

“AIR WRITING”

Idea Lab Report

*Submitted in Partial Fulfillment of the
Requirements for the Degree of*

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

By

**Sagar Maheshwari
17BEC091**

Under Guidance of

Dr. Sachin Gajjar




**Department of Electronics and Communication Engineering,
Institute of Technology,
Nirma University,
Ahmedabad 382 481**

June 2020

DECLARATION

I do hereby declare that the technical project report submitted is original, and is the outcome of the independent investigations/research carried out by me and contains no plagiarism. Efforts have been put to improve on the already available state of the art methods. This work has not been submitted to or supported by any other University or funding agency.

I do hereby, further declare that the text, diagrams or any other material taken from other sources (including but not limited to books, journals, and web) have been acknowledged, referred and cited to the best of my knowledge and understanding.



Sagar Maheshwari
17BEC091

Department of Electronics &
Communication Engineering,
Institute of Technology,
Nirma University,
Ahmedabad 382 481

Dr. Sachin Gajjar
Associate Professor

Department of Electronics &
Communication Engineering,
Institute of Technology,
Nirma University,
Ahmedabad 382 481

Date: 14th June, 2020

Place: Ahmedabad

Nirma University
Institute of Technology
Idea Lab
Electronics and Communication Engineering
Annual Report of the work done on the Idea Lab Project

1. Idea Lab Project ID: IDEA_2019_EC_02
2. Project Title: Continuous Recognition of 3D Space Handwriting using Deep Learning/ Air Writing
3. Period of Project: 1st May, 2019 to 31st March 2020
4. Name of Student: Sagar Maheshwari
5. Name of Mentor: Dr. Sachin Gajjar
6. Total Amount approved: Rs 25000



Sagar Maheshwari
17BEC091
Department of Electronics &
Communication Engineering,
Institute of Technology,
Nirma University,
Ahmedabad 382 481

Dr. Sachin Gajjar
Associate Professor
Department of Electronics &
Communication Engineering,
Institute of Technology,
Nirma University,
Ahmedabad 382 481

Prof. (Dr.) Dhaval Pujara
Head of Department
Department of Electronics &
Communication Engineering,
Institute of Technology,
Nirma University,
Ahmedabad 382 481

Prof. (Dr.) R.N. Patel
Director
Institute of Technology,
Nirma University,
Ahmedabad 382 481

Acknowledgment

This project is the result of my dedication and perseverance towards learning and exploring new fields. An incredible journey, filled with challenges all along, giving a chance to prove myself at every stage.

With great pleasure I deliver my heartiest gratitude and regards to my supervisor Dr. Sachin Gajjar for his at par guidance, noteworthy suggestions and continuous push at all the stages of my research. His attitude and confidence in me were the reason for all the heights I have achieved. I have been fortunate enough to have him as my guide, both as a person and as a professional.

It was a pleasure to connect with Dr. Ruchi Gajjar who helped along the way. To gain help from classmates is always a cherry on the cake. I mark this opportunity to make the most out of what I gained from my peers and will continue doing so in the future.

Lastly, I would like to thank my parents and fellows who constantly appreciated me along the way and encouraged me to give my best.



Sagar Maheshwari

Abstract

I attempt to present novel input methods that helps enable byzantine free of hands interface through recognition of 3D Handwriting. As obsolete as physical input methods are becoming day by day, a new method of input was bound to be brought up. The motion is detected wirelessly by the use of the inertial measurement unit (IMU) of the Arduino 101 board. Two different approaches are discussed to tackle the continuous recognition problem. One approach is to use the Pattern Matching Engine (PME) of the Intel® Curie™ module on Arduino 101 mounted on the back of the hand. Second approach uses the IMU input to a well-structured Recurrent Neural Network. The spotting of handwriting segments is done by a support vector machine. The former approach, being indigent of memory, is not preferred over the latter. The Deep Learning approach can continuously recognize random sentences. The model was trained on 1000 freely definable vocabulary and was tested by only one person, achieving the lowest possible word error rate of 2%.

