Hand Gesture Recognition

Detailed Project Plan

Name: Dhruv Agrawal

UID: 3035344405

Supervisor: Dr. K.K.Y. Wong

Course Code and Title: COMP4801 Final Year Project

Date: 14 Sep. 2019

Project Website: Hand Gesture Recognition Website

1.0 Project Background

1.1 Motivation

Both Hand Pose Estimation and Hand Gesture Recognition have a large number of real world applications. Some of the most popular ones include Human Computer Interaction, Human Action Recognition and Sign Language Translation. Wachs *et al. [1]* introduces a number of use cases for vision based hand gesture applications. These include Lexicon Design and, Medical Systems and Assistive Technologies. Therefore, the need for such a technology is quite evident. However, the present state of the art performers in the field cannot satisfactorily meet the requirements for viable real world use. Wachs *et al. [1]* also mentions a number of challenges such as responsiveness, user adaptability and ,most importantly, accuracy. Therefore, there is need for research into the field in order to gain insight on how this technology can be furthered.

1.2 Related Works

Hand Gesture Recognition is one of the difficult problems in Computer Vision and Deep Learning. Unlike Body Pose Estimation or Action Recognition, Hand Gesture Recognition suffers from problems of high degree self-occlusions and a high freedom of movement for each joint in the hand. This necessitates use of sophisticated architectures. For example, Narayana *et al [2]* use as 12 data channels and measure optical flow in their proposed architecture. Such implementations introduce an execution penalty and render such applications not ideal for real time applications. Since most of the proposed use cases of such technology require a real time gesture sensing system, such a problem remains largely unsolved. Attempts at real time gesture recognitions have been made still. Köpüklü *et al. [3]* try to alleviate the problem by dividing their architecture into a detector and a classifier. The detector

is a light weight model that only detects when a gesture has occurred in the input video. When a gesture is detected, only then the heavy weight classifier is activated to detect the said gesture. This method shows promising results however, splitting the process into two different sub architectures less than desirable. Since an end-to-end learning solution will introduce fewer points of failure, it is more suitable for wide applications.

In this project, I attempt to build a light weight model that can carry out hand gesture recognition from an input stream.

2.0 Project Objective

The objectives of my project are two fold. Firstly, most of the current industrial implementations of Hand gesture Recognition use an array of sensors as in the Kinect from Microsoft. Even though, the Kinect performs satisfactorily for Body Pose estimation and for real time gaming, its use in Hand pose estimation is still being researched. The Kinect is a bulky device due to the multiple sensors it houses. This decreases its portability. Therefore, the first objective of my project is to make a Hand Gesture Recognition system that uses an input from a single RBG-D sensor.

Secondly, most of the current day models are very complex. This complexity introduces an execution penalty that makes most of the systems undesirable for real time use. Therefore, the second objective of this project will be to gather evidence on the feasibility of solving the Hand Gesture Recognition problem with lightweight architecture suitable for real time usage.

3.0 Project Deliverables

The project deliverables will include an API implementing the different structures of the Hand Gesture Recognition. Furthermore, the project will also deliver a GUI implementation of the beforementioned library. More details about the deliverables can be found below:

3.1 HandNet and GesReg APIs:

The APIs will have the implementations of the final structures of both the Hand Pose Estimation models and the Hand Gesture Recognition models respectively. The APIs will furthermore include the required classes for data loading and preprocessing. Lastly, the API will implement classes for quantitative and qualitative analysis and training and validation classes tailored to the implementations of the aforementioned data loading classes.

3.2 GUI application:

The second deliverable of the project will provide a user-interface for interacting with the API discussed in section 3.1. The application will allow the users to input an image or a video. The image will be used for Hand Pose Estimation whereas the video will be used for hand Gesture Recognition. The users will be able to view the results and also save annotated images highlighting the pose in the images.

3.3 Prototypes:

Besides the final deliverables, there will also two interim deliverables of the aforementioned API . These deliverables will be incremental upgrades of each other. The first interim prototype, called GesReg 1, will be the first implementation of the Hand Gesture Recognition system. This system will feature the data preprocessing

conducted using the Hand Pose estimator. This prototype should be complete by the end of November. The second prototype, similarly named GesReg 2, will be a structural upgrade from GesReg 1. It will feature changes in its data processing and output stages to provide a better performance from GesReg 1. This prototype should be ready by mid January 2020.

