CS CAPSTONE REQUIREMENTS DOCUMENT

APRIL 28, 2020

GESTURE RECOGNITION USING NEW INTEL REAL SENSE LIGHT CODED CAMERA

PREPARED FOR

INTEL

EDUARDO X. ALBAN	Signature	Date
SATOSHI SUZUKI	Signature	Date
PO-CHENG CHEN	Signature	Date

PREPARED BY

GROUP 33

GESTURE RECOGNITION

JONATHAN HULL	Signature	Date
NICHOLAS DAVIES	Signature	Date
SHANE CLANCY	Signature	Date
SHIHAO SONG	Signature	Date
ULISES ZARAGOZA	Signature	Date
ZHIDONG ZHANG	Signature	Date

Abstract

Our group's chosen project for the quarter is the '*Gesture Recognition using new Intel RealSense coded light camera*', and our group consists of the following individuals: Shane Clancy, Nicholas Davies, Jonathan Hull, Shihao Song, Zhidong Zhang and Ulises Zaragoza. This requirements document aims to outline the system requirements of the project and go into detail of how we are expected to verify different project components. We will go into detail about the functionality of the Intel RealSense camera, machine learning model, and graphical user interface, as well as outline the performance requirements for the machine learning model and the graphical user interface. We will also cover the system interface, environmental conditions, information management, policies and regulations, and packaging, handling, shipping, and transportation. Verification of each of these requirements will also be discussed.

Contents

1 Introduction

1.1 System Purpose

Our team will collaboratively work to create and train a machine learning (ML) algorithm with the task of gesture recognition using the Intel RealSense Depth Camera SR305. In addition our goal is to build a Graphical User Interace (GUI) application to house the model and utilize it. The machine learning model will be expected to have the ability to classify American Sign Language (ASL) gestures from a single image frame that is provided by the RealSense Camera.

1.2 System Scope

The system scope of our project has an end goal of having a ML be able to classify sign-language gestures from the ASL domain, and present this classification via a text message to the user through the housing GUI application. This GUI application will allow interface capabilities with the RealSense camera, and will be a direct source of input to the ML model. Our project will take on a much smaller initial scope, as we seek to first build and train a foundation ML model that can perform basic gesture recognition tasks from just a few letters of the domain. Once this has been established, we will work to fine-tune the model and prepare it to handle our end-goal task of sign-language recognition of all letters, excluding the letters J and Z because of their need for multiple frame recognition.

1.3 System Overview

1.3.1 System Context

The camera we will be using is the Intel RealSense Depth Camera SR305. This indoor camera has an incredible quality of depth, specifically when used at the ideal one meter distance. The code written for this device is easily transferred to utilize on other Intel RealSense gadgets, making it user friendly. This camera is also very cost effective, making the product affordable for a wider range of customers.

ML Algorithms are a branch of artificial intelligence that allows computers to learn for themselves. ML Algorithms can be supervised or not, allowing them to learn for themselves or have strict guidance depending on the application. In this context, our ML Algorithm will be supervised, and is going to learn to recognize patterns through visual recognition and coded light in the training frames that we provide. This will then allow the system to classify new image frames of letters from which it has been trained.

Coded light systems, a tool utilized by RealSense cameras, interpret the depth and distance by using infrared light. The infrared light is a single pixel that is cast onto the object to see if the pixel lands on a dark or light stripe. Then, a binary code is created for each pixel. This data is then recorded and memorized on the device to help it interpret certain gestures. The Intel RealSense Depth Camera SR305 has a 640x480 depth resolution as well as 60 frames per second, making for quality video/image capturing capabilities[1].

1.3.2 System Functions

Our team will be utilizing the SDK provided by Intel on GitHub which will allow us to interface with the camera and use the camera's features. The way we will be implementing our ML Algorithm is through transfer learning approach which is using a pre-trained ML Algorithm as a baseline classifier, and will then train the task of gesture recognition from this foundation.

1.3.3 User Characteristics

The individuals we are targeting with this software are those with language barriers. Our goal is to make a software that allows language-disabled people to communicate smoothly with the movement of their hands. We hope to tackle the barrier of interpreting sign language for those not familiar with the gestures, and furthermore to achieve the goal of helping people with language disabilities communicate with the outside world.