CONTENTS

Chapter No.	Title	Page No.
	Acknowledgment	i
	Abstract	ii
	Index	iii
	List of Figures	iv
1	Introduction	1
	1.1 Introduction	1
	1.2 Related Work	2
2	Gesture Recognition using Arduino 101	3
	2.1 Deep Learning using Intel Curie	3
3	Gesture Recognition using Deep Learning	7
	3.1 Spotting Stage	7
	3.2 Recognition Stage	8
	3.3 Experiments and Results	9
4	Conclusion	10
	References	11

LIST OF FIGURES

Figure No.	Title	Page No.
1	Prototype of gesture recognition glove	3
2	Outline of the data processing pipe	4
3	Drawing “A”, raw accelerometer data	5
4	Under sampled version of ‘A’	5
5	SVM architecture	7
6	RNN network structure	9

Chapter 1

Introduction

1.1 Introduction

Hand gestures are a pervasive, common and significant piece of a communicated language. The advent of recognition of 3D hand gestures has enticed variety of research concerns in various emerging fields namely pattern recognition, computer vision and human-computer interaction. There are several ways of sensing gestures, one of which is a low-cost method of sensing through hand mounted sensors that include accelerometers and gyroscopes [1].

Past research concentrates more on detection of a very exhaustive list of gestures that are mapped to corresponding commands beforehand. This sets a limit on the number of possible commands. Operations like writing or comprehending text or some different convoluted tasks entail more expressive capacity than a limited bunch of secluded gestures [2].

This paper presents novel approaches that combine the intuition gathered from gestures to express it in the form of handwriting, specifically as a text output. The approaches are for wearable motion detection techniques dependent on inertial sensors and use of Deep Learning, allowing us to spot and recognize continuous stream of words written in the free space.

There are several challenges that arise during the operation. First, in everyday life, the gestures are not limited to specific handwriting segments, but also include the normal day-to-day activities, introducing a lot of irrelevance in the text input interface. The handwriting segments should be identified beforehand in the continuous stream of data. Secondly, as the accelerometer data is noisy, it should be filtered before sending it to recognition stage. Third, the actual text input must be recognized in the whole data stream.

For the continuous recognition, we use two approaches. The first approach involves the use of Arduino 101 [3]. The Intel® Curie™ module embedded on the Arduino 101 provides us a pattern matching engine that can be used to recognize the gestures.

The second approach is to use the 3-axes accelerometer of Arduino 101 and divide the process into 2 stages. The first stage is the spotting stage, which involves the use of a

support vector machine [4] to classify between the writing and non-writing segments. The second stage uses Recurrent Neural Networks for recognition of the gestures [5]. The two stages can be evaluated on different datasets independently but combining them gives a significant boost in performance after the further filtering of untrue positive sections.

While the existing proposed scheme is based on recognition of text, this can be utilized as further proof-of-concept for any kind of gesture recognition scheme which is built on a primeval alphabet of freely definable gestures. The first approach lacks suitable memory for large datasets i.e. it is limited to only 128 bytes of memory per neuron for 128 neurons. The second approach, however can be applied to large definable vocabularies, larger than V1K.

1.2 Related Work

Recent research suggests the paradigm shift to mobile device computing by facilitating hands-free action. Gestures allow to foster interface that is independent of any handheld tool. Hence, allowing faultless incorporation into day to day activities.

Mini-projectors portray the screen on a rigid surface in front of the user and the gesture is tracked via a camera or any other medium [6]. However, the approach depends on a sensory input and hence would perform poorly in case of continuous stream recognition. Other researchers propose that 3-D communication is doable without any sort of graphical output. The operator needs to imagine a blank surface which serves the purpose of the screen [7]. Handwriting can be predicted as text lacking any optical or sensory feedback, a method which is used here.

In any accelerometer data, spotting of relevant signal segments is necessary. This is possible by employing a binary classifier to detect probable segments and then classify the gesture afterwards [7]. This approach, however introduces a latency and therefore, the overhead involved reduces the efficiency of the recognition system. The other method is to sort the input constantly and eliminate any irrelevant outputs.

