Early forest fire detection using Computer Vision

Cristian Sorescu 879091

June 2019

Abstract

Wildfires have a huge impact on our society. It poses danger for the wildlife, agriculture, fire-sensitive forests and human lives every year. However, governments fail to recognise the severity of the issue and continue to use methods that have been used for centuries now [1]. Having technology advancing at such a great pace it is time to rethink and improve existing solutions. I will research, implement and evaluate an algorithm using infrared sensors, for object and fire detection and tracking. Even though there is a great number of implementations for object detection and tracking using daylight cameras, there are limited algorithms for InfraRed imagery. In the end, I will deploy the system on an unmanned aerial vehicle (UAV).

Project Dissertation submitted to Swansea University
in Partial Fulfilment for the Degree of Bachelor of Science

Department of Computer Science
Swansea University

Declaration

This work has not previously been accepted in substance for any degree and is not being currently submitted for any degree.

May 13, 2019

Signed: Cristian Sorescu

Statement 1

This dissertation is being submitted in partial fulfilment of the requirements for the degree of a BSc in Computer Science.

May 13, 2019

Signed: Cristian Sorescu

Statement 2

This dissertation is the result of my own independent work/investigation, except where otherwise stated. Other sources are specifically acknowledged by clear cross referencing to author, work, and pages using the bibliography/references. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure of this dissertation and the degree examination as a whole.

May 13, 2019

Signed: Cristian Sorescu

Statement 3

I hereby give consent for my dissertation to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

May 13, 2019

Signed: Cristian Sorescu

Contents

1 Introduction

The topic of this project is object and fire detection/tracking using electro-optical imaging in unmanned aerial vehicles (UAVs). This section will give a brief description of the existing issues with forest fires and the motivation behind this project. In addition, a description of the proposed idea and how it relates to the issue. Finally, a list of aims for this project.

Natural disasters have always been catastrophic occurrences. People lose their lives, either being a victim or a rescuer, land and families. More specific, wildfires, can last for days and sometimes weeks leaving everyone in a helpless state during that period. Every year there is irreversible damage to the agricultural land, wildlife, forests and people's lives. Living in a country prone to this issue, Greece, I get to witness this phenomenon every year but no major measures have been taken to improve the situation so far. We see on the news the disastrous results of wildfires all over the world but proactive planning for future incidents. Unfortunately, governments deny the scale and the seriousness of the problem and the fact that current approaches are not as effective anymore.

Wildfires today are acted upon as they happen with no prior proactive planning. Perhaps, new ways of dealing with the issue are needed. Specifically, the problem is when the **forest fires** escalate into wildfires, whether it was caused by a natural phenomenon or the human factor. On the other hand, not all fires are necessarily a bad occurrence, some are part of the natural lifecycle of a forest. But it is very important that we **monitor** the fires and keep them under control. Some cases of wildfires are results of a weather phenomenon, lightning, more specific dry lightning which is difficult to predict because of its nature [2]. However, there is the human factor as well, either directly or indirectly [1]. People use fire for various tasks, historically as a source of heat for either food preparation or warmth. Nonetheless, to a great extent, people use fire irresponsibly for the simple reason that it is a cheap and effortless land management tool. The use of technology will improve the situation at great scale. Zones of risks should be monitored during fire seasons to prevent escalation.

Nowadays the technology has progressed to a point where it is very easy and inexpensive to monitor the fires live. Namely, infrared cameras have not always been popular or accessible. Furthermore, infrared sensors have been mainly used in military applications. However, in recent years with increasing resolution and decreasing size and prices the IR sensors have established new application areas [3]. Moreover, UAVs, such as quadcopter drones, have never been available for everyday use until recently. And still, most of the commercially available ones, ready of the box, use proprietary technology

and restrict use in various ways. Those are mainly used for photography or videography. Even though technology is developing at such great pace, not everyone is taking advantage of that and we still fight the issue of forest fires with methods used decades ago, which, unfortunately, are not effective. Generally, forest fires are not monitored, and measures are taken when it is already too late. Weather services are researching new ways of weather prediction every day, using contemporary technology, but there is still a great amount of work ahead of them. And even if they succeed, the human factor still remains. Managing such great issues requires a joint effort of people and technology.

With this project, I aim to **build an object and fire detection/tracking system** based on Raspberry Pi and IR camera module. InfraRed imaging technology works by creating an image based on the difference of infrared radiation between the object and background highlighting the observed object [4]. Furthermore, compared with visible light imaging, IR imaging has the advantage of detecting the objects even when there is no light.

In the end, I intend to deploy the prototype system on a **quadcopter drone** as a proof of concept, to demonstrate how it can be used in real life situations. Moreover, the system's object detection/tracking feature can be used for collision avoidance. The system will communicate with the drone, overriding user commands, guiding it without crashing or causing additional harm to the surroundings. I will investigate multiple sensor technologies for the proposed system allowing it to detect danger and react quick enough to avoid it. A human controlling it may not be fast enough to identify a threat or miss it completely, because of limited visibility. Thus, an expensive piece of equipment may be lost, in fact, this is one of the issues with existing quadcopter drones. Such a drone could be used in harsh environmental conditions, forest fires, for monitoring. Usually, for wildfires, a helicopter would be used which is very expensive and still has limitations.

1.1 Aims

Namely, the **aims** of this project are the following:

- Research technologies used to monitor (detect and track) forest fires.
- Research infrared camera based fire detection methods.
- Implement and evaluate algorithms for fire detection using an IR sensor.
- Build a wireless physical prototype based on an existing embedded system, e.x raspberry pi board, and deploy it on a quadcopter drone.

2 Computer Vision

During CES 2015, the company Ascending Technologies demonstrated how a UAV equipped with Intel RealSense technologies, computer vision, could perform tasks such as playing a game without crashing into other entities around them. Later, at CES 2016, Intel CEO, Brian Krzanich, presented a collision avoidance system for commercial drones. [5] However, those drones use depth cameras which would be unusable for fire detection, as it uses infrared projection and fire does not reflect anything back for it to detect [6].

2.1 InfraRed and Daylight Cameras

There is a debate about how much resolution is needed for object detection using daylight cameras and identify what it is. On the other hand, the criteria for thermal cameras are predefined by Johnson's Criteria [7]. Although newer methodologies exist today, such as NVED's NVESD's "Night Vision Image Performance Model" (NV-IPM), Johnson's Criteria are still widely used in the security industry today [8]. Also, the two types of cameras operate in different spectrums of light. Daylight cameras are capable of capturing only visible light, whilst InfraRed cameras operate in the infra-red spectrum, see Figure 1.

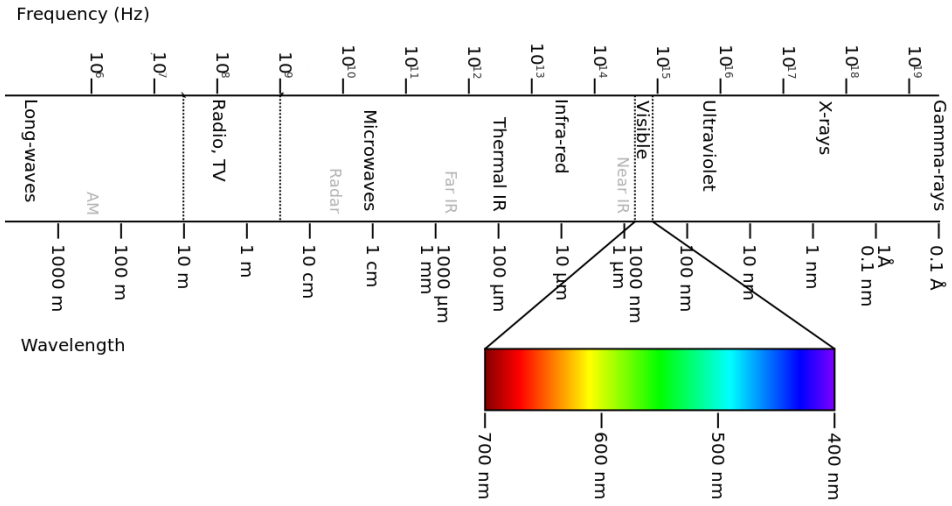


Figure 1: The electromagnetic spectrum. Modified [9]

2.1.1 Thermal Cameras

Figure 2: Picture from a thermal camera showing 5 individuals. [7]

John Johnson's criteria for object detection using thermal cameras are the following:

- **Detection**
  Ability to differentiate the entity from the background. "The ability to subtend 2 pixels, when using an LCD monitor, across the critical dimension of the subject. At the range that this occurs, regardless of the target type, the observer could detect that a subject was in the field of view, 50% of the time." [8]

