IT Carlow –

BSc.

Software Development

UAV using Convolutional Neural Networks

Design Manual

Student Name: Josh Hudziak

Student Email: C00231846@itcarlow.ie

Date: 13/11/2020

Supervisor: Dr. Oisin Cawley

Supervisor Email: oisin.cawley@itcarlow.ie

Abstract

Within this document the design of the UAV with convolutional neural network design will be detailed. The system architecture and functional design will be outlined in detail as found from research and ongoing implementation. The project plan will also be laid out and the technologies being used to implement the architecture.

Table of Contents

Introduction

A UAV is an unmanned aerial vehicle. This project is aimed to create a UAV capable of autonomous navigation using a CNN (Convolutional Neural Network). Therefore, the UAV will be able to navigate over paths, roads, and tracks without human aid.

A CNN is based on traditional neural networks but works on an image dataset. The idea of convolution is to interpret features of an image and create a feature map using a feature detector. From these feature maps a comparison with other images or features on that image can be made and therefore identify and classify the image. There are many forms of CNN to work with, for this task the ResNet architecture will be used. This architecture is residual based and therefore can be built with many layers while still being computationally smaller than other architectures.

To train this model to navigate a UAV around different roads, tracks and paths a dataset is needs to be obtained, for this project the dataset is compiled of video recordings taken from roads, tracks and paths and have been cut into frames and labelled into three categories; left, right and straight. These frames can also be pre-processed and augmented to drastically improve the model's performance, some of these techniques will be shown in this document.

The UAV currently available for this project is a Parrot Anafi. This specific UAV runs with Parrot's own SDK and controller software called Olympe, simulation software is also available to run trial projects using Parrot's hardware and software, this is called Sphinx. The Olympe controller can connect to the physical Anafi and is being run on a Windows laptop running a Virtual Box virtual machine which contains Ubuntu 18.04, this is the easiest way to run Parrot's software.

The Dataset

The dataset for any machine learning or deep learning model will always reflect the task at hand. Here, the task of the UAV is to navigate a path, track, or road autonomously, therefore, the dataset to complete this task must reflect navigation of roads, tracks and paths.

To acquire such a dataset an Akaso Brave 7 LE action camera was used to capture different roads, paths, and tracks. The road footage has been shot inside a car while mounted to the car's dash, the track and path footage was shot while the camera was strapped to a walker, this method requires the walker to stay as steady as possible while they stay always facing towards the track or path.

This footage is shot at 30 frames per second and in 1080p resolution. All footage is open to a wide variety of outside lighting and weather conditions, but the camera controls all exposure and white balancing automatically, therefore, reducing motion blur. Image stabilization was also utilized in the camera setting, this gives a more neutral stabilization between frames from driving and walking.

Each frame of this footage is then separated into 3 divisions; Left, Right and Straight. As a result, these classifications will outline the direction of the road, track, or path. Therefore, a deep learning model can learn directions of each image and classify it into a left-hand turn, a right-hand turn or a straight.

Figure 1, 2: Here is an aerial view of the type of track the drone should be able to navigate autonomously.

Figure 3: Image from the dataset of a right-hand turn. This image would indicate the UAV to turn right.

Figure 4: Image from the dataset of a left-hand turn. This image would indicate the UAV to turn left.

Figure 5: Image from the dataset of a straight. This image would indicate the UAV to maintain a straight path.

This dataset must be split into training and testing data. In the case of this data 80% of the final dataset will belong to the training set and the remaining 20% will belong to the test set. This can be easily changed in the codebase if improvements can be made by creating a larger/smaller training or test set.

Data Pre-Processing

All images that the model will be trained on must be of a fixed size, therefore, all images must be resized to this specific shape. As the dataset being worked on is in colour, RGB channels will be maintained, this leaves the image dimensions of (x, y, 3) -> (width, height, RGB channels).

Next to improve computation and reduce complexity image noise will be removed. Image smoothing is a great function to reduce a lot of noise. The Gaussian function blurs an image, as the functionality of the model does not require image detail blur will take out that detail and maintain the basic image structure. Segmentation may also be added to the image if it creates a clear enough contrast between the road, path or track and its surrounding area. Segmentation separates an image's foreground and background. Further improving the Segmentation technique, markers can be used to truly outline objects.

As a final step data augmentation can be applied. This may increase the size and diversity of the original dataset by changing aspects of images in the original dataset and then compounding them together. This is also an excellent way to prevent overfitting. Such techniques include flipping the image, zooming, and shearing an image.