Gesture recognition using accelerometer data has been studied heavily in the past where normally a number of secluded gestures are expressed and sorted [8]. Many researchers propose variety of methods to recognize gestures through accelerometer input. This paper discusses two approaches to recognize gestures using accelerometer data.

Chapter 2

Gesture Recognition using Arduino 101

Arduino 101 is a development board used for learning purposes which comprises of Intel® Curie™ Module, comprehensively designed to assimilate the core's low power usage and elevated ease-of-use [3]. The Arduino is packed with Bluetooth Low Energy capabilities and has an onboard 6-axes accelerometer/gyroscope. The module consists of 2 miniature cores, an x86 (Quark) and a 32-bit ARC architecture core, both of which are clocked at 32 MHz. The Real-Time Operating Systems (RTOS) and framework designed by Intel are both open-sourced.

2.1 Deep Learning using Intel® Curie™

The Pattern Matching Engine (PME) is a parallel data recognition engine having 128 parallel processing elements (PE) each along with 128-byte input vector, 128-byte model memory and 8-bit arithmetic units. It also has two distance evaluation norms with 16-bit resolution: L1 norm (Manhattan distance) and Lsup norm (Chebyshev distance).

It supports 2 classification techniques: Radial Basis Function (RBF) and k-nearest neighbors (KNN) and supports 127 contexts. Arduino 101 provides CuriePME API that can be used to train and classify gestures. It provides us with learn and classify functions that are essential to recognize gestures.

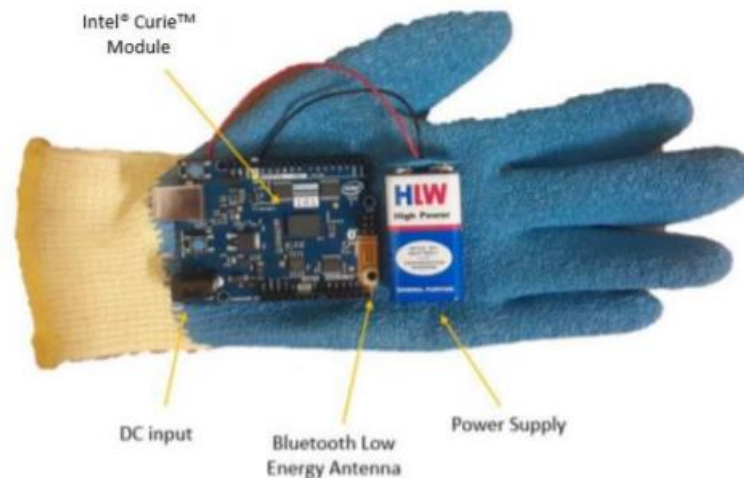


Figure 1: Prototype of gesture recognition glove

Additionally, the module also provides with an inertial measurement unit having 6 degrees of freedom, and each sensor sample (s_o) can be represented as a 6-dimensional vector corresponding accelerometer and gyroscope values:

$$s_o = (\mathbf{a}, \mathbf{g}) = ((a_x, a_y, a_z), (g_x, g_y, g_z)) \dots\dots\dots (1)$$

As stated earlier, the QuarkSE core on Curie module comes with 128 neurons, with 128 bytes of memory per neuron. And hence, there is a trade-off between memory and the data that can be classified. We propose a system shown in Fig. 2 which is user dependent and gives a comparatively poor performance in terms of Word Error Rate on person-independent setup. This system gives better performance when the dataset comprises of utmost 3 syllable words. The system is known to give 100% accuracy when we have single letters to be classified. Continuous recognition of words is also possible with this setup but is not recommended due to memory indigence.