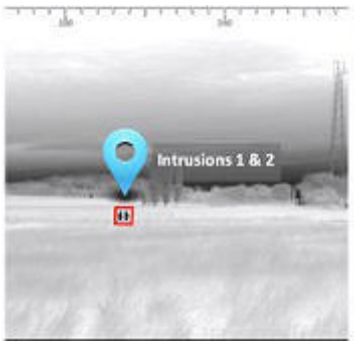


Figure 3: At a distance of several km away, 2 targets are distinguished from the background. [7]

- **Recognition**
  Ability to classify the object, human, vehicle, etc... "The ability to subtend 8 +/- 1 pixels, when using an LCD monitor, across the critical dimension of the subject. At the range that this occurs, regardless of the target type, the observer determines the type of subject, a human or a car, for example, 50% of the time." [8]

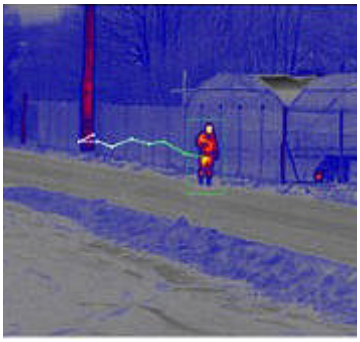


Figure 4: A human is walking along the fence. [7]

- **Identification**
  Ability to describe the observed entity. "The ability to subtend 12 +/- 3 pixels, when using an LCD monitor, across the critical dimension of the subject. At the range that this occurs, regardless of the target type, the observer could detect the subject." [8]

Figure 5: Two males with trousers and jackets are identified, one is smoking. [7]

2.1.2 Daylight Cameras

Optical cameras are also very important for firefighting. Those are very useful for large-scale mapping and post-burn impact assessment [10]. Nonetheless, we will not consider optical sensors because of their limitations.

2.1.3 Daylight Cameras without an infrared filter

These cameras are the middle ground between **Thermal Cameras** and **Daylight Cameras**. They provide some aspects from both ends. The image generated by such cameras is easy to interpret as there is not much change from what is seen with the naked eye. Furthermore, it also captures heat, as tints of colours, in warm weather the captured image has a slight red tint

and in cold weather a blue tint. Furthermore, it captures the infrared light that cannot be seen with the naked eye.

A notable example is the Raspberry Pi NoIR Camera module V2. The module has a Sony IMX219 8-megapixel sensor, it has the same specifications as the normal Pi Camera Module, but slightly differs, there is no infrared filter. The module is popular with wildlife enthusiasts [11], especially for night-time observations. A picture of the camera module is provided in section 6.3.1, Figure 13.

2.2 Object and Fire Detection

A fire from a computer's point of view can be considered an object that meets specific criteria. Thus, object detection is an important step before fire detection. First, we determine what objects represent and how they are defined. And only then we can define what a fire is. It works as inheritance, fire is a type of an object with additional properties.

2.2.1 Object Detection

Object detection is a hot topic in the Computer Vision field. There are numerous implementations, including the Open Computer Vision library, which consists of a large number of models. Specifically, the OpenCV library is using the Haar Feature-based Cascade Classifier for Object Detection approach. This model is creating a cascade of boosted classifiers working with haar-like features. The top classifier consists of multiple simpler classifiers that are being applied to the specified area. The classier is trained with hundreds of samples, scaled to the same size, called positive examples, and of arbitrary size, called negative examples. Subsequently, once the classifier is trained, it can be applied to the region of interest. It returns 1 if the region is likely to contain the desired entity or 0 if not. [12] The current OpenCV implementation uses the following haar features:

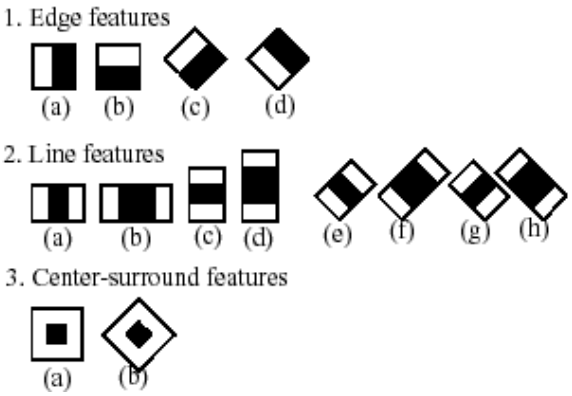


Figure 6: Haar features. [12]

2.2.2 Signatures and Algorithms for Fire Detection

As fires burn they emit **heat** and it provides a strong signal for detection. Generally, heat is transported from the fire by conduction, convection and radiation, but the radiated heat is the sense that is remotely sensed. "The emission of a blackbody radiator at thermal equilibrium follows Planck's law, which describes the spectral radiance distribution as a function of wavelength of the emitted electromagnetic radiation. The total energy (integrated across all wavelengths) radiated from a blackbody surface increases rapidly with its temperature (proportional to the fourth power of temperature, T^4, as described by the Stefan-Boltzmann law." [10]

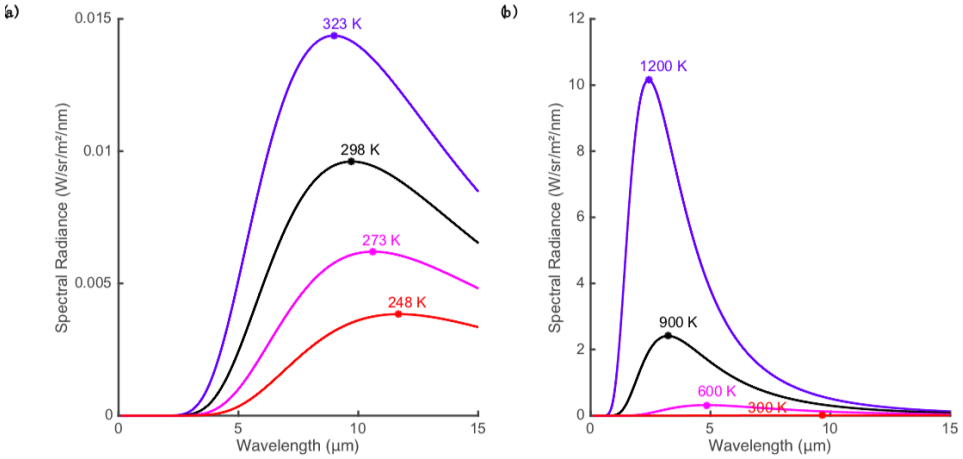


Figure 7: (a) Plank's law curves for ideal blackbody radiation at moderate temperatures (-25 to +50°C); (b) Higher temperature sources radiate much more intensely (note the large change in scale) and peak spectral response is shifted toward smaller wavelengths. [10]

Today, most imaging techniques to detect heat signatures of fire, tend to use (Mid-wave InfraRed) MWIR and (Thermal InfraRed) TIR sensors. Whilst both have their advantages and disadvantages, the TIR sensors have a critical advantage, for forest fire monitoring, of detecting hot spots even through thick smoke. Also, TIR can image fire and the background without saturating the sensor. [10] On a bigger scale, space agencies use algorithms like (Satellite Fire Detection) SFIDE Algorithm in their satellites for early fire detection.

There are plenty of open source implemented algorithms for fire detection analysing the visible light spectrum, such as [13]. On the other hand, to the best of my knowledge, there are, almost, no widely available implemented fire detection algorithms base on IR imaging.

3 Fire Detection and Tracking using InfraRed Cameras

In recent years, there has been a great advance in the field of object detection and tracking using daylight cameras. And even though there is not as much research done on InfraRed camera-based object detection and tracking, there are a few noticeable papers. An even less common research topic is fire detection using IR cameras in UAVs.

3.1 Object detection and tracking

Object detection has always been one of the main topics in the computer vision field. There is a great amount of widely available, powerful, trained classifiers for object detection and tracking.

In 2013, Frederik Stendahl Leira submitted a paper, partial fulfilment of the requirements for the MSc degree at the Norwegian University of Science and Technology, on Infrared Object Detection & Tracking in UAVs which will be used for my object detection research. Frederik Stendahl Leira used pre-trained classifiers for object detection and an estimate-and-measure algorithm for tracking. Later two tracking classifiers have been trained. However, it was found that linear estimator would struggle to keep track of the object and a non-linear one would benefit the system. Furthermore, the system was able to simulate control over the UAV. [14]

In 2014, at the IEEE International Conference on Mechatronics and Automation a group of researchers, Zhao Wang, Haitao Song, Han Xiao, Wenhao He, Jiaojiao Gu and Kui Yuan, published a paper on real-time object detection based on infrared image which I will analyse for the applied

methodologies. The researchers worked with an existing, efficient, object detection algorithm customised for the needs of their system. The system was tested in experiments. Firstly, they used a low-cost camera to test how effective the system was. And lastly, a simulation scenario was used to measure the performance of the embedded system. [4]

3.2 Fire detection and tracking

Fire detection has been a hot topic for decades, whether it was used for indoors or outdoors. With the InfraRed cameras becoming more accessible and smaller a great interest sparked in the market. More companies use them to improve their technologies and more research is being done.