Model Architecture

To solve the problem of autonomous UAV navigation we must classify what the problem is and the result being the problem of image classification. This model must train a UAV to navigate based on the image retrieved by the UAVs onboard camera. The CNN model being used to solve this classification problem can be broken into two main stages.

The base, where features of the image are extracted by performing layers of the convolution function and the head, where the image is classified, is usually a dense layer that culminates outputs using a function. The function used depends on the type of classification, binary classification uses the Sigmoid function and categorical classification uses the SoftMax function.

Feature extraction which takes place at the base can be simplified into 3 phases. The first phase is to filter the image for features, next, the features must be detected, this detection is made possible with the Rectified Linear Unit. The rectifier function takes out any negative values within the filtered image. The last phase is to condense the image, this produces a clearer filtered image, within this model several layers of Max Pooling will be used to accomplish this.

Each layer of a CNN must be connected. To connect layers, a feature detector of sizes usually 3x3, 5x5 or 7x7 is used. This feature detector acts as a lens in which certain features of an image are highlighted, using convolution which basically gets the dot product of a matrix, the feature detector and section of image being examined by the feature detector are passed through the dot product function. The output of this convolutional function is a feature map. Using feature maps a CNN can classify an image as it learns features of the dataset in training.

Once image filtering is finished, the outputs, which are the feature maps, will be run through the ReLU function. This function can be a separate layer or part of the convolutional layer. As stated previously, this function sets any negative values to zero. As it is a nonlinear function adding multiple layers maintains diversification of outputs.

To condense this model, Max Pooling will be used. As a result, from using the ReLU function some pixels are set to zero. This still provides special information but if we apply Max Pooling this space is used up by the feature and thus amplifying that feature. Using Pooling creates translation invariance, which is basically the destruction of special information, but this can help the network as less data is therefore needed in training the model.

Residual Network Architecture

Building a deep CNN can create large numbers of parameters which in turn heightens computational complexity. So, to combat this problem in this model, the ResNet architecture will be used. As a result, this model could speed up the response of the UAV while deciphering the patterns in its returned image.

The concept of residual learning allows models to be created that have a lot more layers than conventional CNNs without causing vanishing gradients, accuracy degradation and optimization difficulties.

ResNet uses skipping of layers to maintain some identity of the previous activation. To have this functionality available, all convolutional layers must be the same size, which is most commonly a 3x3 convolution. In a worst-case scenario, where the activation function is negative, learning is not taking place but if the identity of previous layers is added, some representation of learning can continue, thus, more layers can be added.

To further enhance the ResNet model, Batch Normalisation can be used, which in a way is a form of pre-processing at every layer. It works by normalizing the previous layers' outputs.