2 System Requirements

2.1 Functional Requirements

The following sections outline the functional requirements that this project needs to meet. We will also evaluate how we will measure these functional requirements in order to meet the needs and expectations of the clients. We will be measuring the functionality of the Intel RealSense camera, machine learning model, and the Graphical User Interface.

2.1.1 Intel RealSense Camera

The Intel RealSense camera is a light coded camera that we will be using the capture data for our project. The software for this component has already been developed in an open source environment by Intel. The camera needs to be attached to the computer through USB 3.1, then the Intel RealSense SDK 2.0 has the capability to find and read input of the Intel RealSense camera itself. Our project repository will always include the proper SDK that is compatible with the Intel RealSense SR305 Camera. We will be measuring the functionality by making sure that the depth of certain objects is measured correctly through manual measurements.

2.1.2 Machine Learning Model

The machine learning model is the brain of the project, and will need to interact with the Intel RealSense camera, as well as the GUI application. We will be doing our development for the machine learning model using Python3 and the PyTorch library, which need to be installed in order for the model to be able to train and classify information that is provided to it through the data storage that we will setup to store our information. Measuring the functionality of the machine learning model will be done after the model is developed and trained. We hope to achieve high accuracy with our model, and we will be measuring its accuracy throughout the development on a set of data that we set aside for testing. The metrics by which we will test our model's performance is through a confusion matrix, which measure the per-class accuracy of the classifications made by the model.

2.1.3 Graphical User Interface

The graphical user interface will be in the form of an executable that should be deployable on all platforms. We will need to make sure that this application can be deployed on each platform before we release the project. The user interface needs to be accessible to people who lack verbal communication, and customizable to their needs. Consistent manual testing on the interface needs to be done to ensure the usability meets these standards.

2.2 Performance Requirements

The following sections outline the speculative performance requirements for our project's chosen ML Model and the GUI application our team will create to house and utilize the model. See the Verification section for information on how our group will ensure performance criteria is met.

2.2.1 Machine Learning Model

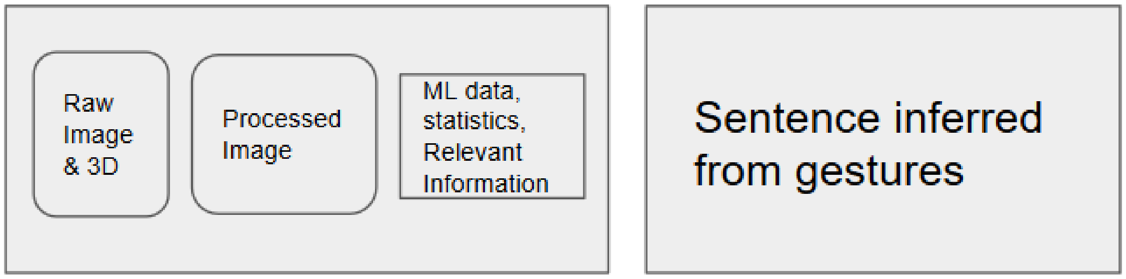
Our ML model will be expected to classify single frame gestures from the American Sign Language alphabet, excluding the letters J and Z because of their need for multiple frame recognition due to the movement needed to sign the two letters. No quantitative metrics have been established on the amount of accuracy in which the model will be able to perform these recognitions, however it is expected to have sufficient training data across all planned classification letters in order to appropriately train the model. Our group is expected to utilize the output from the confusion matrix to determine how well our model is at classifying the letters, and will help dictate if more data or training is needed for our model.

2.2.2 Graphical User Interface

The performance requirements expected of our GUI include the ability to securely transfer RealSense image feed output as input to the ML model of our project. Our encasing GUI must be able to securely and dependably read input from the RealSense camera, without communication issues between the software and hardware. Furthermore, it is expected that our model's classification is output in a reasonably quick manner. Exact time specifications may be detailed upon deeper experimentation with RealSense cameras and related SDK's.