In 2011, a group of researchers, Bosch, Ignacio & Gómez, S & Vergara, Luis, published a paper early forest fire detection based on InfraRed signal processing which I will be using for my case study. They developed a ground remote automatic system for forest fires monitoring. Advanced techniques have been used for IR signal processing. Also, they have evaluated false alarms based on a persistence detector and increase detector they have developed. Experiments have been conducted to measure the delay in detection. The researchers have achieved a 100% detection probability at a false detection rate of 10^{-9}. [15]

In 2016, Robert S. Allison, Joshua M. Johnston, Gregory Craig and Sion Jennings published a review paper on airborne optical and thermal remote fire monitoring for wildfires. I will be using it for my fire detection case study. Particularly, they have investigated the use of InfraRed sensors on small UAVs, such as drones. They concluded that deploying small sensing platforms for continuous monitoring could enable critical improvements for fire suppression. However, they mentioned that more robust automatic algorithms are needed before such a system can be deployed. [10]

3.3 Drones Assisting Firefighting

As technology advances and drones are more and more widely available, scientists are researching different ways of using drones to assist humans with important tasks. Firefighting is a dangerous job where everyone is at risk even the firefighters. Firefighters would benefit a lot from drones. Drones can offer a lot of help, perhaps even save lives.

In May 2019 Md, Nafiz Hasan Khan and Carman Neustaedter have reported their findings while researching the use of drones as firefighting assistance in their paper "An Exploratory Study of the Use of Drones for Assisting Firefighters During Emergency Situations". They have reported

that emergency service 9-1-1 will get new technologies. Drones will be one of them. In order to fully understand how drones will affect the service, they conducted a study with citizens who were reaching out to the 9-1-1 service and firefighters. The research has shown that drones numerous benefits for both sides, callers and firefighters. People did not seem to be concerned with the privacy aspect of a drone network. However, the safety was questioned. [16]

4 Project Management

4.1 Methodology

I have chosen to use the agile software lifecycle (SLC) model for this project. In contrast to the more usual waterfall model which requires you to have a pre-planned design before proceeding any further and requires a tremendous amount of rework in case of changes, sometimes not allowing changes at all, the agile model is prone to changes, which is a critical feature when building software and hardware together. Also, I will apply the build research methodology in this project as will be building an artefact as proof of concept. A Raspberry Pi single board computer and an InfraRed camera seem to be most suitable, non-proprietary, widely available components to build a system for object and fire detection/tracking. Furthermore, I will be using a combination of C, C++ and possibly Python as the main programming languages. Software-wise, the object detection functions from the OpenCV library [12] will be used as a starting point.

4.2 Fire Detection Techniques

The Open Computer Vision library, object detection model, is based on the **Haar Feature-based Cascade Classifier for Object Detection**. "First, a classifier (namely a cascade of boosted classifiers working with haar-like features) is trained with a few hundred sample views of a particular object (i.e., a face or a car), called positive examples, that are scaled to the same size (say, 20x20), and negative examples - arbitrary images of the same size. After a classifier is trained, it can be applied to a region of interest (of the same size as used during the training) in an input image. The classifier outputs a "1" if the region is likely to show the object (i.e., face/car), and "0" otherwise.". Also, the main classifier is consisting of multiple sub-classifiers. [12]

Fire is dynamic by its nature and changes from frame to frame. An efficient algorithm is needed for individual component analysis. According to [17] the **Geometrical Independent Component Analysis (GICA) model** has been found to show the best performance for fire detection in

comparison with the other tested algorithms.

$$\begin{pmatrix} X_1^{(1)} & (\mathrm{t}) \\ X_2^{(1)} & (\mathrm{t}) \end{pmatrix} = \begin{pmatrix} 1 & \alpha \\ \beta & 1 \end{pmatrix} \begin{pmatrix} X_1^{(0)} & (\mathrm{t}) \\ X_2^{(0)} & (\mathrm{t}) \end{pmatrix} \tag{1}$$

Other algorithms, such as Gaussian Mixture Model, RGB and HSI colour models, were considered but these show slow performance and false fire detection. [17]

For fire detection using an IR camera, I will mainly consider TIR imaging. In addition, I will be experimenting with the matching pursuit algorithm based on adaptive decomposition. The algorithm produces three important pieces of information, the set of decomposition coefficients α, the set of residues RI, and a list of dictionary elements chosen to approximate of $I(x, y), g_\gamma$. Using these factors we can completely define the image $I(x, y)$. Their energies are given by the formula below. [18]

$$E = ||\alpha|| + ||RI|| + ||g_\gamma|| \tag{2}$$

Also, I will consider an improved version of the (Satellite Fire Detection) SFIDE Algorithm. The original SFIDE algorithm exploits a geostationary satellite sensor (SEVIRI, Spinning Enhanced Visible and InfraRed Imager, on board of MSG, Meteosat Second Generation, satellite series). [19]

Conclusion: I have decided to use the already existing library OpenCV which is very modular. A Haar-feature Cascade Classifier can be trained to recognise fires and the applied on any given image. Furthermore, it works the best with Daylight Cameras without an infrared filter type of camera which has been used in this project.

5 System Overview

5.1 Project requirements and specifications

The main components **required** for the proposed system are the following:

- **Processing unit/brains**
  It should be capable of powering all dependent modules, communicate with the control station, log data on local storage and perform intense processing for object and fire detection. I will be using a Raspberry Pi single-board computer.

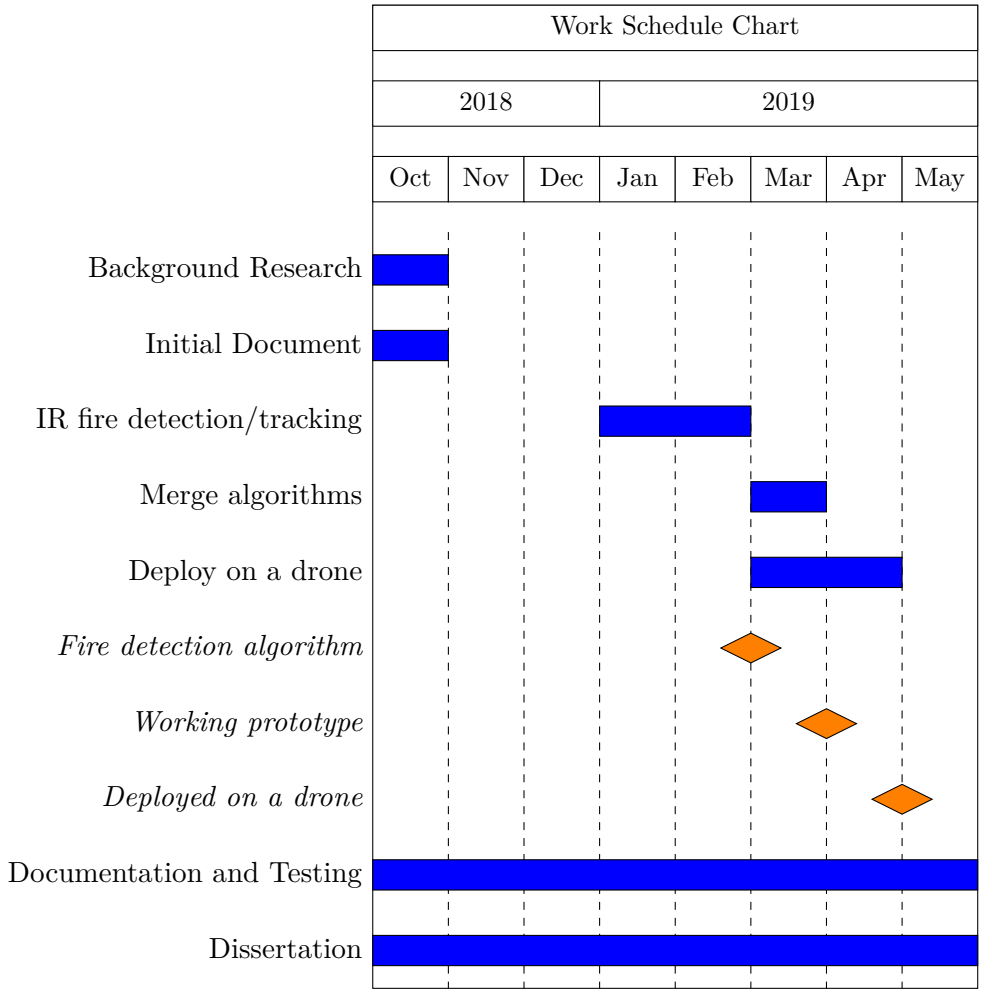
- **Camera module/sensor**
  The camera module should be compact and have a reasonable frame rate.