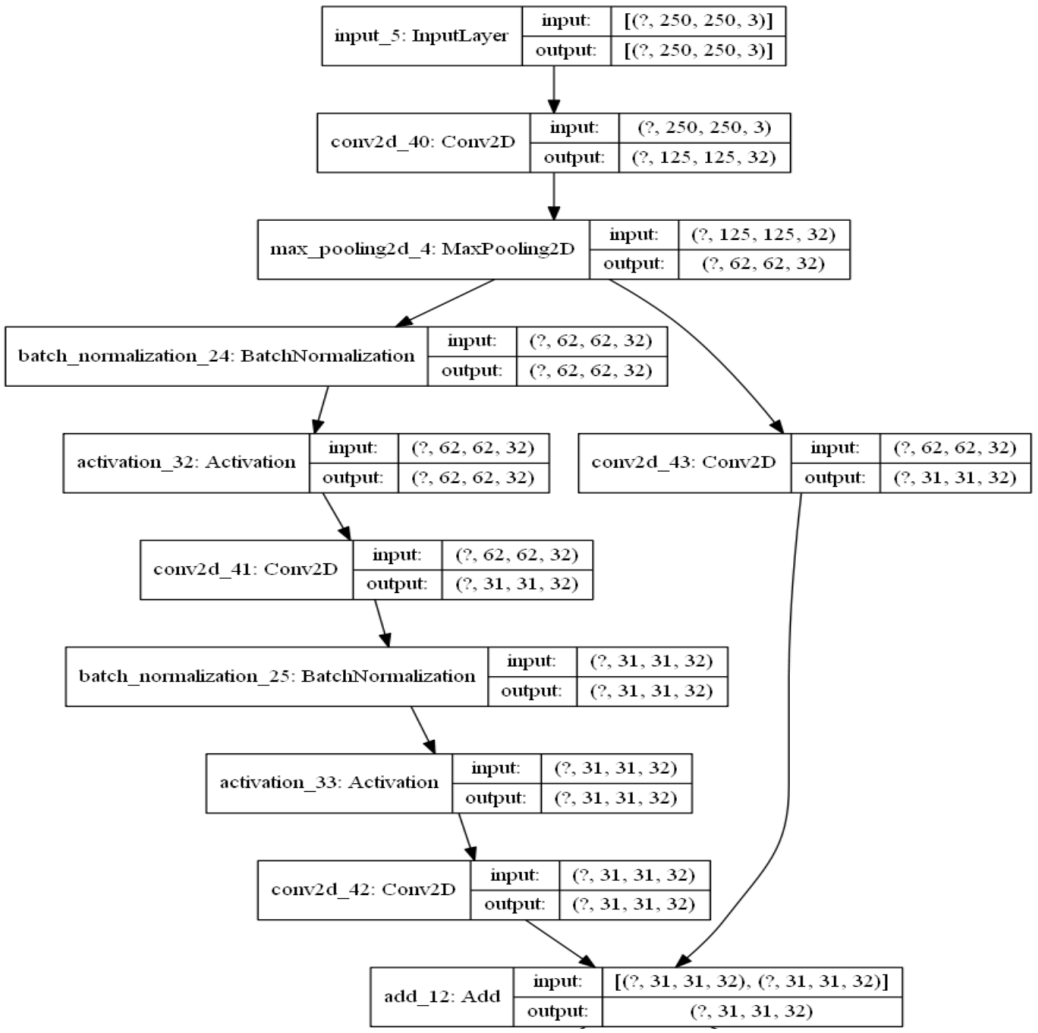


Figure 6: Input layer, which is a pre-processed image, followed by the first ResNet block, where the identity variable skips to and adds into the next residual block.