2.3 System Interface

The design requested by the client is to provide user with two different screens. The first screen is used to capture/present data from the camera. Within the first screen there will be three components displayed to the user: an RGB and 3D image, the filter-processed image, and relevant ML interpretation information information (e.g. accuracy, statistics, etc.). The second screen is used to display the ML gesture recognition in its string literal form. A representation of this visual using the two screens can be seen directly below:


Raw Image & 3D
Processed Image
ML data, statistics, Relevant Information
Sentence inferred from gestures

This layout will display to the user all relevant information that is calculated from our project. Beginning with the raw input they give to the camera, and ending with the final recognition made from our ML algorithm. This UI layout captures all parameters of our project, and will effectively allow a user to input gestures to the camera, and receive the ML classification.

2.4 Environmental Conditions

The GUI Environment should be on the development machine the client chooses. If no machine is given, then the development location is subject to change. The GUI should be able to run on Windows 10, IOS, and Linux. The environment on which it runs should have internet access and enough processing power in order to properly run all system components and program requirements.

2.5 Information Management

The information most pertinent to our group's project is the training data we will use as input for our chosen ML model. The management of this data will involve identifying a candidate storage method followed by beginning the process of collecting images to use as training/test/validation input for our ML model. Due to the high importance of obtaining plentiful and quality forms of training data, the process of collecting data itself will also take on a form of management. This form will involve a strict set of guidelines detailing which types of gestures we are obtaining, image quality, number of different gesture iterations, involving diverse participants, and an equal number of training images across the separate labels. Our group has decided upon 500 training images for each letter we will be classifying. The letters to capture will be spread across all members of the team to ensure diversity in training examples.

2.6 Policies and Regulations

For work completion, all group members are in accordance with the given work assigned. If the work completed is code-based, then a peer-review has to be completed prior to submission. If the work completed is text-based, then at least two other group members have reviewed the document. Each process should establish an outline at the beginning and the task should meet all requirements specified by the outline. Work should sufficiently accomplish task at hand. Work should be well thought out with clearly fleshed out ideas. Meeting attendance, is required for every group member unless properly communicated, or unforeseen circumstances occur. If we encounter problems, it is best to solve them internally. If we can't solve them, then we will escalate as needed.

2.7 Packaging, handling, shipping, and transportation

The cameras were initially sent to one group member's home and have been secured. We have divided the cameras amongst the team members, and plan to give equal access throughout the first term so we may all familiarize ourselves with the equipment. Equal equity among camera access is essential for proper understanding of project completion and components.

3 Verification

3.1 Functional Requirements

The following sections outline the verification techniques our group will act on in order to make sure all components of the project are functional.

3.1.1 Intel RealSense Camera

The camera needs to cover many edge cases that could be encountered based on the depth calculations and bad IR sensor data. Although the product has been thoroughly tested by Intel, we will need to make sure that the camera can obtain edge case data for certain gestures that we want to classify. While our group is gathering data, we hope to find new edge cases to explore with the camera. This will help the accuracy and training of the machine learning model directly.

3.1.2 Machine Learning Model

The machine learning model will need to be regularly tested in different ways. Once a model is built, we will verify feature importance and relevance, feature thresholds, and feature relationship with output. Some features might not be relevant to classifier, and can be pruned accordingly to increase speed of training and advancement of the model. Feature thresholds are bounds for specific features, and if features can specifically fall within a range, this can be analyzed and a multi-layer model can be developed in order to account for these ranges relating to specific output that is wanted. As a group we will need to continually verify that features are being given different priorities in order to create an accurate model.

3.1.3 Graphical User Interface

The graphical user interface needs to go through manual testing, by many different people. This will make sure that the user interface is intuitive and usable by different types of people with different goals for the program. The user interface needs to be customizable to different needs, and any specific need that is not met, should be addressed during verification steps of new feature introduction.

3.2 Performance Requirements

The following sections outline the speculative verification techniques/guidelines our group will implement in order to ensure the validity of the performance requirements established above.

3.2.1 Machine Learning Model

An mentioned prior, the metrics by which we will measure the performance of our ML model is by testing the performance on a set of data that is completely independent of the set of data used to train it. The specific metric will be done via confusion matrix, which measure the per-class accuracy of the classifications made by our model. This will give our group a solid idea of what classifications are being properly done by our model, and which labels will require more training data increase overall accuracy.