- **Electric Power Source**
  Battery module should be compact and able to supply stably the system with electric power.
- **Communication Module**
  Wireless communication module should have a long range of communication and be able to stream data while the system is in operation.
- **UAV**
  A UAV, namely a quadcopter drone, for system deployment. Should be able to carry the system.

I propose the use of a system with an IR sensor for fire detection/tracking which can be deployed on existing UAVs. An InfraRed Sensor on a Raspberry Pi single-board computer will allow real-time data processing for fire detection/tracking. Doing this will reduce the involvement of firefighters in dangerous operations and get faster more accurate results.

5.2 Project planning

I have mainly followed the schedule presented in the Gantt Chart below. I have achieved every milestone that has been set and completed all tasks in time.



Main Points in my project planning.

- **Background Research, 2018-10-01 to 2018-10-31**
  By the end of October 2018, I plan to have done most of my background research and start working on the project.
- **Initial Document, 2018-10-01 to 2018-10-30**
  The deadline for the Initial Document is on 1st of November 2018. I should have it done a day or two before the deadline to allow time for review and corrections.
- **IR fire detection/tracking, 2019-01-01 to 2019-02-27**
  Once the IR object detection algorithm is ready or allowed timeframe starts, 1st of January, 2019, I will start working on the fire detection part. Again, two months to allow for further research and tests. However, if I finish sooner than the dedicated time I will proceed to the next part of the project.

- **Deploy on a drone, 2019-03-01 to 2019-04-30**
  Finally, while I will be merging the two algorithms, I will start deploying the system prototype on a quadcopter drone.
- **Documentation and Testing, 2018-10-01 to 2019-05-28**
  As I will be working on the project, I will document all the stages, either locally or online, and run tests periodically to ensure the stability of the system.
- **Dissertation, 2018-10-01 to 2019-05-28**
  In addition, I will be writing the dissertation as I progress with the project.

The milestones for my project are as follows.

- **Object detection algorithm, 2018-12-31**
  By the end of December 2018, I intend to finish any object detection and tracking research and have the IR object detection/tracking algorithm implemented. Thus, allowing enough time for the next stage in the project.
- **Fire detection algorithm, 2019-02-27**
  In February 2019, I plan to have finished research related to the fire detection/tracking and the algorithm implemented.
- **Merged the two algorithms, 2019-03-30**
  In March 2019, I should have the two algorithms work together.
- **A working prototype of the system (no drone), 2019-03-30**
  In March 2019, before merging the two algorithms and deploying the system on a drone I will have a working prototype of the system.
- **Deployed on a drone, 2019-04-30**
  Finally, in April 2019, I should have the system deployed on a quadcopter drone and ready for demonstration.

5.3 Risk analysis

Developing a system involving software and hardware doubles the risks of the project. In addition to software failures, there is the factor of hardware failure that could cause harm or damage. Specifically, there are technical and personal limitations.

Risk	Mitigation Strategy	Impact	Likelihood	Risk Score
Calculating the distance to an object using an IR camera.	Use a second camera.	8	10	80
Using batteries as the main power source.	Shield the battery, regular system backups.	10	8	80
The InfraRed camera module, range, frame rate and quality.	Find camera modules that will meet the requirements. Possibly use a second camera.	6	8	48
Wireless communication, range, and interferences.	Research available wireless modules, and previous successes using them.	6	5	30
Get injured.	Wear protective gear. Approach the dangerous aspects of the project in a serious and careful manner.	5	2	10
Lose or break my equipment.	Have equipment in one place, make regular system backups, and fly the drone in a safe environment.	10	3	30
Losing all of my data.	Upload the data online on a cloud service and make regular backups on external drives.	10	5	50

5.3.1 Technical risks

Technical risks I may encounter while working on this project.

Calculating the distance to an object using an IR camera is a challenging topic. Without using a depth camera, it is hard to estimate how far away is an object in a picture/frame. To make this feature possible additional research will be needed and possibly a second camera.

Using **batteries** as the main power source for the entire system can be dangerous on its own as they are not protected in any way. Dropping them could cause life hazards, such as fire, smoke or even outburst of chemical substances located inside the batteries. Furthermore, if they fail during testing it could cause modules to fail or file system corruption leading to irreversible damage. I will take measures to avoid this, some of which are, shield the battery to protect it from a direct impact upon drop and regular system backups.

Having to analyse live data it would require a good **InfraRed camera module**. But even then, there are certain limitations, some of which are, visibility, range and quality. Also, transmitting live video over a wireless connection could be challenging because of the signal interference. Furthermore, inspecting fire up close could cause permanent damage to the module and a new one may be required. To avoid the aforementioned issues, I will thoroughly search for camera modules that would be suitable for this project and use them with caution to avoid any damage. However, if object and fire detection are not possible with a single camera, I will have to consider using a second IR camera.

It may be found that **wireless communication** may not be stable. As the system will be deployed on a UAV, there needs to be a strong connection between the ground controller and the system. A wireless signal is often interrupted by obstacles or other interfering signals. To circumvent this, I will carefully research available wireless modules for the specified communication mean and previous successes using them.

5.3.2 Personal risks

Personal risks I may encounter while working on this project.

I could get electrocuted or **injured** while working with the hardware components. Working with hardware which involves electricity can be very dangerous and needs additional attention. Getting injured can cause unplanned delays with the development and missed deadlines. I will wear pro-

tective gear while working on the hardware part of the project and approach the work in a serious and careful manner.

I could accidentally **lose or break** my equipment. Having to move equipment around for lab testing or demonstration I could damage it or lose it if not packed carefully. In addition, quadcopter drones sometimes are unpredictable and could fly into an obstacle damaging itself permanently. Moreover, in case of a software error, data may end in a corrupt state, meaning that all prior work is lost. To confront the above-stated concerns, I will make sure not to move equipment around if not necessary, make regular system backups and fly the drone in safe environments.

There is a risk of **losing my laptop**, thus losing all my data. Computers tend to break a lot, either because of software or hardware error. In some cases, data is not affected, but often times, everything is lost. There is software for file recovery, however, it will not restore 100% of the lost data. To prevent the above-stated issues, I will upload the data online on a cloud service, such as iCloud and GitHub, and make regular backups.

6 Implementation

6.1 Initial Approach: Arduino as the Main Board

My initial plan was to use an Arduino micro-controller as it would allow me to build a minimal and lightweight system. The Arduinos by their nature are quite small and do not require lots of power to operate, which would be ideal because the system is intended to be wireless. The less power the system consumes the more it would be able to operate without recharging. However, the Arduino micro-controllers presented numerous limitations, such as no parallel modules communication supported and inability to process graphical data in a suitable timeframe.

6.1.1 Hardware

I have considered Arduino single-board micro-controllers, Uno and Nano, as the main board for this project.

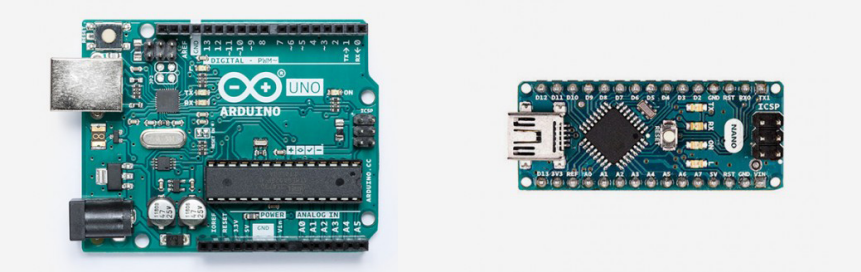


Figure 8: On the left is the Arduino Uno. On the right is Arduino Nano. Source: `https://store.arduino.cc/arduino-uno-rev3`

Also, the HM-10 BLE Bluetooth 4.0 CC2540 CC2541 Serial Wireless Module [20] and HC-12 433Mhz SI4463 Wireless Serial Port Module [21] were considered for wireless communication with the system.

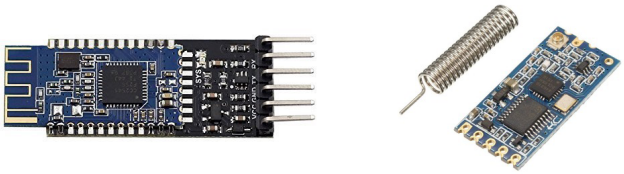


Figure 9: On the left is the HM-10 HM-10 BLE Bluetooth 4.0 module. On the right is the HC-12 Wireless Serial Port Module. Source, left: `http://fab.cba.mit.edu/classes/863.15/doc/tutorials/programming/bluetooth.html`, right: `https://www.alibaba.com/product-detail/HC-12-SI4463-433Mhz-Wireless-Module_60557524362.html`

6.1.2 Algorithm

Furthermore, I have written some testing code for wireless communication using the above-stated modules.