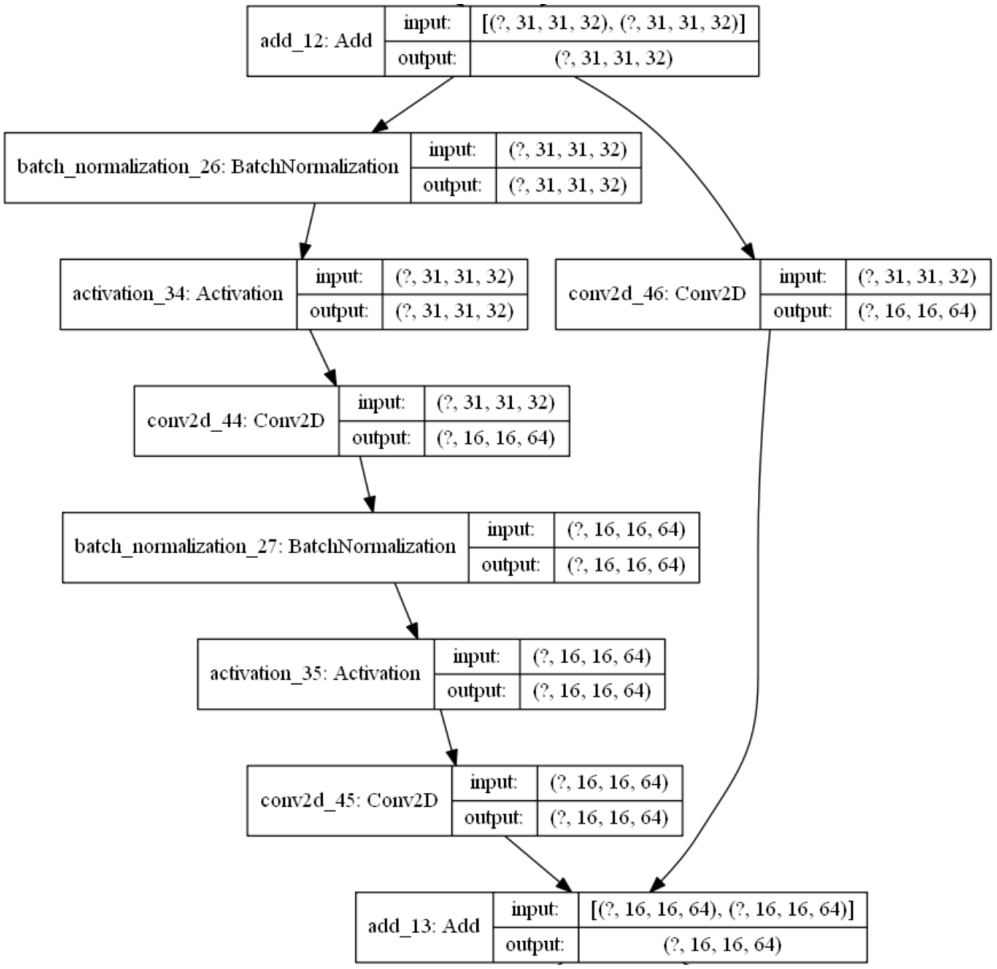


Figure 7: Second ResNet block. Notice that the blocks are the same setup of batch normalisation, the ReLU activation function and a 2-dimensional layer of convolution.

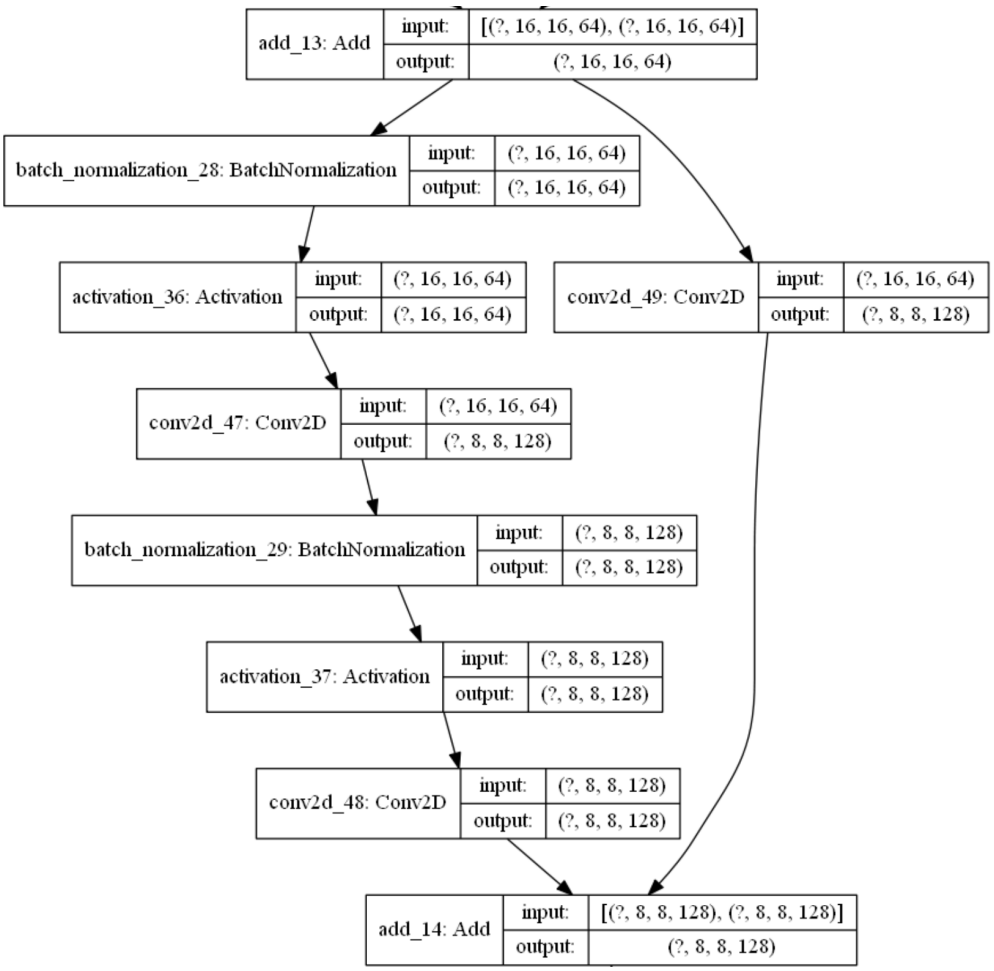


Figure 8: The third and final ResNet block follows the same pattern.

