HAND SIGNALS WITH VISUAL DATA

CAPSTONE PROPOSAL

[CSC 59866 – DD1]
SENIOR DESIGN 1

BY
AYUSHYA AMITABH
CHRIS PANICAN
GERRY XU
OMAR ELNAGDY

ABSTRACT

This paper presents the concepts behind the development of a proposed software – “Hand Signals with Visual Data”. Furthermore, this paper discusses the use, demand, and challenges behind the development of such a software. This paper presents the full feature specification as well as the technologies that will be utilized in order to develop a software that would utilize on-device deep learning to recognize hand gestures and trigger commands based on the combination of gestures and finger count. The paper also presents the ideals behind deep learning, its utility in our modern world and also, the role it plays in the development of the proposed software.

Team Contacts

Ayushya Amitabh ..... aamitab000@citymail.cuny.edu

Chris Panican ..... cpanica000@citymail.cuny.edu

Gerry Xu ..... gxu000@citymail.cuny.edu

Omar Elnagdy ..... oelnagd000@citymail.cuny.edu

Keywords

Deep Learning, OpenPose, OpenCV, Caffe, Gesture Recognition, Convolutional Neural Network, Recursive Neural Network

THIS PAGE LEFT INTENTIONALLY BLANK

TABLE OF CONTENTS

INTRODUCTION

"Hand Signals with Visual Data" is the title of our proposed project that will work to create a software to allow for gesture based controls on the end user's computer with the aid of deep learning to help recognize the user's hand as well as the number of fingers being held up by the user.

The Need for Hand Gestures

The motivation behind *"Hand Signals with Visual Data"* is split between fantasy and the urge to push technology to new limits. Like self-driving cars, speech recognition, and many other dreams that Artificial Intelligence has helped us achieve – we believe that Deep Learning (a sub-field of Artificial Intelligence) can help us achieve this fantasy too. The use of hand gestures has been a common place in many futuristic movies – a thing of fantasy that only CGI could achieve. This, however, is no longer true – with the recent developments in AI we can now works towards realizing the dream of controlling our computer with effortless hand gestures without having to lay hands on the computer itself.

Throughout this paper we will explore the basics of Deep Learning, how we will use it, the specifications for the software that we intend to develop, and the challenges we will face in the process.

CHALLENGES

This section focuses on discussing the challenges that we will face in developing the specified software. Our group is expecting to face challenges that might hinder our project's performance. There will be more challenges that will show up as we work on the project which are not mentioned below. Those challenges will be documented as we encounter them.

Optimization

The biggest challenge we will face in the production of *"Hand Signals with Visual Data"* is being able to optimize it in real-time on the average users' personal computer without abusing the computing resources available. We are anticipating that the software will be most costly to the CPU and the RAM because the method of prediction requires that we keep most of the data in the memory allowing for the processing. We would also need to limit the framerate and resolution at which the camera input is taken – lowering the framerate while keeping the resolution at the bare minimum would help us avoid hogging the available resources. We will optimize OpenPose to process only our hands and ignore everything else. And lastly, the Electron framework supports hardware acceleration but, it will only run when needed.

Gesture Recognition Training

Our next challenge would be training either a new model or modifying the existing model from OpenPose to allow for gesture recognition and to enable counting fingers. Apart from developing the methods that would allow for the interpretation of visual data from the camera input – the bigger challenge would be being able to collect a dataset sufficiently large to allow for training.

FEATURE SPECIFICATION

The software will be built with the following features.

Input

The software will use any available camera as an input. While the software at its core only needs an image to detect the hand position and pose, we will require a video feed that would allow for the estimation of hand gestures – allowing for interpreting hand movements over time. The software will limit the camera frame limit to ensure that the software does not abuse the resources available on the user's personal computer.

Target Platforms

The software will be aimed at all major desktop operating systems – Windows, MacOS, and Linux distributions. The varying degrees of compatibility in each operating system

will depend on the tool used to mock keyboard shortcuts. The development will most likely be done on the Ubuntu 18.04 operating system with the use of *"xdotool"* to mock keyboard inputs.

Customization

The software will allow for triggering built-in operating system shortcuts from a combination of gestures and finger count. The limitations to finger count and the variety of gestures available in the software will depend on the results of further research and development. For instance, allowing for usage of both hands might consequently cause performance issues and allowing for more gestures would require a larger and more varied gesture dataset to sufficiently and properly train our models.

Privacy

With the use of a camera as the primary input would raise concerns of privacy, an issue that we can combat by using on-device deep learning rather than having to send a video feed from the user's machine to a remote server. The software would however, we able to communicate with our server for OTA updates to models that would allow for gradual increases in the accuracy and detection improvements over time.