4.0 Project Methodology

4.1 Datasets Used

For the training of the various models in this project a number of datasets will be required. Currently, I have selected the following datasets:

1. NYU Hand Pose dataset [4]: This dataset contains 8252 test-set and 72757 training-set frames of captured RGBD data with ground-truth hand-pose information. All the images comprise of various hand poses. For each hand pose, Kinect data from three different angles is captured. Finally, this dataset is presently popular among Hand Pose researchers due to the high variability of Hand Poses captured in this dataset.

2. ICVL Hand Pose dataset [5]: This dataset annotates 16 joint locations with (x,y,z). Coordinates available for each image. The x and y coordinates are measured in pixels while z coordinate is measured mm.

3. MSRA Hand Pose dataset [6]: This dataset contains images from 9 subjects' right hands are captured using Intel's Creative Interactive Gesture Camera. Each subject has 17 gestures captured and there are about 500 frames for each gesture.

4. Multiview Hand Pose Dataset [7]: This dataset captures hand pose from different angles. This dataset not only provides the 3D hand joint locations for each image but also provides the bounding boxes for the hands in the images.

5. EgoGesture [8][9]: This dataset contains 2,081 RGB-D videos, 24,161 gesture samples and 2,953,224 frames from 50 distinct subjects. The authors define 83 classes of static or dynamic gestures focused on interaction with wearable devices.

6. NVIDIA Dynamic Hand Gesture Dataset [10] : nvGesture is a dataset of 25 gesture classes, each intended for human-computer interfaces. The dataset has 1532 weakly segmented videos, which are performed by 20 subjects at an indoor car simulator. The dataset is then split into training (70%) and test (30%) sets, resulting in 1050 training and 482 test videos [11].

Currently, I have been focused on the Hand Pose Estimation Problem. And hence, the majority of the datasets listed pertain to hand pose recognition. However, with the progress of the project other datasets focusing gesture recognition will also be added to the list.

4.2 Development Details : Division of Work into two phases

The project has been primarily spilt into two phases. The first half of the project focuses on Hand Pose Estimation and the second half of the project focuses on Hand Gesture recognition. Currently, the project is in Phase I. The details of the two phases are as follows:

4.2.1 Phase 1: Hand Pose Estimation Problem

For this phase, I am attempting to create a light weight hand pose estimating model. To achieve this, I have spent a majority of the time during the summer internship reading research papers regarding the same problem. After gaining suitable knowledge about the state of the art performance on the subject, I came up with my own insight into the problem. This being, the movements of the hand joints is primarily hierarchical and we can use this fact to refine the estimations made by the model.

After this I designed my first model and I have been refining its design ever since. The implementation details of these models are as follows:

4.2.1.1 Implementation Details: Using branched architecture

The salient features that are common to the designs over all designs include a branching system specializing in regressing the positions of a subset of the complete joint set. The architecture consists of three branches. The first branch is responsible for regressing the position of the palm joint as well as the finger bases. The second branch then uses the positions predicted by the first branch in addition to its own feature extraction layer in order to detect the center joints of each finger. Note that this layer also acts as a refining layer for the joints detected in the first stage. This is achieved by adding the set of joints predicted by the first layer to set of joints predicted by the second layer. Lastly, the third layer computes the position of the fingers tips using the output of the second layer in addition to its own feature extraction layer. Also note that similar to the second layer, the third layer also acts as a refining layer for the joints predicted by the preceding two layers. Hence, the output of the third layer is the complete set of joints in the hand.

Currently, I am on the sixth iteration of my design. For this iteration, I am combining the initial feature extraction stage for the three sub – networks. This design allows me to significantly reduce the number of parameters that need to be trained. This

also as an effect of significantly lowering the execution time of the model. Other than the sixth iteration, I also have design ideas for future iterations. One of the main ideas is to embed a generic hand model into the architecture itself. This change is design allows to restrict the output latent space from purely 3D to 2.5D. This decrease in the dimension of the output space, should help with the output accuracy.

4.2.2 Phase 2: Hand Gesture Recognition Problem

After completing the hand pose estimation problem satisfactorily, I plan to build upon that model for hand gesture recognition. I am currently searching for state of the art research material regarding the subject and different architectures employed for the task. My current plan is to use a RNN based model that would use my model from phase 1 as a data preprocessor. This phase will be elaborated once phase 1 is complete and I can start working on the corresponding problem for this phase.