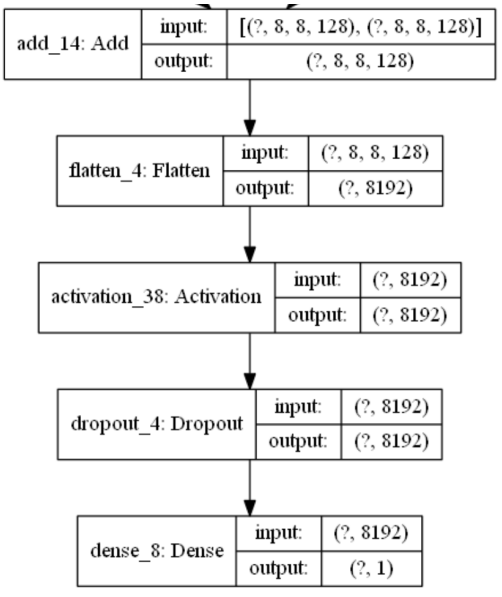


Figure 9: The last layers of this network entail Flattening to create a linear 1-dimensional array and SoftMax which gives a value between 0-1, this creates a dense network to classify image input.

Building Technologies

This implementation would be nearly impossible to create from the ground up, in a short amount of time. Thankfully, technologies exist that simplify all the mathematical procedures which are used to create the architecture previously mentioned. Unfortunately, to run this model in a stable version, all these technologies are reliant on each other and pair specifically to each other, this also holds true regarding the technology's version.

As an overarching technology, Ubuntu 18.04 is being used as an OS which currently is the operating system of choice to run Parrot's Olympe software as stable as possible. Therefore, the model will run in this environment. As an added remark, in this case, Ubuntu is running on the latest version of Virtual Box hosted on a Windows 10 environment.

At a higher level, the following are the technologies of choice:

Olympe

A programming interface that communicates with the Parrot Anafi. This program will act as a controller to the UAV and for simulated UAVs in Sphinx. Olympe connects to the UAV using Python scripts, thus, the user can send commands and receive statuses, information, and errors from the UAV. This program is also the main reason for using Ubuntu 18.04 as it only runs stable on this OS or Debian 9.x.

Sphinx

Another Parrot software that simulates environments to test created features for the UAV. Sphinx runs Parrot's firmware and uses a software called Gazebo to produce virtual environments and a UAV simulation to test functionality and programs on. Sphinx can produce flight data and virtual behaviours.

Python 3.7

The base language for all development on this project. Python is used in this project, as many of the other tools pair well and use Python. Python has built in garbage collection and is very easy to implement. It is widely used in machine learning and has many libraries and frameworks for this such as scikit-learn. Anaconda is an easy-to-use Python environment that runs on Linux and Windows. During this project Anaconda will be used to bundle technologies together and run the model on the dataset.

TensorFlow with Keras

These libraries have many machine learning features built in and will simplify implementing the ResNet, ReLU, SoftMax and CNN. TensorFlow is an open-source library built by Google's brain team. It holds a large repository of neural network algorithms and models which entail mathematical solutions such as convolution. Keras is a high-level API used for TensorFlow. It creates an easy implementation method for neural networks by using a block method. As the machine which runs this project uses Windows 10 primarily, the model will be run on Windows using TensorFlow and Keras' GPU enabled versions. This will drastically improve the model's learning rate.

Open CV

OpenCV is used to process images from the collected dataset. Using NumPy, this library has a repository of over 2500 algorithms to detect and track objects, thus solving a lot of problems within the computer vision field. We will be using it with Python, but it is also compatible with many other languages such as C++ and Java.

NumPy

NumPy is a library that works on large multi-dimensional arrays, it is extensively a Python library. It holds a vast repository of high-level mathematical algorithms. Using NumPy can generate results 50 times faster than normal Python lists. In this project we will use it in tandem with OpenCV to process images in the dataset.