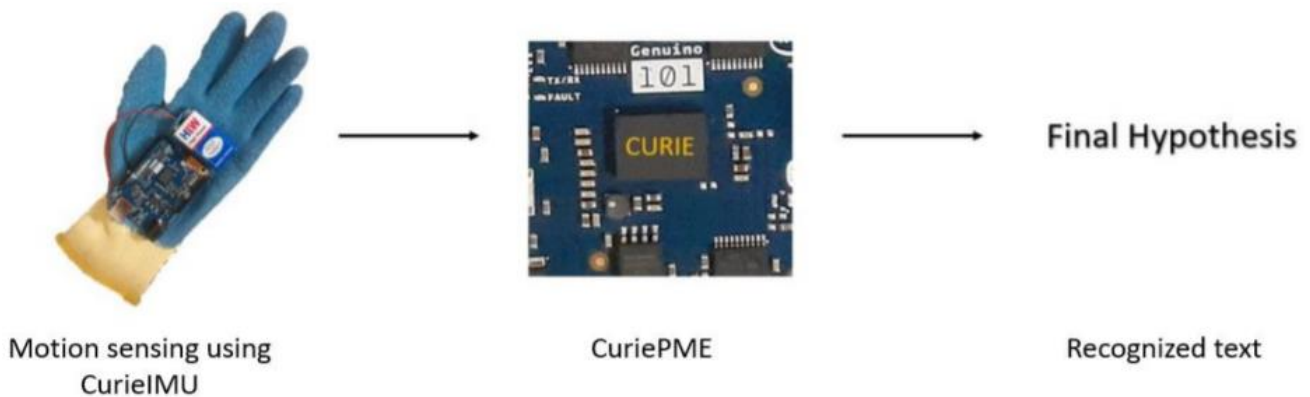


Figure 2: Outline of the data processing pipe. After the accelerometer data is acquired by Curie inertial measurement unit, the Curie Pattern Matching Engine will try to classify the data and hence provide a Final Hypothesis

For instance, drawing the letter A, takes almost 2 seconds, which is 200 samples at 100 Hz. Now, the accelerometer values correspond to 3 axes values in ‘int’ which are 4 bytes each. So, this makes up 2400 bytes per letter. But due to memory constraints, our pattern can be no larger than 128 bytes. So, to throw off at least 95% of data without affecting the results require the use of under sampling. With the use of under sampling, we can achieve the max size of 128 bytes per letter. Also, the data can be noisy. To remove the

noise, we use an averaging filter. From the above discussion, it is clear that the memory management is not efficient with this system. And which leads to a lot of data being wasted. The CuriePME library is mostly used for (1) learning patterns (2) classifying and recognizing patterns (3) storing and retrieval of pattern matching as knowledge. The use of CurieBLE library ensures wireless function of the glove [10].