TECHNOLOGIES

Software Technologies

The technologies that will be used to create the software are listed and described below.

Electron Framework

The *"Electron Framework"* is an open-source project created and maintained by GitHub that simplifies making cross platform applications. Electron accomplishes this by using utilizing Google's open-source Chromium runtime and combining it with the Node.js runtime. Being built on a JavaScript runtime allows for the development and use of a single user interface across all platforms.

Node.js Runtime

Node.js is an asynchronous-event-driven JavaScript runtime, designed to build scalable network applications. Our software will use the Node.js runtime to create an exclusively locally hosted server to allow for communication between our Deep Learning library, the system tool used to mock inputs, and the software's user interface.

React

React is an open-source JavaScript library maintained by Facebook that allows for component-based user interface development. React combines the most common languages used in web-development – HTML, JavaScript, and CSS – into a single component.

Deep Learning Technologies

The technologies used to train, develop, and predict hand models, referred to as Deep Learning technologies, are listed and described below.

Caffe

Caffe is a deep learning framework made by Berkley's AI Research department with expression, speed, and modularity in mind. Caffe was built with on-device inference in mind – it allows for GPU based training and then CPU based inference.

OpenPose

OpenPose is a real-time system to jointly detect human body, hand, facial, and foot key-points. OpenPose will be used as the foundation for hand detection – using the prebuilt models from OpenPose we will create addons to enable isolated hand detection and add features like gesture detection (analyzing movement) and the ability to count fingers.

OpenCL

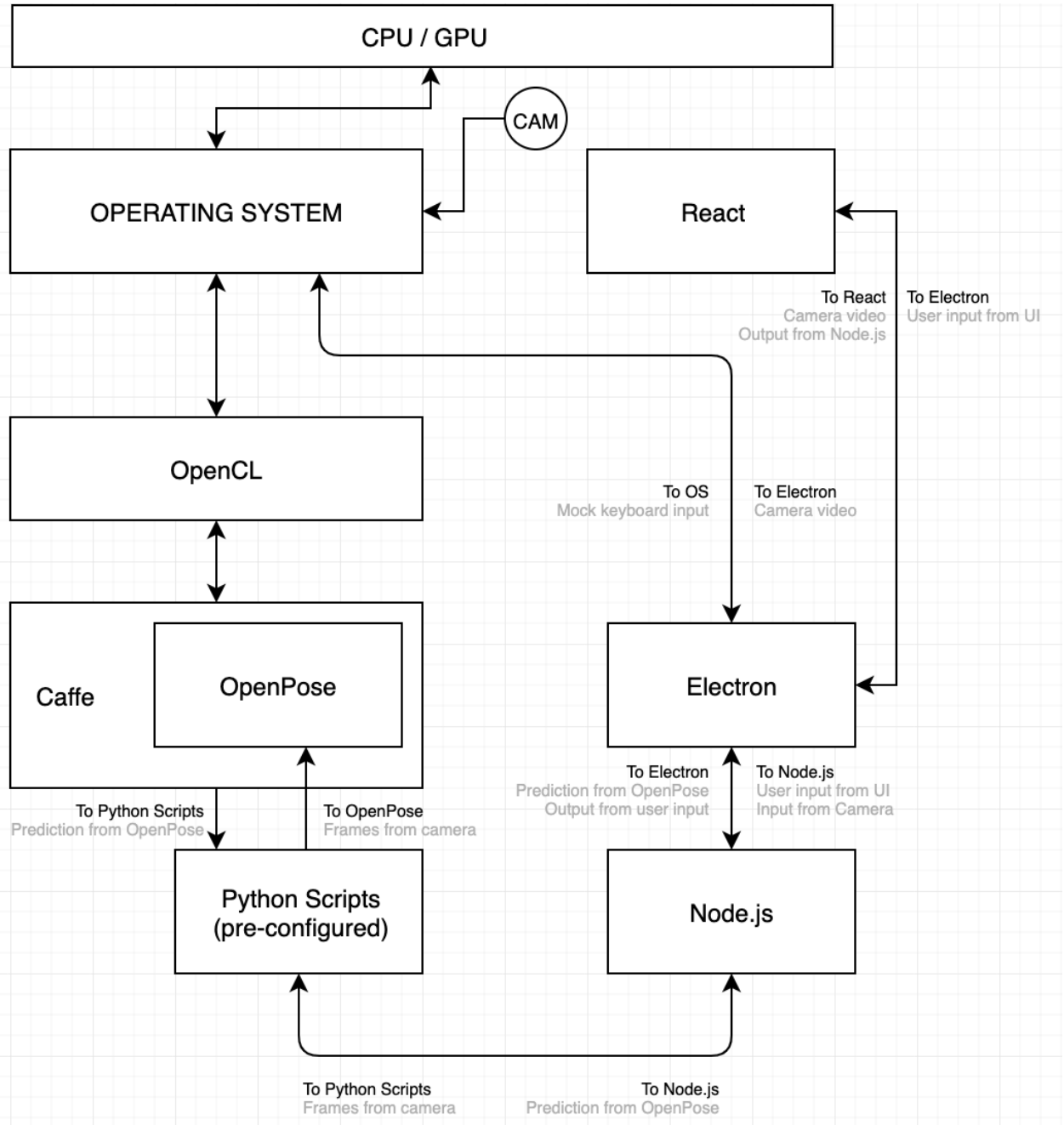
OpenCL is a framework for programs that execute across hardware accelerators. We chose to use OpenCL because it is the most widely available framework regardless of the hardware it is running on. In comparison to NVIDIA's CUDA, and AMD's ROCm – OpenCL is available for all Intel Processors as well as being available for NVIDIA and AMD's GPUs.