Overview of Architecture

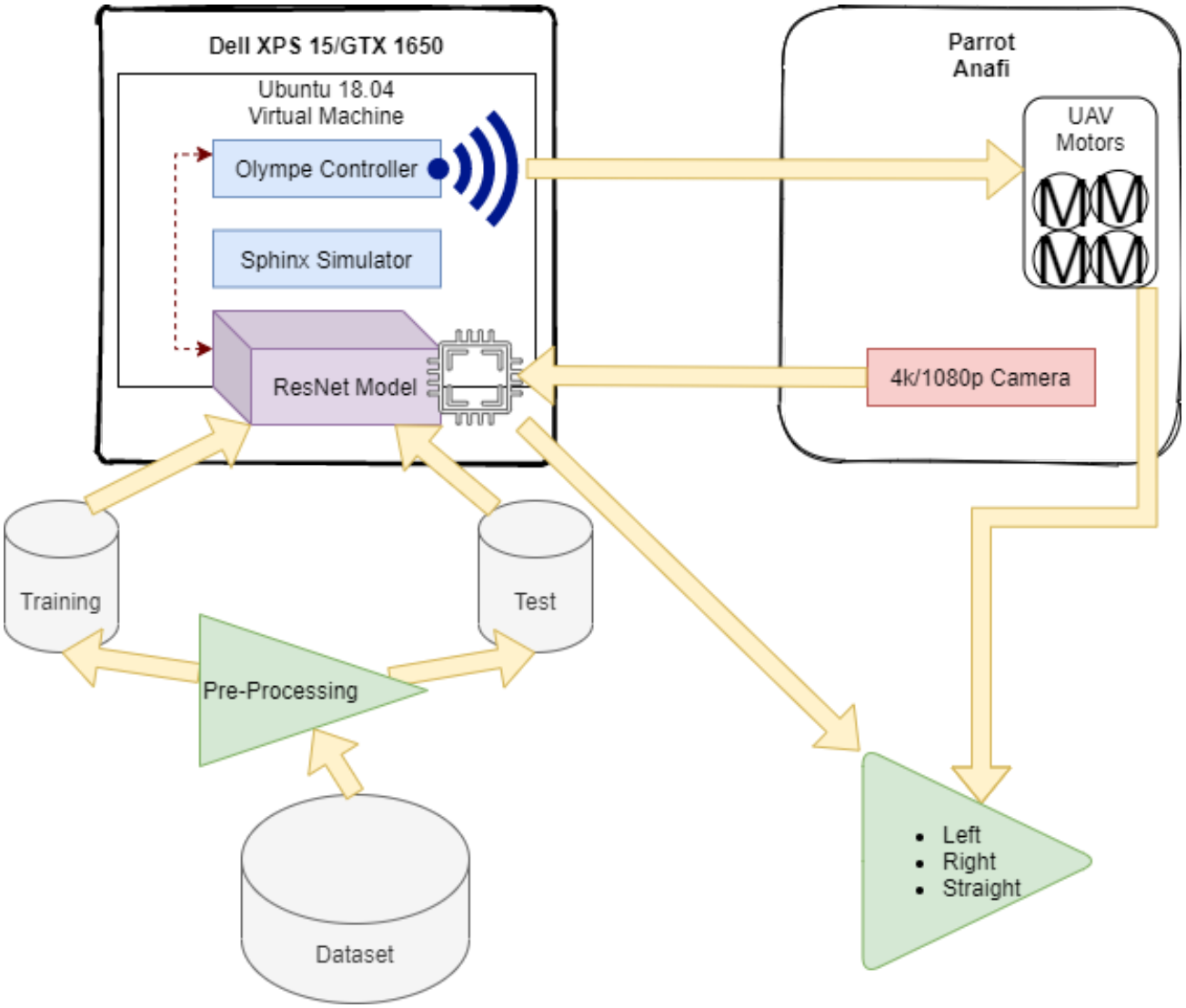


Figure 10: Above is the Flowchart at a low level of communications and data flow within the project.