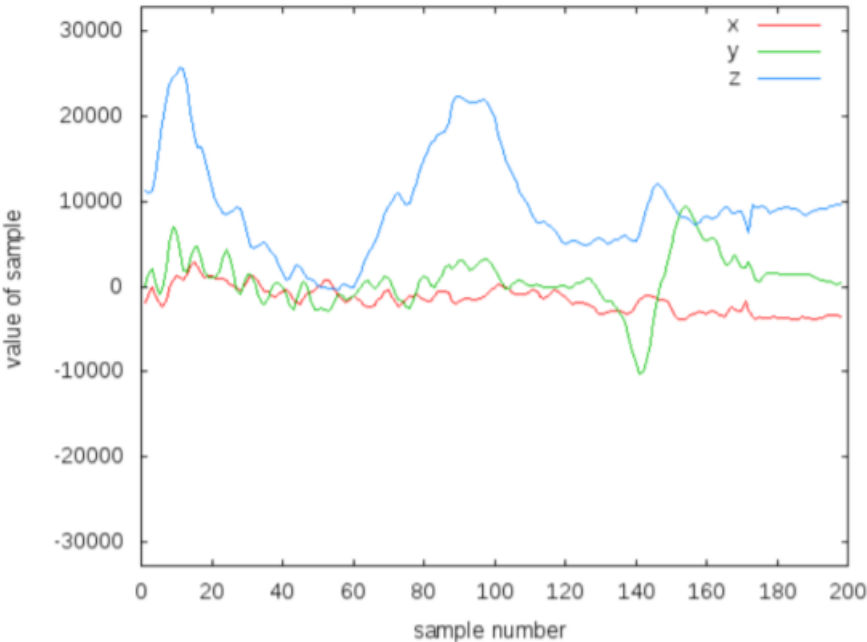


Figure 3: Drawing "A", raw accelerometer data

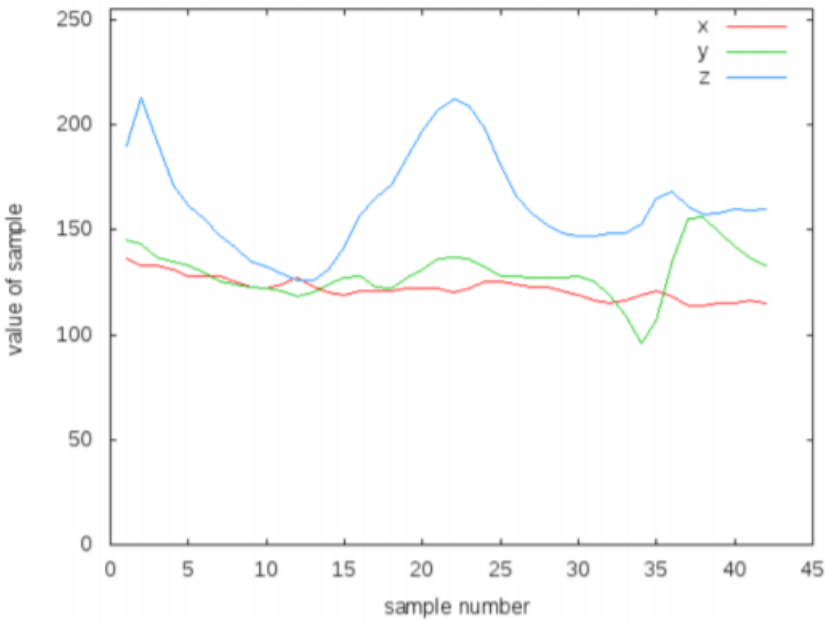


Figure 4: Smoothened accelerometer data

This is also the result of using averaging filter on the entire accelerometer sequence. Due to the memory constraints and the lack of efficient memory management, we propose a new method with the use of deep learning for gesture recognition that gives a better accuracy in response to large datasets and longer streams of input.

Chapter 3

Gesture Recognition using Deep Learning

This method using deep learning provides a more robust approach in recognition of gestures. This approach can be divided into two stages, namely spotting stage and the recognition stage. The combination of the two stages introduces no overhead and therefore, there is no effect on the accuracy of word detection. The process can be seemingly pipelined and real-time detection of gestures is possible.

3.1 Spotting Stage

It is very important to differentiate between the writing and non-writing segments. And this is the role of the spotting stage. It uniquely identifies the segments in the accelerometer data stream which correspond to a word in the sentence. The intuition of the spotting stage is derived by C. Amma et al. [4].

The segments that are correctly recognized as the writing segments are then carried forward to the detection stage. In ideal cases, the spotting procedure should identify all the

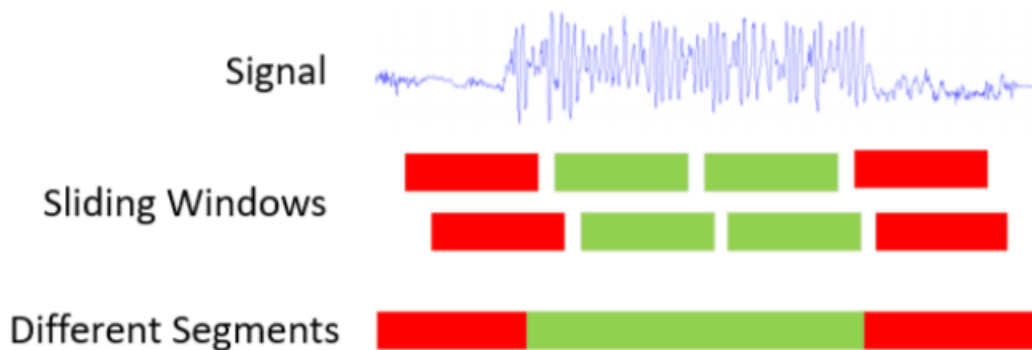


Figure 5: SVM architecture, red and green segments show nonwriting and writing segments respectively