Figure 10: Diagram of the communication system using HM-10 and HC-12 modules.

```
#include <SoftwareSerial.h>

SoftwareSerial HC12(10, 11); // HC-12 TX Pin, HC-12 RX Pin
SoftwareSerial HM10(7, 8); // TX, RX

void setup() {
  // Open serial communications
  Serial.begin(9600);
  // Start each software serial port
  HM10.begin(9600);
  HC12.begin(9600);
}

void loop() {

  HM10.listen();

  while (HM10.available() > 0) {
    char inByte = HM10.read();
    Serial.write(inByte);
  }
}
```

Listing 1: HM-10 to HC-12 Arduino Code

6.2 Conclusion

However, because of serial port limitations on the arduinos used for testing, I was unable to get the two modules to work at the same time. Also, the processing power is not enough for the set tasks, InfraRed imagery analysis. Taking into consideration the aforementioned, I decided to use a raspberry pi as the main board for this project.

6.3 Final Approach: Raspberry Pi 3 Model B as the Main Board

Soon I realised that the Initial Approach: Arduino as the Main Board does not satisfy the project requirements. I started researching other, more powerful, single board micro-controllers and computers. Among the numerous options found on the internet, the **raspberry pi 3 model b** seemed to be a suitable candidate. The raspberry pi 3 was of acceptable size and quite cheap in relation to the other options. I started exploring it and got myself familiar with the operating system it runs. I set up my workspace on the raspberry pi 3 and started working on the project.

In addition, I needed a camera for this system as stated in Project requirements and specifications. Initially, I was considering two different types of cameras, Raspberry Pi camera modules which has a specific connector and USB cameras which are more universal. After careful consideration in Daylight Cameras without an infrared filter, I decided that the Raspberry Pi NoIR would be a good choice for this project. The complete system architecture is presented in Figure 11.

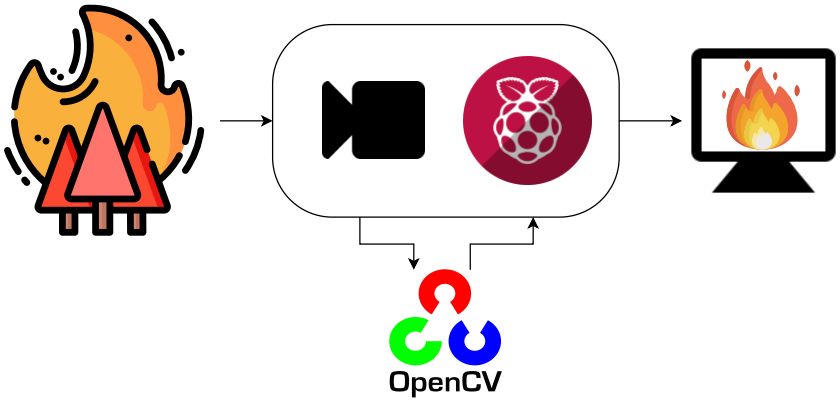


Figure 11: System Architecture.

6.3.1 Hardware

Once the software environment was set up it was time to start the assembly of the hardware aspect of the project by connecting the camera module, Figure 13, to the main board, Figure 12. As I was assembling the system I realised that the camera module has no support whatsoever, it was just hanging in the air.

Figure 12:
Raspberry Pi 3 Model B

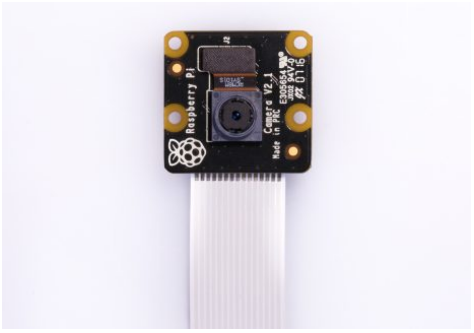


Figure 13: Infrared Camera Module v2 (Pi NoIR).

At that point I decided that I need an enclosure, a container so to speak, that would contain the system together as one piece. I started building one out of cardboard as it is easy to manipulate and cut. I took measurements of the Raspberry Pi and cut the base first. After I cut-out some holes for the plastic screws that are provided with the board, I attached it to the first segment of the enclosure. Next were the long sides walls, one of them required a few cut-outs for the HDMI, audio and micro-usb ports. Then I glued the segments together, see Figure 14.

Following segment of the enclosure was the top cover. I planned to have the camera in it so a cut-out was needed. This time the cut-out had to be more complex, a smaller middle hole for the camera sensor and a bigger, but not full, cut for the chip. Also, the second section of the cut had to hold the chip in place as well, so the cut was a bit smaller than the chip such that the edges could slide in between the cardboard layers, see Figure 15.

Figure 14: Raspberry Pi 3 Model B: Process of building the enclosure.

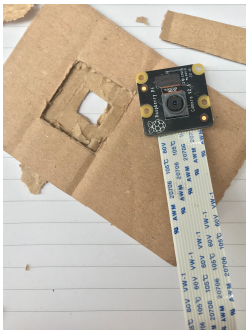


Figure 15: Enclosure wall for the Infrared Camera Module v2 (Pi NoIR).

Lastly, the remaining parts were the short sides walls. One of them re-

quired a big cut-out in the middle for the USB ports and Ethernet port. On the other side, some adjustments were made to the wall such that the sdcard is accessible easily. In addition, a transparent piece of plastic was used to boost the light coming from the Raspberry Pi LED indicators, see Figure 16. At this point, all that was left to do was to assemble the enclosure, in other words, put it all together, see Figure 17.

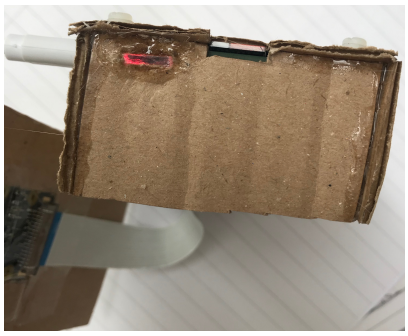


Figure 16: Assembly of the enclosure.

Figure 17: Final product.

6.3.2 Obtaining Visual Data

One of the main requirements of this project is to provide feedback efficiently based on the acquired visual data. Therefore, the entire process, including visual data acquisition, should be as efficient as possible. There are two ways of obtaining and displaying live, Figure 18, data from the Raspberry Pi camera module in python that interest us.

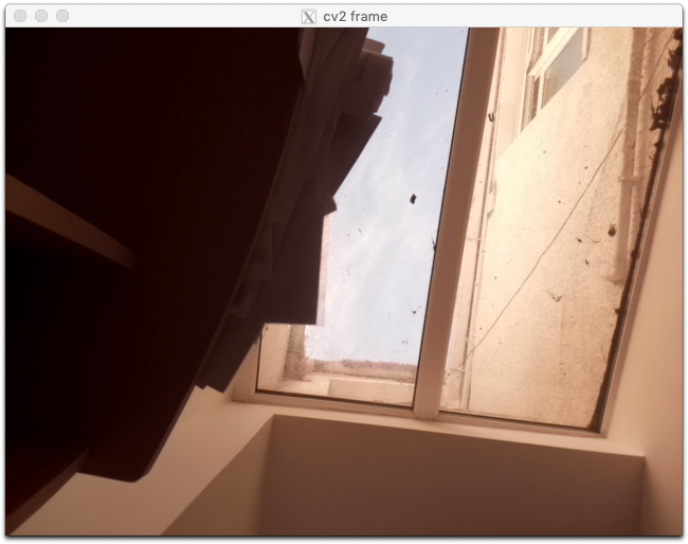


Figure 18: Streaming live data from the Raspberry Pi camera module using the OpenCV library.

First, the default Raspberry Camera library, *picamera*. The Pi Camera library in Python allows streaming only on a display that is physically connected via the HDMI port, which violates the rule of being wireless. Thus default *picamera* library is not suitable for this project, as one of the aims is to be wireless.

Second, the $VideoCapture()$ function from the OpenCV library. The issue observed above while using the *picamera* library can be resolved by using the improved version of the $VideoCapture()$ function from the OpenCV library. The $VideoCapture()$ function makes it possible to stream the camera module feed in a window on any display, either physical or virtual. This way, the camera feed can be streamed over *ssh* or *tcp* using such tools as XQuartz and Xpra, see Figure 18 and 19.