Process Flowchart

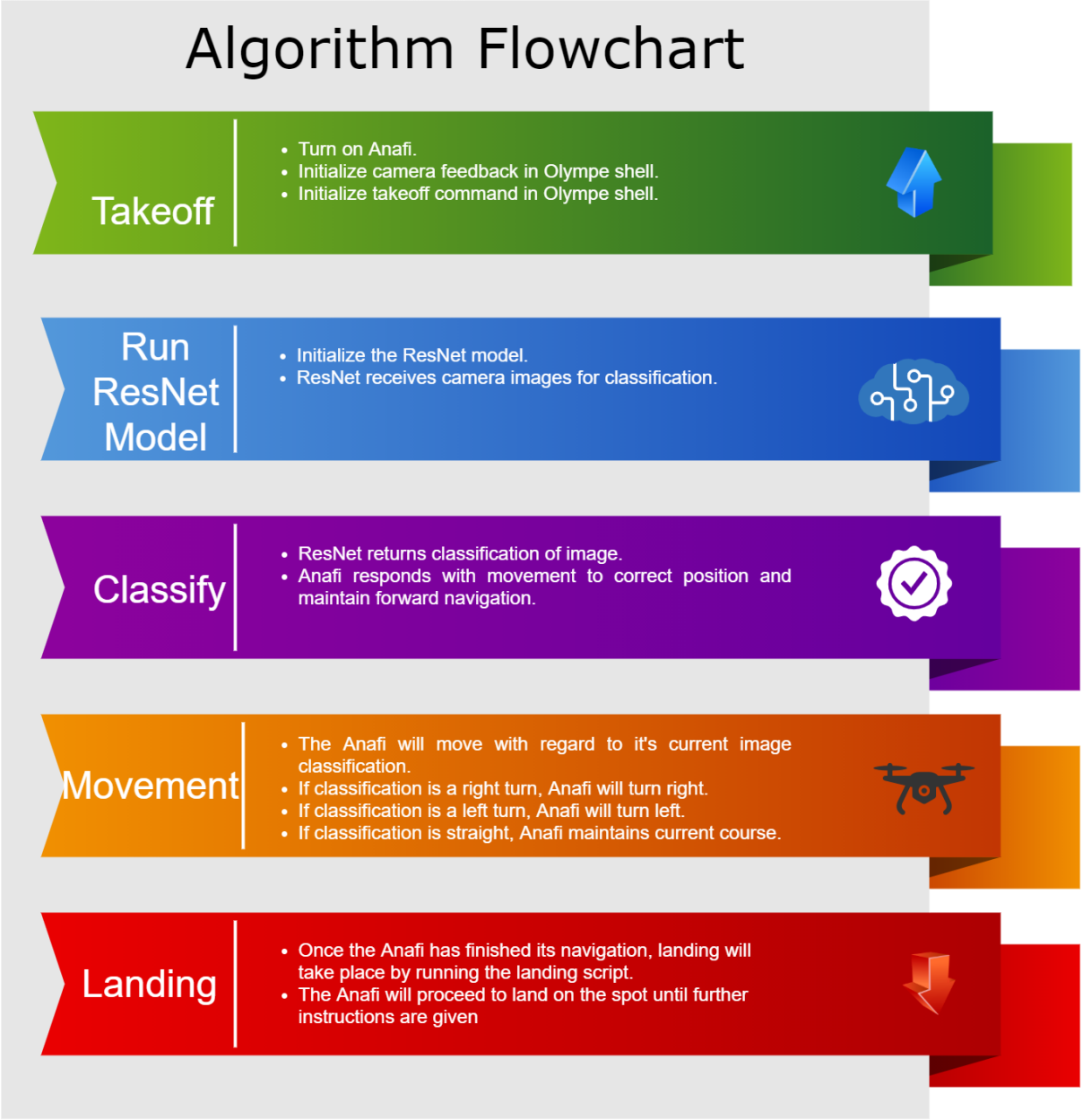
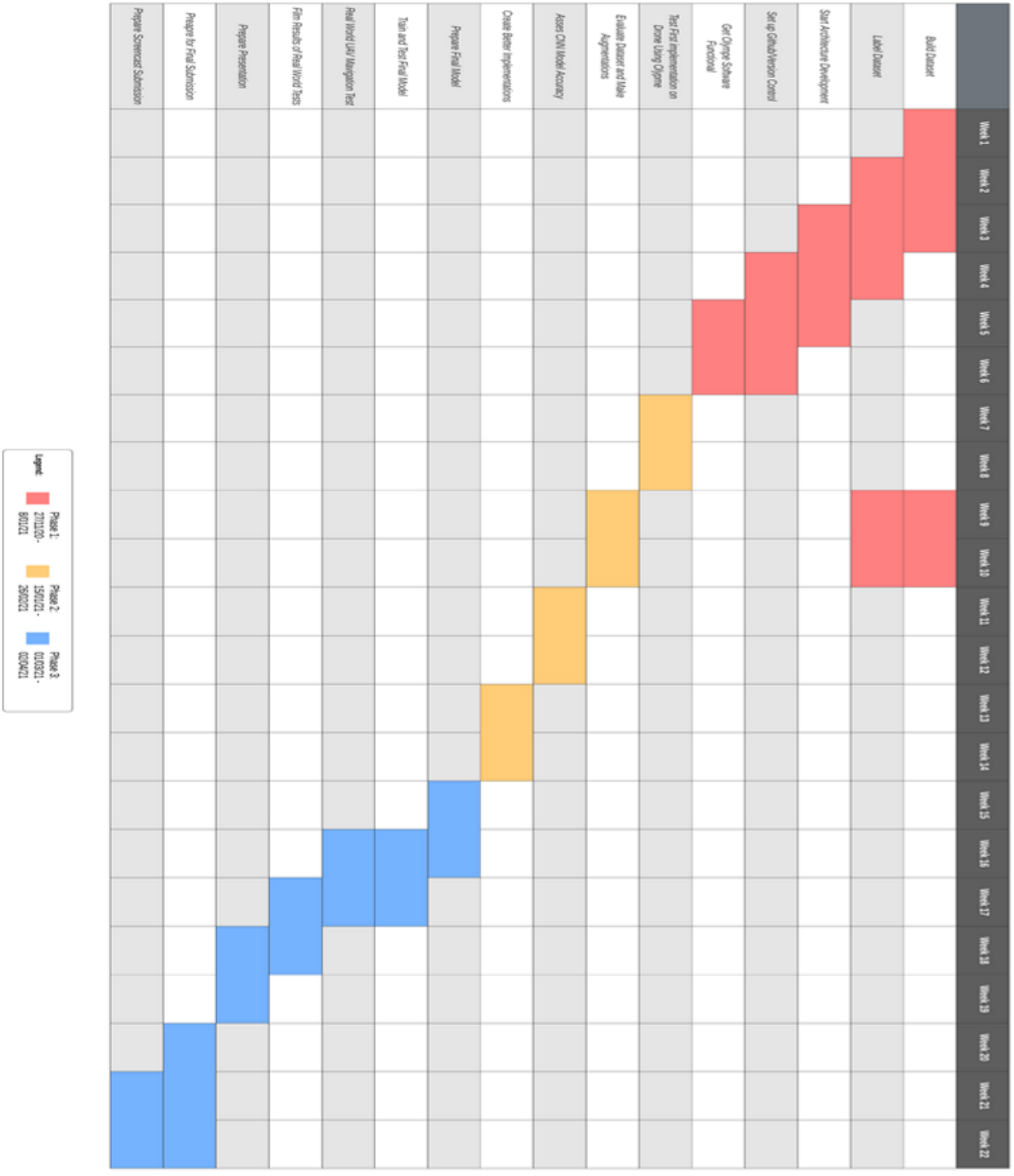