writing segments without producing untrue positives and imposing absolutely no delay. The spotting stage uses a binary support vector machine (SVM) classifier with an RBF kernel ($\gamma=2$, $C=126$) to differentiate probable writing signal sections from non-writing signal. For usage on continuous data streams and in real time, approach of sliding window is more suitable. The overlapping sliding windows are classified and accumulated to send to the recognition stage. The window of length 0.9s and shifting width of 0.1s is used in the approach. Fig. 5 depicts the architecture of the spotting stage. Visual inspection depicts that

the handwriting part has high frequency and amplitude than the non-writing part. For each window w_t , the SVM Classifier $C(w_t)$, returns 0 when no handwriting segment is detected and returns 1 otherwise. One sample of sensor, s_t is classified as a handwriting motion if at least one window consisting of s_t is classified as handwriting motion [4].

This system is biased towards the detection of writing motion. Also, minute

$$C(s_t) = \max_{k: s(t) \in w(k)} C(w_k) \quad (2)$$

intervals during writing will not lead to gaps in the detected writing segments. The real time experiment results show that the chosen values are suitable for the model.

As the system is biased, a high recall of 98.2% is attained in the process and the low precision of 32% is attained. As a comparable result in [4], these values are reasonable.

3.2 Recognition Stage

The purpose of Gesture detection is to build a dominant classifier and hence several deep learning models that exhibit temporal dynamic behavior come into play. These state-of-art models include Recurrent Neural Networks (RNN) [5], Long Short-Term Memory (LSTM) [5] and GRU (Gated Recurrent Units) [5]. We discuss Recurrent Neural Network approach to recognize gestures as text. Recurrent Neural Networks are utilized to process time sequence data. In a conventional neural network model, from the input layer to the output layer, the layers are fully connected. However, this kind of neural network is not suitable for timeseries data. Hence in RNN, the present output of the sequence is also related to the past output. The network would remember the previous output and apply this information for calculating the current output. In theory however, RNN can manage infinite time-series data. In practice, to reduce intricacy, the current state is only related to the previous few states, according to need [5]. The following features are extracted from the accelerometer data extracted from inertial measurement unit of Arduino 101.

- Averaged 3D acceleration
- Averaged 3D Angular rate

Fig. 6 shows the RNN network structure that is used.

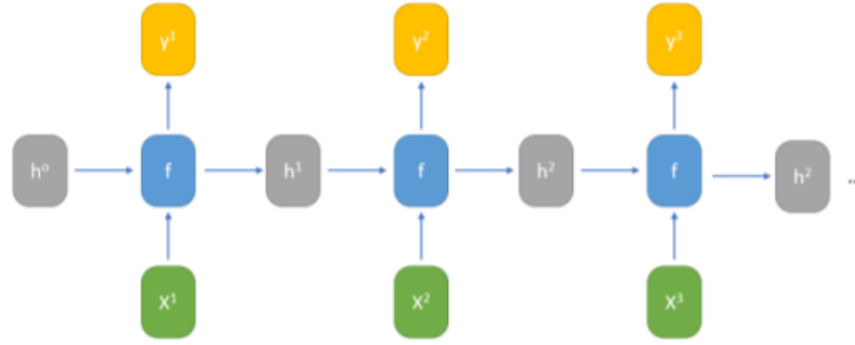


Figure 6: RNN network structure

The formula for the network can be derived by the following equation [5]:

$$h^t = f(w^h h^{t-1} + w^i x^t) \quad (3)$$

$$y^t = f(w^o h^t) \quad (4)$$

x_t depicts the input at $t=1, 2, 3, 4, \dots$ h

t is the hidden layer at step t , (network's memory unit)

y_t is output of step t

And f , is usually a non-linear activation function such as ReLU or Leaky ReLU or tanh.