3.2.2 Graphical User Interface

The main form of verification for performance of the GUI application, which houses the ML model, will come through various forms of testing of the application. Usability testing, unit testing, regression testing, and other forms, will be utilized on our GUI application to ensure its quality in performance and functionality. Furthermore, this will help our group ensure that proper communication is occurring through our entire project's pipeline. Starting from the input received by the camera, followed by assurance that the RealSense video feed is being properly fed into our ML algorithm. Lastly, tests will help ensure that the output being displayed by the GUI is true to the output produced by the ML model. The various forms of testing conducted by our group on the housing GUI application will aid in ensuring the performance quality of the entire project, not just solely the GUI application.

3.3 System Interface

Users will be able to consistently open the GUI application to interact with the RealSense camera, and the housed ML model. Apart from the rigorous manual testing our group will perform for verification of performance and functional requirements, specifical manual tests will be performed to ensure the quality of the usability for our GUI application. We will be looking to establish metrics such as ease of use, and specifically we hope to address issues that could arise for users with specific accomodation needs.

3.4 Environmental Conditions

The system interface should be able to run on popular operating systems. Anything from Windows 10, IOS, and Linux. The focus will be on whatever development machine we are tasked with building this on. The machine will need Internet connection, ports to connect to the camera, processing power to run the machine learning model, and access the database. The interface can also be made to be portable such as exporting it to a Raspberry Pi or Arduino to run if the group decides that this is its end state. The database will need to have an adequate amount of space to store training videos. It is important that the computer running this interface is not hindered by its processing power to avoid unnecessary delays.

3.5 Information Management

Ensuring quality and an adequate quantity of training data is vital to the success of our group's project. Therefore, a formal team meeting needs to take place in order to detail the guideline set and timeline for our project's data goals. Furthermore, a form of data storage needs to be decided amongst our group members and clients, with the idea in mind that we are going to potentially need a large storage room to store videos as our training/test/validation sets. Once this meeting has taken place, along with the specification of guidelines, timelines, and data storage methods, this document may be signed off.

3.6 Policies and Regulations

The policies and regulations of this system interface should reflect those of our client and Oregon State University College of Engineering. This project should aim to keep anonymity, as well as privacy of any individuals caught in the camera's lens. Gesture collection should only be used for maintaining the accuracy of the machine learning model and keeping and up to date database. If this project is used commercially, then this section is subject to change for its correct purpose.

3.7 Packaging, handling, shipping, and transportation

The only package-able items for this interface would be the GUI. The machine learning model and the database can be remotely connected to. The could should either be able to be zipped up or tarred to another machine that can access the machine learning model and the database. Transportation may vary from email, cloud drives, or any sort of transferring service. Shipping is not applicable to software unless the interface becomes too large to send over the Internet.

4 Appendices

4.1 Assumptions and Dependencies

4.1.1 Assumptions

- This document is a fluid requirements list created by the group members in collaboration with the project clients. Future meetings and work will help cement details within this document that reflect the needs of both the group and the client. This document is subject to change.

4.1.2 Dependencies

- The Intel RealSense SDK will be used by our group to interface with the camera and receive video feed[2].

4.2 Acronyms and Abbreviations

- SDK: Software Development Kit
- ML: Machine Learning
- GUI: Graphical User Interface
- CNN: Convolutional Neural Network

4.3 Gantt Chart Timeline

Figure 1 below details the tentative timeline for group's projected milestones. This timeline currently displays only high level goals and is anticipated to change once finer project details have been firmly established.