Figure 11: This is the operational flowchart for the whole system. These are the steps that must happen to complete navigation.

Timeline Gantt Chart



Timeline

Phase One

The first phase of this project will take place from November 27th to January 8th.

Work Scheduled

- Take footage of different roads, tracks and paths using the camera of a One Plus 7 pro and an Akaso Brave 7 action camera.
- Label frames from all the footage with 3 categories; left, right and straight.
- Develop the architecture needed and run it within the Sphinx simulator.
- Set up a Git repository for Version control and backups.
- Get the Olympe software fully functional in both Sphinx and on the Parrot Anafi.

Phase Two

The second phase of work will be carried out from January 15th to February 26th.

Work Scheduled

- An implementation of the dataset should be used on a CNN model developed so far. This would include training on the dataset.
- Process the results from training and evaluate the dataset.
- Assess the accuracy of the data set using the root mean squared algorithm and the explained variance ratio.
- Further develop a ResNet model and make changes where necessary.

Phase Three

The third and final phase of development will be carried out from March 1st to April 2nd.

Work Scheduled

- Prepare the final ResNet model.
- Train and then test the model on the dataset acquired for the final time.
- Test the UAVs navigational skill on different types of roads, tracks, and paths.
- Film the results of different tests to composite in reports and presentations.
- Create a presentation on the work achieved, improvements that could be made and anything that would be changed if the project were started again.
- Prepare Final report for submission.
- Create a screencast for submission.

Plagiarism Declaration

Declaration

- I declare that all material in this submission e.g. thesis/essay/project/assignment is entirely my/our own work except where duly acknowledged.
- I have cited the sources of all quotations, paraphrases, summaries of information, tables, diagrams or other material; including software and other electronic media in which intellectual property rights may reside.
- I have provided a complete bibliography of all works and sources used in the preparation of this submission.
- I understand that failure to comply with the Institute's regulations governing plagiarism constitutes a serious offense.

Student Name: Josh Hudziak

Student Number: C00231846

Signature: Josh Hudziak

Date: 30-04-21