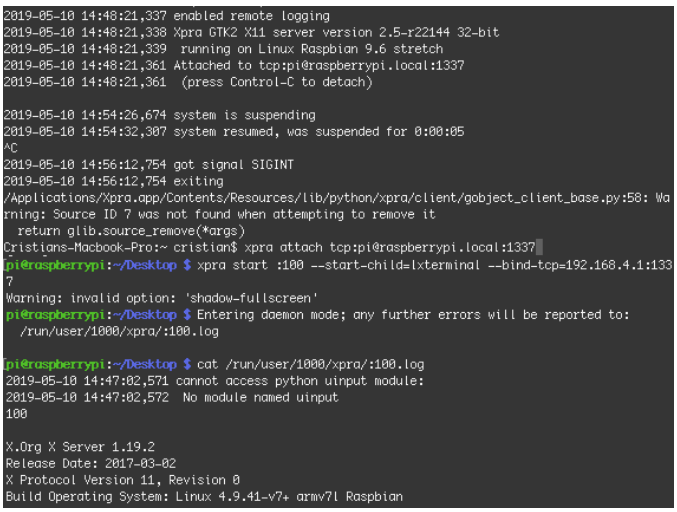


Figure 19: Client/Server Xpra.

Later I have found an improved version of the $VideoCapture()$ function called $VideoCaptureAsync()$. The $VideoCaptureAsync()$ function is using Python threads library to maximise the performance of frame acquisition from the Raspberry Pi camera module. A comparison of the VideoCapture() and $VideoCaptureAsync()$ functions, in terms of time, is presented in Figure 20.

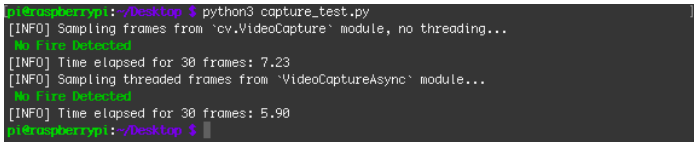


Figure 20: Comparison of time taken to process 30 frames using $VideoCapture()$ and $VideoCaptureAsync()$ functions.

As we can see there is a significant improvement of 1.33 seconds. Even though it seems like an insignificant improvement, it is very crucial for a system like this where efficiency is of high importance.

The latter method seemed more appropriate and was used for this project. To acquire the data from the camera device, one needs to create a $Video-Capture$ object, Line 2 in Listing 2. After the created object can be used to capture frames, one at a time using the object's $read()$ function, Line 8 in Listing 2. In other words, the data is obtained as frames, a $while(true)$, also known as infinite, loop runs and requests a single frame at a time, procedure given in Listing 2.

```
1  # begin capturing video
2  cap = VideoCaptureAsync(0)
3  ...
4
5  # operate on live feed
6  while(True):
7      # Capture frame-by-frame
8      ret, frame = cap.read()
9      ...
```

Listing 2: Obtain Visual Data

6.3.3 Initial Algorithm: Fire Detection Based on Colour

As a starting point, I decided that fire can be recognised by colour. I started researching this idea and discovered some existing implementations. One of the implementations I found was called Fire Detection using Computer Vision. [22] This project was very close to what I was doing. The author used OpenCV to obtain and process images of fire. Furthermore, there was a video demonstration and the results were very promising.

The algorithm that the author came up with was converting the images from RGB to HSV colour space. Also, Gaussian Blur is applied to the image to remove noises. Then, $upper = [35, 255, 255]$ and $lower = [18, 50, 50]$ limits

are defined, any colour in the range between *lower* and *upper* is examined and displayed to the screen, everything else is black. In other words, everything that has a colour from the defined range is a fire. Flowchart of the algorithm is presented in Figure 21.

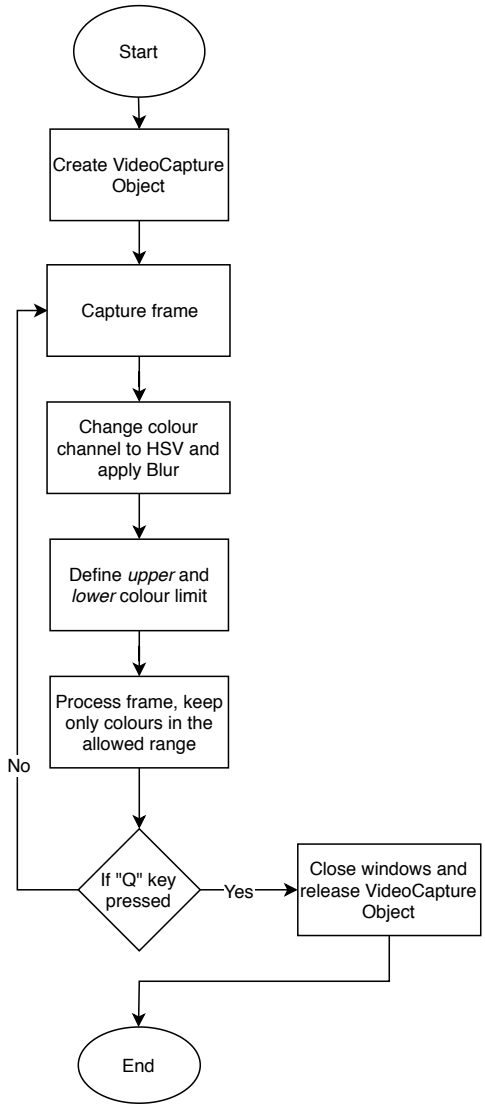


Figure 21: Flowchart of the algorithm presented in [22].

In order to check the promised results, I started experimenting with it myself. First thing was to check if it can detect fire. Also, check for false positives, a lot of objects have colours similar to the ones of fire, in the defined range. The algorithm failed straight away when presented with a small fire from a lighter, see Figure 22 and 23. Perhaps, this happened because

the colours are slightly different since the camera has the infrared filter removed. The difference in colours is an important factor as the algorithm was designed to work with regular cameras.

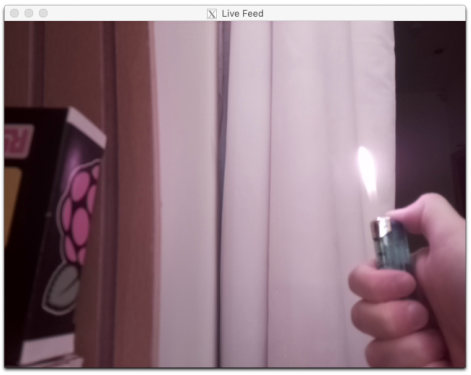


Figure 22: Image from camera module.

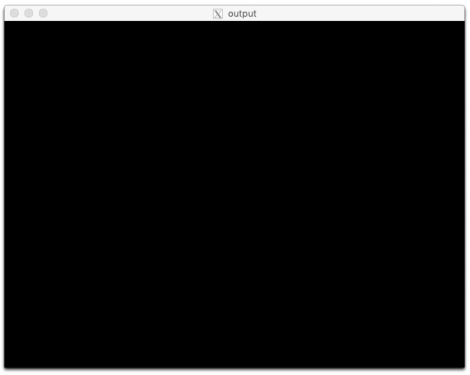


Figure 23: Image after it has been processed.

Even though it failed to detect the fire, it succeeded detecting a box of colour defined in the allowed range. I used a boxed that was of bright red colour, see Figure 24. This result is quite the opposite of what was expected.

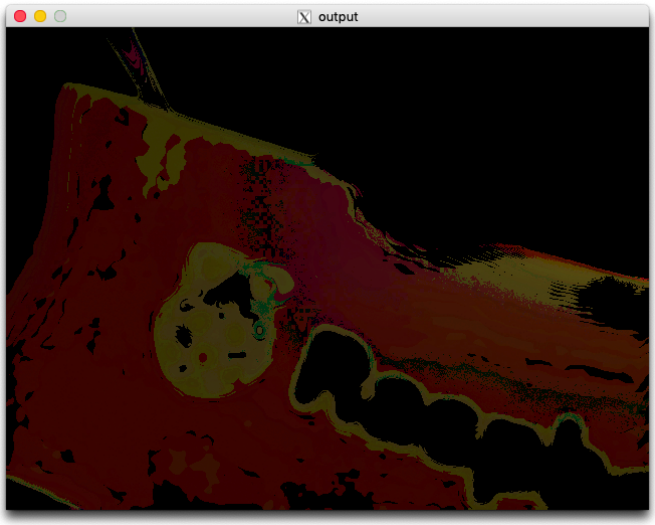


Figure 24: Algorithm recognised red box as fire.

