

Figure 2. The Proposed *Virtual Slate* Architecture

4. Implementation Details

To implement the *Virtual Slate,* we programmed the Kinect sensor using C++. The platform for the implementation is Microsoft Visual Studio as it offers several features that simplify the process of programming the Kinect sensor. The various stages of the implementation are as follows:

4.1 Object Tracking

Microsoft Kinect does this object tracking with very high accuracy and with very less effort. The APIs to track joints allow us to choose the Left and Right hand joints directly (Frati et al., 2011). Once we have the coordinates of these joints, we simply monitor them continuously for changes that will reflect the User's intentions.

The left hand indicates whether the user wishes to start drawing or not. When the left hand is down in the normal position, we consider this as the start state or the stop state depending on whether the user has been drawing till now or not. Once the left hand is raised, we begin tracking the right hand. The right hand is the equivalent of the user's pen. The left hand being thrust forward indicates that the user wishes to start drawing i.e, *pen-down* state. When he brings his hand back it represents the *pen-up* state. When the left hand is raised, we begin recording the varying coordinates of the left hand and storing them in a bitmap file. When the left hand is put down again the bitmap file is saved (Figure 2).

4.2 Converting the Image to Text

Once the bitmap file is saved we must convert this image to text and for this we must rely on OCR. At this stage, we call the Teserract API and pass the bitmap image to it. Before this stage we need to do some basic image processing in order to make sure that the image is of sufficiently high quality so that Teserract can convert it into text with good accuracy. The API returns the probable character or text that the image represents with the confidence level which it associates with the conversion.

4.3 Difficulty Level

Now we have the probable text form as well as the confidence level of the conversion. We now need to alter the results based on the difficulty level. This portion of the application is purely for the purpose of giving feedback to the user on whether their handwriting is improving or not (Figure 2). Based on the chosen difficulty level, we then alter the results based on varying confidence level. We permit the conversion of the image to text only if it is greater or equal to the minimum required confidence level at that difficulty level. Hence, if the drawing does not get converted to the text then the user receives immediate feedback that their drawing is not close enough to the accepted shape of that character.

5. Result and Analysis

5.1 Experimental Setup

The experiment was set up and tested with more than 60 students with the age group of 4 to 7 years. The system configuration for the proposed experiment is shown in the Table 2.

Table 2: System Configuration Details

System Configuration		
Sl. No.	Hardware	Software
1	Microsoft Kinect	Visual Studio
2	NVIDIA GeForce 630 Graphic Card	Kinect SDK
3	Intel i7 Processor	OCR-Tesseract
4	8 GB Ram	
5	64 bit Operating System	

5.2 Usability analysis

Initially a video of two minutes is shown to the children from which they will learn how to use *virtual slate*. Then they were made to stand in front of the Kinect Sensor which is next to the monitor. This Monitor contains the GUI which shows the recognized character and the difficulty level they have chosen and also it shows the difficulty level they have completed. Further, each set of play contains recognition of 5 particular alphabets with their own chosen Difficulty Level (i.e, Confidence Level (CL) from 1 - 10). There is also a provision of starting the game directly without choosing the difficulty level. Depending on the gesture recognized by the *Virtual Slate*, the difficulty level is automatically selected and displayed. If the highest difficulty level is not attained, then the user can continue and complete the highest difficulty level. Monitor is preferred to the voice acknowledgement since the system proposed by Schick et al. (2012) is criticized for continuous voice feedback for every character recognized, which is annoying.

The Confidence Level (CL) was set for a scale of 1-10 and 10 denotes exact 100% match for that particular alphabet. Table 3 contains the data of those 30 students who successfully completed the confidence level 10 for five particular chosen alphabets. The initial CL number in the Table 3 denotes the CL Value from which they have started and the Table 3 also contains how many attempts they took to reach the highest confidence level.

Table 3: Confidence Level Analysis

User No.	Initial CL No.	No of Attempts to Reach Max CL	User No.	Initial CL No.	No of Attempts to Reach Max CL	User No.	Initial CL No.	No of Attempts to Reach Max CL
1	3	17	11	5	5	21	5	9
2	4	15	12	5	8	22	6	10
3	3	13	13	3	11	23	7	5
4	4	11	14	5	15	24	8	5
5	5	9	15	5	6	25	8	6
6	4	11	16	6	5	26	8	6
7	4	10	17	7	6	27	8	8
8	3	12	18	6	7	28	8	7
9	4	8	19	7	7	29	5	9
10	5	9	20	7	9	30	4	5

According to Alan et al. (2004), in order to make an application successful it should be used (Make people want to use it). In order to check this, we kept our *Virtual Slate* in a classroom for the entire day for 1 week. Whenever students had leisure time they were allowed to come and play. It is observed that most of the students who had played earlier and hadn't completed all the levels for all the English Alphabets, repeatedly played to complete the level. Further, surprisingly those who had completed also played and tested for their consistency for maximum confidence level i.e., 10.

We also collected the feedback from the students either orally or through the written feedback form for Virtual Slate for more than 30 students and the summary is shown in Table 4.