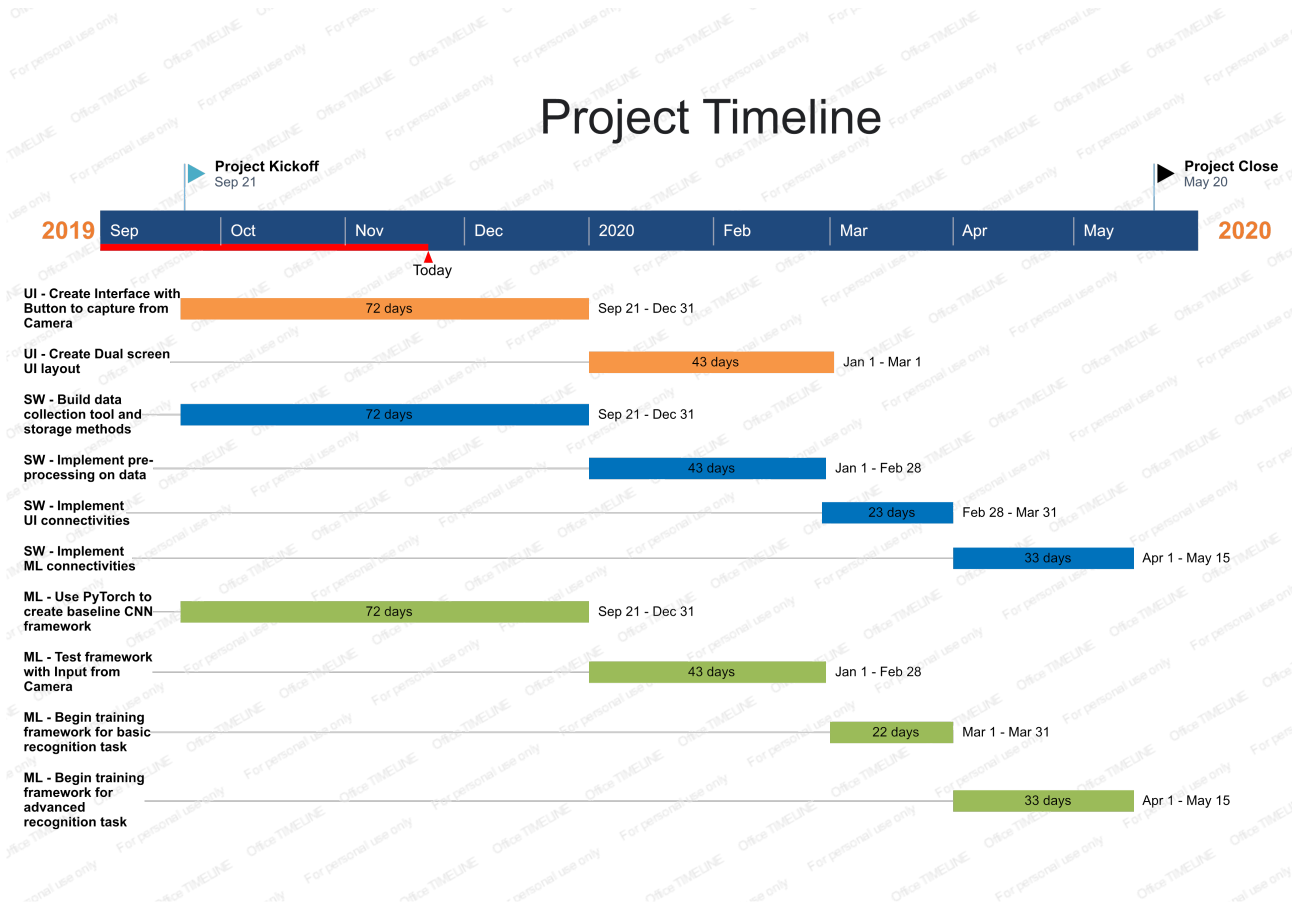


Fig. 1. Gantt Chart

5 CHANGES

5.1 Change Table

Change Table	
Project Component	Changes made
Machine Learning	The main changes that have occurred on the Machine Learning portions of the project include the fact that our project scope has changed from video recognition of ASL gestures, to single frame recognition of gestures from the ASL alphabet. Furthermore, the type of data being processed by the ML algorithm has changed from matrices of raw pixel values, to depth images that have been applied a color-map that represent different ranges of depth within the image. This type of data suits the Resnet-18 expected input more naturally than our original implementation of the matrices of pixel values.
Software	After our first implementation, we decided to go with a different approach to capturing images for the machine learning model. Instead of using the raw depth images, we decided to convert eat depth image to a color-map, and then process these images with the machine learning model. We were able to capture more training examples this way as each image took less time to capture. This gives us a more accurate training set for our machine learning model to use to classify gestures.
UI	The main change that has occured to the UI is addition of extra instructions upon the startup of the interface, as well as connecting the information display of classification confidence from ML portions, and depth values from SW portions.

References

[1] Intel. *Intel® RealSense^{TM} Depth Camera SR305*. Intel, 2019. Retrieved on October 15, 2019 from https://www.intelrealsense.com/depth-camera-sr305/.

[2] Intel. *Intel® RealSense^{TM} SDK*. Intel, 2019. Retrieved on October 11, 2019 from https://github.com/IntelRealSense/librealsense.