Conclusion: This algorithm is far from ideal, even a step backwards if it is used with a non-ordinary camera. It does the opposite of what it is supposed to do, detect fire. It has a very high false positive rate and it fails to recognise actual fire. In addition, while testing this specific algorithm I

realised that building a system that would detect fire based on colour is not feasible, no matter how good it is there will always be a high rate of false positives, any object in a given colour range.

6.3.4 Final Algorithm: Fire Detection Using a Pre-trained Fire Classifier

One last attempt to achieve the desired outcome. Since the previous approach did not succeed, I started researching other methodologies and decided to use the full potential of the OpenCV library. In this section, I will explore the machine learning aspect of OpenCV, specifically trained classifiers to detect fire.

I have already researched how cascade classifiers are defined in the Object Detection section. I had tow options, train my own classifier which it would involve finding large image sets of specific dimensions or finding a classifier that someone else trained. Luckily, I was able to find a trained classifier which would be able to identify fires. Then I started writing the Python program, a flowchart of the algorithm is presented in Figure 25.

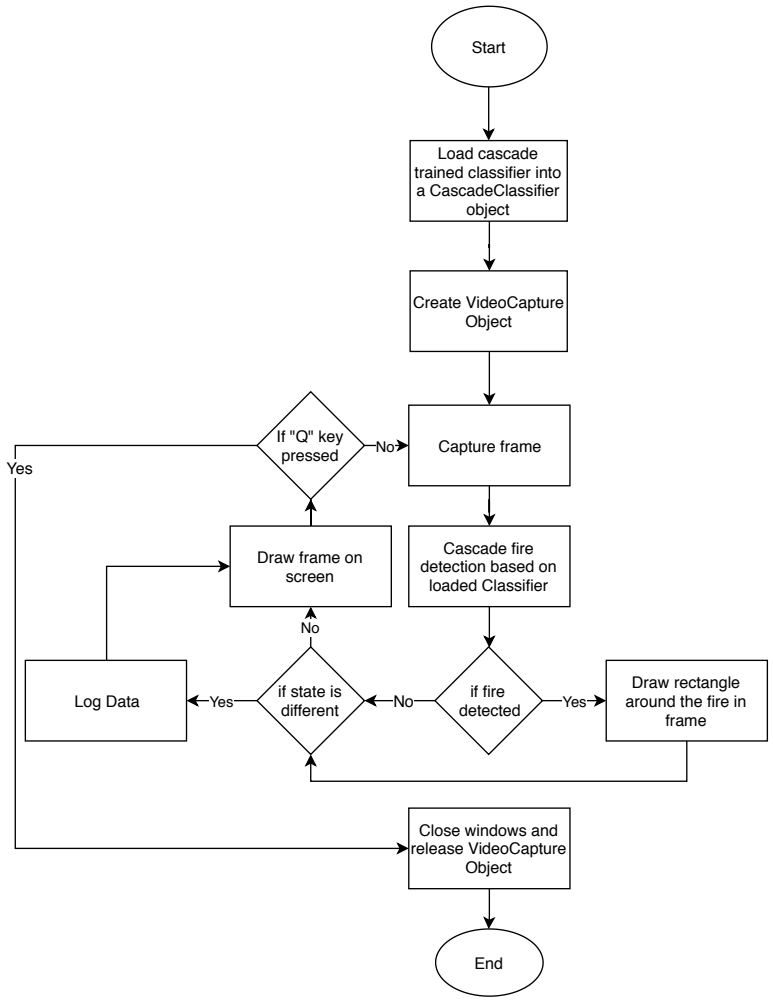


Figure 25: Flowchart of the algorithm presented in the Final Algorithm: Fire Detection Using a Pre-trained Fire Classifier section.

Fire Detection: Haar-feature based cascade classifiers are similar to Convolutional Neural Networks, the Haar-feature is similar to the kernel in the later. There are several Haar-features defined in the pre-trained classifier that can identify a fire in an image. The pre-trained classifier is a large XML file containing weightings for each Haar-feature.

To detect fire in a frame, the pre-trained classifier needs to be loaded into a Cascade Classifier object. Then, the program captures frame-by-frame is processing each individually. Once a frame is acquired the $detectMultiScale()$ function is used to analyse the frame. One of the arguments passed to the $detectMultiScale()$ is the classifier object and the frame itself. Once the function is done, it returns an array of detected objects, fires, in the frame, if any. If the array contains any elements, then for each entry pixel data are

analysed and a rectangle is drawn around the detected fire in the frame, see Figure 26 and 27. Then the frame is presented to the user.

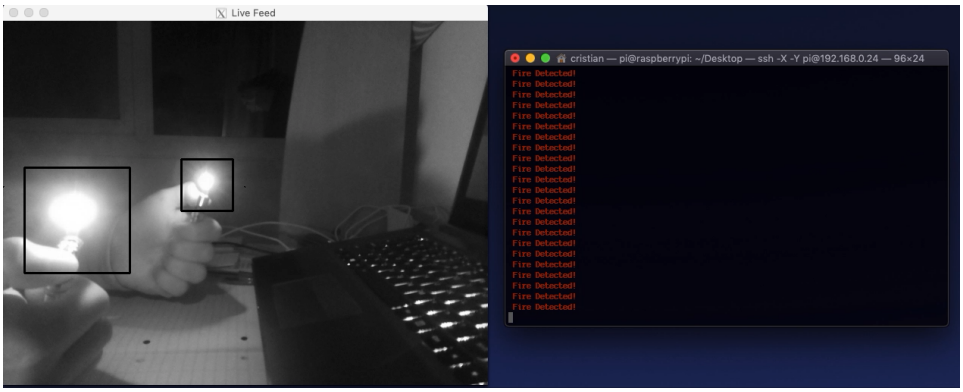


Figure 26: Fire detected during night-time: live feed and console feedback.

Camera was covered with a blue light filter to further reduce noise and enhance fire detection. Thus, colours seem less vibrant, see Figure 27 however that is not of interest.

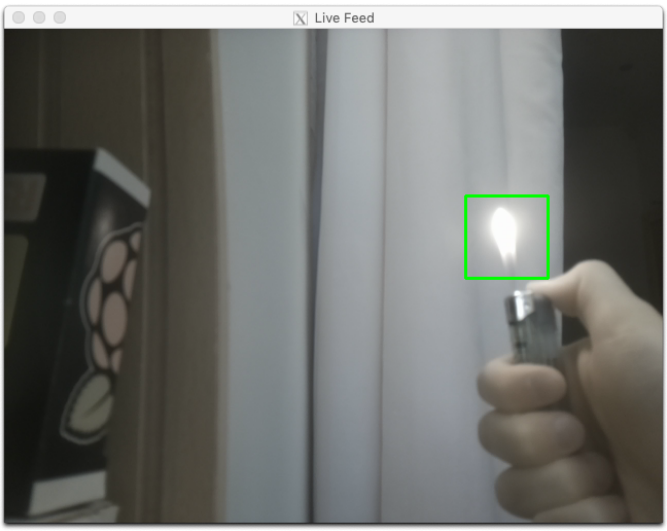


Figure 27: Fire detected using the blue light filter.

Log: The system in addition to the live video feedback, is equipped with text logging. There is live feedback in the console stating whether a fire was detected or not. In addition, for every detected fire a snapshot is taken and save on the disk together with the log from the console. Example output is

presented in Figure 28.

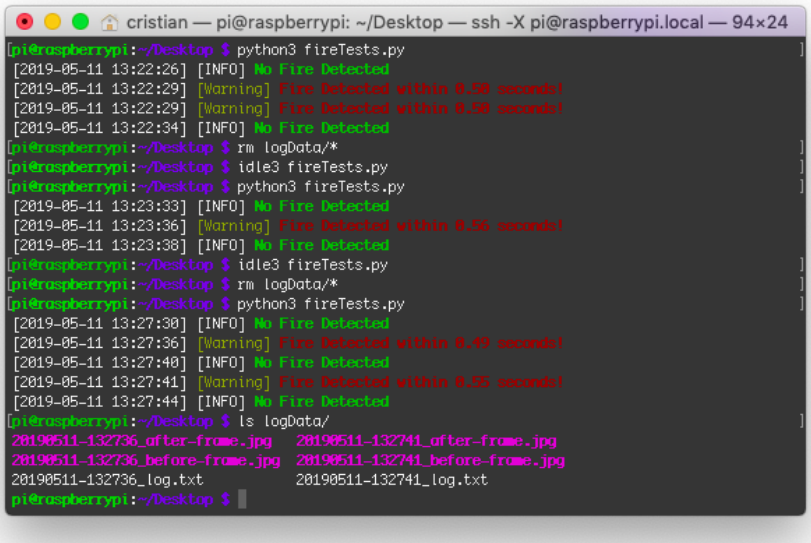


Figure 28: Example of log output and snapshots of detected fires.