3.3 Experiments and Results

The experiment was conducted by a single subject. The length of sentences varied from 2 to 4 words. The user had to write 10 English sentences without moving the wrist with an approximate height of 15 cm per character. In total, the user wrote 37 words with 245 characters. The Neural Network took half an hour to train on a small vocabulary (V1k) that contains 986 words on NVIDIA GeForce GTX 1050Ti laptop [10]. The Word Error Rate was calculated as depicted in the C. Amma et al. [4]. A word error rate of 2 % was achieved.

Chapter 4

Conclusion

A wearable method of gesture input is suggested that is adept in writing and recognizing text input written in air centered on inertial measurement unit of Arduino 101. It was observed that while using Arduino 101 for gesture recognition using pattern matching engine of Intel® Curie™ module, the system works well on detection of gestures containing words with utmost 3 syllables and works with 100% efficiency when single syllable is input. However, the data that can be trained is not flexible and wide, because the system is widely indigent of memory and more memory is required to widen the data set. To tackle with the memory requirements of the first approach, we suggested a second method which makes use of deep learning for gesture recognition task. It was observed that during spotting stage, 98% recall and 32% precision was achieved which works in accordance with the system. Training of the Neural Network was conducted on a very small vocabulary (V1K). Experiments were conducted on a very small dataset of approximately 300 words. Dependent on the subject, the Word Error rate of 2% was achieved. This proposed architecture is assumed to perform better on versatile dataset of large vocabulary (V8K and above).

References

- [1] H. Cheng, L. Yang and Z. Liu, "Survey on 3D Hand Gesture Recognition," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 9, pp. 1659-1673, Sept. 2016.
- [2] C. Amma and T. Schultz, "Airwriting: Demonstrating Mobile Text Input by 3d-Space Handwriting," in *Proc. of the ACM international conference on Intelligent User Interfaces (IUI'12)*, 2012.
- [3] "Arduino - Arduino101", *Arduino.cc*, 2020. [Online]. Available: <https://www.arduino.cc/en/guide/arduino101>. [Accessed: 05- Apr- 2020].
- [4] C. Amma, M. Georgi and T. Schultz, "Airwriting: Hands-Free Mobile Text Input by Spotting and Continuous Recognition of 3d-Space Handwriting with Inertial Sensors," *2012 16th International Symposium on Wearable Computers*, Newcastle, 2012, pp. 52-59.
- [5] T. Du, X. Ren and H. Li, "Gesture recognition method based on deep learning," *2018 33rd Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, Nanjing, 2018, pp. 782-787.
- [6] F. Chen *et al.*, "WristCam: A Wearable Sensor for Hand Trajectory Gesture Recognition and Intelligent Human–Robot Interaction," in *IEEE Sensors Journal*, vol. 19, no. 19, pp. 8441-8451, 1 Oct.1, 2019.
- [7] S. Gustafson, D. Bierwirth, and P. Baudisch, "Imaginary interfaces: spatial interaction with empty hands and without visual feedback," in *Proc. of the 23rd annual ACM symposium on User interface software and technology (UIST'10)*, 2010.
- [8] M. Elmezain, A. Al-Hamadi and B. Michaelis, "Hand trajectory-based gesture spotting and recognition using HMM," *2009 16th IEEE International Conference on Image Processing (ICIP)*, Cairo, 2009, pp. 3577-3580.
- [9] 74F. Zhan, "Hand Gesture Recognition with Convolution Neural Networks," *2019 IEEE 20th International Conference on Information Reuse and Integration for Data Science (IRI)*, Los Angeles, CA, USA, 2019, pp. 295-298.
- [10] Support for Intel® Curie™ Modules", *Intel*, 2020. [Online]. Available: <https://www.intel.com/content/www/us/en/support/products/94036/boards-and-kits/intel-curie-modules.html>. [Accessed: 05- Apr- 2020].

[11] "GeForce GTX 1050 Ti | Specifications | GeForce", *Geforce.com*, 2020. [Online]. Available: <https://www.geforce.com/hardware/desktop-gpus/geforce-gtx-1050-ti/specifications>. [Accessed: 05- Apr- 2020].