As mentioned earlier, the project is currently in phase 1. I should be able to complete phase 1 by mid October. After that, I will initially focus on elaborating the details of the second phase and then move in development of the hand Gesture recognition system.

4.3 Challenges

I do realize that there are a number of challenges for this project. Chief amongst them being:

1. Finding a complete hand gesture recognition dataset: Even though a number of hand gesture datasets exist, each of them have their own shortcomings. Some of them have a very small number of unique gestures while some of them have poor lighting conditions. Another big shortcoming for most datasets is the low resolutions of the videos due to the file size restrictions.

2. Making a satisfactory Hand Pose Estimator: Since my goal for Hand Pose estimator is to keep it as light weight as possible, I need to find a balance between accuracy and execution speed.

Currently, I am still working on fixating other important parts related to methodology such as Hardware and Software requirements and Quantitative Analysis Criteria. These details will be added in future reports once available.

5 Project Schedule and Milestones

5.1 Tentative Project Schedule:

This table summarizes the different periods in the project and the main objectives of each period. Each period is in between 4 and 6 weeks in length.

1 Sep 2019 – 20 Oct 2019	Complete Hand Pose Estimation models.
21 Oct 2019 – 01 Dec 2019	Working on first prototype for the Hand gesture recognition system. Also work on input stream and output stream implementations.
24 Dec 2019 – 15 Jan 2020	Work on the second prototype of the Hand gesture recognition system.
16 Jan 2020 – 14 Feb 2020	Work on the final implementation of the Hand gesture recognition system.
15 Feb 2020 – 31 Mar 2020	Testing and Debugging
1 Apr 2020 – 18 Apr 2020	Real world testing and debugging

5.2 Milestones and Tentative Completion Dates:

29 Sep 2019	Submission of Inception Deliverables.
20 Oct 2019	Phase 1 complete. A working hand pose estimator ready.
30 Nov 2019	First hand gesture recognition prototype ready.
15 Jan 2020	Second hand gesture recognition prototype ready.
2 Feb 2020	Submission of Elaboration Deliverables.
14 Feb 2020	Third hand gesture recognition model ready.
19 Apr 2020	Submission of Final Deliverables.

6 References:

1) Wachs, J. P., Kölsch, M., Stern, H., & Edan, Y. (2011). Vision-based hand-gesture applications. Communications of the ACM, 54(2), 60-71.
2) Narayana, P., Beveridge, R., & Draper, B. A. (2018). Gesture recognition: Focus on the hands. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5235-5244).
3) Köpüklü, O., Gunduz, A., Kose, N., & Rigoll, G. (2019). Real-time hand gesture detection and classification using convolutional neural networks. arXiv preprint arXiv:1901.10323.
4) Tompson, J., Stein, M., Lecun, Y., & Perlin, K. (2014). Real-time continuous pose recovery of human hands using convolutional networks. ACM Transactions on Graphics (ToG), 33(5), 169.
5) https://labicvl.github.io/hand.html

6) Sun, X., Wei, Y., Liang, S., Tang, X., & Sun, J. (2015). Cascaded hand pose regression. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 824-832).
7) Gomez-Donoso, F., Orts-Escolano, S., & Cazorla, M. (2019). Large-scale multiview 3D hand pose dataset. Image and Vision Computing, 81, 25-33.
8) Zhang, Y., Cao, C., Cheng, J., & Lu, H. (2018). Egogesture: a new dataset and benchmark for egocentric hand gesture recognition. IEEE Transactions on Multimedia, 20(5), 1038-1050.
9) Cao, C., Zhang, Y., Wu, Y., Lu, H., & Cheng, J. (2017). Egocentric gesture recognition using recurrent 3d convolutional neural networks with spatiotemporal transformer modules. In Proceedings of the IEEE International Conference on Computer Vision (pp. 3763-3771).
10) Gupta, P. M. X. Y. S., & Kautz, K. K. S. T. J. (2016). Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural networks. CVPR.
11) Kopuklu, O., Kose, N., & Rigoll, G. (2018). Motion fused frames: Data level fusion strategy for hand gesture recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 2103-2111).