Table 4. Summary of Feedback obtained from more than 30 Students
(5-Point Likert Scale)

Q No.	Items	Value
1	The Virtual Slate functions were easy to use.	4.22
2	The Virtual Slate user interface is friendly.	4.00
3	Whether Virtual Slate keeps students from wandering	4.76
4	Virtual Slate Increases the interaction with the students	4.81
5	Virtual Slate Improves the Handwriting	4.85
6	Overall, I enjoy the Virtual Slate.	4.11
7	Overall, Virtual Slate helps me to learn better.	4.21
8	I would suggest this to my friends to improvise their learning	4.53

We also tested on more than 30 teenagers/adults with an age group of 15 to 55 years, they found it as very interesting tools to check/improve their writing skills. Further, we also collected the feedback from them which is shown in the Table 5.

Table 5. Summary of Feedback Obtained from 30 Teenagers/Adults
(5-Point Likert Scale)

Q No.	Items	Value
1	The Virtual Slate functions were easy to use.	4.79
2	The Virtual Slate user interface is friendly.	4.83
3	Whether Virtual Slate Improves the Handwriting	4.85
4	Overall, I enjoy the Virtual Slate and feel satisfied.	4.22
5	Overall, I think Virtual Slate helps me to learn better.	4.17
6	I would suggest this tools for the adults to improve their handwriting	3.21
7	I would suggest this to the children to improvise their learning	4.71

5.3 Users' Feedback

Even though the Kinect sensor's gesture recognition is accurate, it requires sufficient amount of light in the place where it is kept. Further, since we are using Kinect sensor, we need a computer with an operating system. Hence it fails an application portability. Our work is for character by character recognition, those who completed all the levels successfully for all the cases requested for a sentence or word in cursive writing. Players complaint of physical strain due to continuous use of both the hands.

6. Conclusion and Future Work

Children will be able to practice their handwriting and improve it by getting continuous feedback from the *Virtual Slate* system. Also, by introducing a level of Gamification we introduced a motive for them to keep trying to improve. The *Virtual Slate* can possibly revolutionize the way we give the text input to the computer. For the time being, the accuracy and speed of input make this impractical

and lacking when compared to a traditional keyboard. But with some improvements we may be able to reach a stage where the keyboard is made redundant.

Future work includes the usage of handwriting recognition rather than OCR as it offers a larger dataset to process. Also we could replace the drawing of characters with gestures that represent commonly used words or sentences and we could type these with a single gesture. Further, the gesture based text input could be beneficial to differently-abled users such as those suffering from blindness.

References

Alan, D., Janet, F., Gregory, A., & Russell, B. (2004). Human-computer interaction. *England: Pearson Education Limited*, *5*.

Ayshee, T. F., Raka, S. A., Hasib, Q. R., Hossain, M., & Rahman, R. M. (2014, February). Fuzzy rule-based hand gesture recognition for bengali characters. In *Advance Computing Conference (IACC), 2014 IEEE International* (pp. 484-489). IEEE.

Schick, A., Morlock, D., Amma, C., Schultz, T., & Stiefelhagen, R. (2012, October). Vision-based handwriting recognition for unrestricted text input in mid-air. In Proceedings of the 14th ACM international conference on Multimodal interaction (pp. 217-220). ACM.

Mori, S., Suen, C. Y., & Yamamoto, K. (1992). Historical review of OCR research and development. Proceedings of the IEEE, *80*(7), 1029-1058.

Jin, X. J., Wang, Q. F., Hou, X., & Liu, C. L. (2013, November). Visual gesture character string recognition by classification-based segmentation with stroke deletion. In *2013 2nd IAPR Asian Conference on Pattern Recognition* (pp. 120-124). IEEE.

Burgbacher, U., & Hinrichs, K. Synthetic. (2016) Word Gesture Generation for Stroke-Based Virtual Keyboards.

Feng, K. P., & Yuan, F. (2013, December). Static hand gesture recognition based on HOG characters and support vector machines. In *Instrumentation and Measurement, Sensor Network and Automation (IMSNA), 2013 2nd International Symposium on* (pp. 936-938). IEEE.

Chattopadhyay, T., Biswas, P., Saha, B., & Pal, A. (2008, December). Gesture Based English Character Recognition for Human Machine Interaction in Interactive Set Top Box Using Multi-factor Analysis. In *Computer Vision, Graphics & Image Processing, 2008. ICVGIP'08. Sixth Indian Conference on* (pp. 134-141). IEEE.

Chen, Y., Luo, B., Chen, Y. L., Liang, G., & Wu, X. (2015, December). A real-time dynamic hand gesture recognition system using kinect sensor. In*2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*(pp. 2026-2030). IEEE.

Oikonomidis, I., Kyriazis, N., & Argyros, A. A. (2011, August). Efficient model-based 3D tracking of hand articulations using Kinect. In BmVC (Vol. 1, No. 2, p. 3).

Chen, M., AlRegib, G., & Juang, B. H. (2016). Air-Writing Recognition—Part I: Modeling and Recognition of Characters, Words, and Connecting Motions.*IEEE Transactions on Human-Machine Systems*, *46*(3), 403-413.

Frati, V., & Prattichizzo, D. (2011, June). Using Kinect for hand tracking and rendering in wearable haptics. In World Haptics Conference (WHC), 2011 IEEE (pp. 317-321). IEEE.

Smith, R. (2007). An overview of the Tesseract OCR engine.