Python

Python is an interpreted high-level language, and our choice for the OpenPose API.

ARCHITECTURE

The diagram below presents a very basic form of our software architecture.



DEEP LEARNING

Deep learning is a sub-field of machine learning dealing with algorithms based on the structure of the brain called artificial neural networks. In other words, deep learning algorithms can be comparable to how our nervous system is structured in which a complex network of neurons are passing along information. Deep learning utilizes multiple layers to gradually extract higher level features from raw input, and by classifying the data, it can distinguish meaningful information such as faces, digits, etc. Deep learning also touches the field of artificial intelligence because of its ability to replicate human behavior.

Role of Deep Learning

There are many different deep learning architectures with the most common being: convolutional neural networks (CNN), recurrent neural networks (RNN), and generative adversarial networks (GANs). Each neural network can be applied in computer vision, speech recognition, and language processing—in which they could meet or even surpass human expectations. With the use of deep learning, researches have accomplished feats in various fields such as healthcare, transportation and computer security.

Role of Deep Learning In Our Project

Deep learning is essential to our project since our program involves tons of image processing from a webcam. Computers cannot understand raw images that are being displayed on the screen. However, we can utilize supervised learning to teach the computer what humans can interpret from those images. Supervised learning is a way of learning information based on labeled training data. By showing the neural network how our hands' appearance, it can learn how to differentiate how many fingers are displayed from the webcam. We can then link each output processed by our neural network to a gesture that users can configure in our program.

What Are Convolutional Neural Networks

Convolutional Neural Network or CNN is one of the most popular neural networks focused at computer vision tasks such as image recognition. The term "convolution" was derived from a mathematical operation on two functions to output a third function which relates how one was shaped by the other. Therefore, on its core, CNN contains fully connected layers in which each neuron in one layer is connected to all neurons in the next layer.

These are the 4 building blocks in making a CNN:

1. Convolution – receiving input signals from another layer and using filters/kernels to convolve across the input. This is where the network "learns" important filters as it runs.

2. Subsampling – also known as pooling layer. Reduce the dimensions of data with a filter size and a stride. Controls overfitting. Usually inserted after a convolution.
3. Activation – controls how information flows from one layer to another.
4. Fully Connected – connects all neurons in a layer from its previous layer to its sub-layers.

Why Use Convolutional Neural Networks

In our project, we will utilize deep learning by gathering visual data from a webcam and use that data to trigger commands in the user's machine. Our group picked CNN because of its ability to process images. Other neural networks, such as RNN, cannot excel in this task. However, RNNs can process language and speech better than any neural networks.

References

Deng, L.; Yu, D. (2014).
"Deep Learning: Methods and Applications" (PDF).
Foundations and Trends in Signal Processing.
https://www.microsoft....DeepLearning-NowPublishing-Vol7-SIG-039.pdf

OpenPose
https://github.com/CMU-Perceptual-Computing-Lab/openpose

Handtrack.js
https://github.com/victordibia/handtrack.js/
Library for prototyping real-time hand detection

Gesture Recognition
https://github.com/asingh33/CNNGestureRecognizer
https://github.com/avidLearnerInProgress/hand-gesture-recognition
Projects with examples for gestures recognition

TensorFlow Examples
https://github.com/tensorflow/examples
General examples from TensorFlow on gesture classification and object recognition

Kaggle Dataset
https://www.kaggle.com/.../hand-gesture-recognition-database-with-cnn/
Database is composed by 10 different hand-gestures

END OF PAPER