Blackbox: This approach has an operation recorder, blackbox. In addition to live feedback in the console it records fire detection data in blackbox file on the sdcard, see Listing 3. Its name has the following format *balckbox_{timestamp}.txt*. In case of disconnection, out of range or other unpredicted situations the system may be lost. When the device is retrieved, recorded data can be examined from the blackbox file.

```
1  t4 = time.strftime("%Y%m%d-%H%M%S")
2  bbox = open("logData/blackbox_" + t4 + ".txt", "w+")
3
4  # log
5  def log(msg_lvl = "info", msg = ""):
6      st = time.strftime('%Y-%m-%d␣%H:%M:%S')
7
8      white_text = "\033[0;0;40m"
9      green_text = "\033[1;32;40m"
10     yellow_text = "\033[0;33;40m"
11     red_text = "\033[1;31;40m"
12
```

```
13      timestamp = "[{0}]".format(st)
14
15      msg_type = ("{0}[Warning]{1}".format(yellow_text,
            ↪ red_text),
16                  "{0}[INFO]{1}".format(white_text, green_text
                        ↪ ))[msg_lvl == "info"]
17
18      print("{0}{1}␣{2}␣{3}".format(white_text, timestamp,
            ↪ msg_type, msg))
19
20      _log = "{0}␣{1}".format(timestamp, msg)
21      bbox.write(_log)
22
23      return _log
```

Listing 3: Log function

6.4 Communication with the System

Raspberry pi usually requires additional equipment such as a screen for video output, a mouse and keyboard for data input. However, all the aforementioned peripherals would defeat the idea of being a wireless and portable system. Therefore, I decided that all of them can be simulated over the network so I setup wired connection for communication over ssh, on the same network. In other words, I have configured the raspberry pi to use the laptop's keyboard, mouse and screen over the network.

Once everything was working, it was time to carry on with the initial goal, make a wireless system. So far the system communicates with the operator over a network controlled by a third device. That third device, the modem, was not of any benefit, apart from connecting the system to the world. Thus, there was no need to keep the modem in the communication chain. A direct connection between operator and system is possible as long as one of the devices acts as the modem. The Raspberry Pi has the ability to act as an access point in a standalone network (NAT) using dnsmasq hostapd. Now, the Raspberry Pi can be accessed on the 192.168.4.1 address and port 22 wirelessly over Wi-Fi.

To make the connection process faster I decided to make use of the SSH Public Key Authorisation. Now is possible to SSH into the system using public/private key authentication. Faster, secure connection and the password prompt step is skipped. Login process presented in Figure 29.

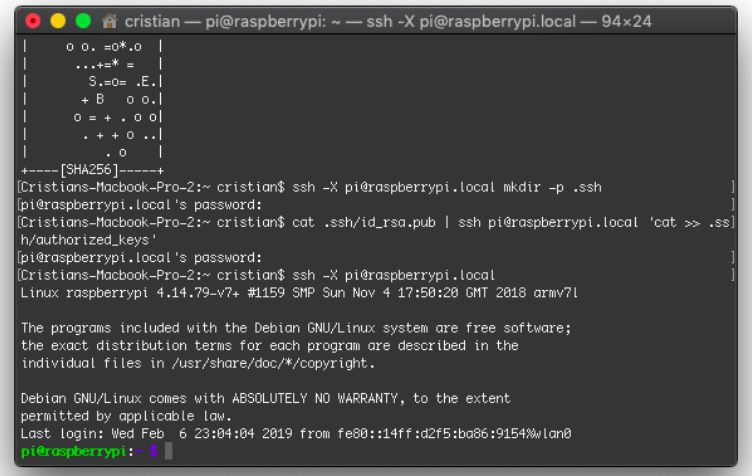


Figure 29: SSH Public Key Authorisation.

7 Experimental Results

During development, I will conduct a series of experiments. As it was expected, some succeeded and some failed. In the coming subsections, I documented the conducted experiments and the results of those experiments. I have conducted all the experiments with a blue light filter, see Figure 30, on and off.

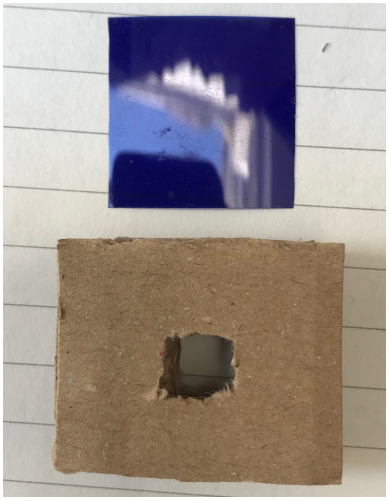


Figure 30: Blue light filter.

7.1 Effectiveness and Efficiency

For this section, I have tested the effectiveness and efficiency of the algorithm using a low cost, low quality, daylight camera with the infrared filter removed. Maximum accuracy of 90% has been achieved during night time experiments, and minimum of 10% during daytime experiments without the blue light filter.

Summary for Night Experiments					
Total Experiments	4	Positive Results Filter Off	3	Average Time Filter Off	0.45333333
Total Experiments	4	Positive Results Filter On	3	Average Time Filter On	0.41666667

Figure 31: Summary for Night Experiments.

Summary for Day Experiments					
Total Experiments	4	Positive Results Filter Off	1	Average Time Filter Off	0.61
Total Experiments	4	Positive Results Filter On	2	Average Time Filter On	0.6

Figure 32: Summary for Day Experiments.

Furthermore, the system has been tested for false positive results. I used a flashlight and the ceiling lightbulb to check whether the system would identify them as fire. Each experiment has been performed twice, first with the blue light filter on and second with the filter off. Presented in Figure 33 are the results of the false positives experiments.

Summary for False Positives Experiments			
Total Experiments	2	Positive Results Filter Off	2
Total Experiments	2	Positive Results Filter On	1

Figure 33: Summary of False Positives Experiments.

An example of false fire detection is presented in Figure 34. The system detects the light in the ceiling as a fire.

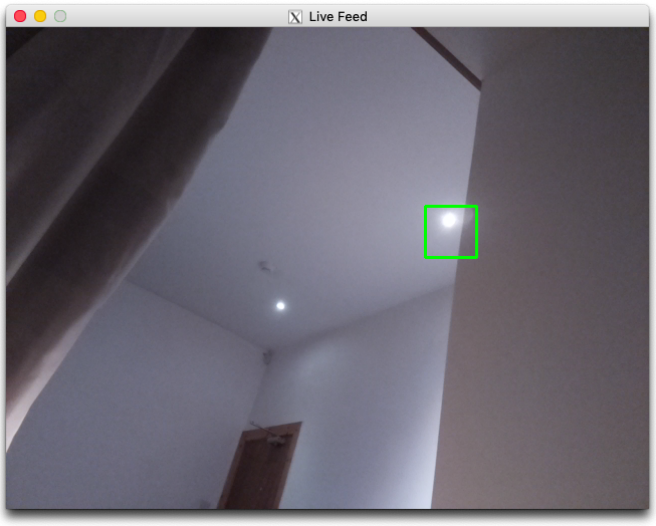


Figure 34: Light in the ceiling is detected as fire.

7.2 Range

In this section, I have presented performed experiments to test how far away can the system detect a fire accompanied with results. I have placed the system in one place and set controlled fires at 0.5m, 2m, 5m and 10m away from the system. Results of experiments during night-time are presented in Figure 35. The system handled very well the set task of detecting fires. At distances of 0.5m, 2m and 5m it detected the fires under one second, see **Time (Seconds)** column in Figure 35. However, it failed to detect a fire that was 10m away.

Range and filter Night					
Experiment	Distance	Filter Off	Filter On	Fire Detected	Time (Seconds)
E1	0.5m	Yes	No	Yes	0.54
E2	2m	Yes	No	Yes	0.34
E3	5m	Yes	No	Yes	0.48
E4	10m	Yes	No	No	-
E5	0.5m	No	Yes	Yes	0.38
E6	2m	No	Yes	Yes	0.37
E7	5m	No	Yes	Yes	0.5
E8	10m	No	Yes	No	-

Figure 35: Range Experiments during night-time.

Same experiments have been performed during daytime. Controlled fire was set at 0.5m, 2m, 5m and 10m away from the system. Close fire at 0.5m and 2m (with the blue light filter on) from the system was detected straight away under one second. All other experiments had negative results, see Figure 36.

Range and filter Day					
Experiment	Distance	Filter Off	Filter On	Fire Detected	Time (Seconds)
E1	0.5m	Yes	No	Yes	0.61
E2	2m	Yes	No	No	-
E3	5m	Yes	No	No	-
E4	10m	Yes	No	No	-
E5	0.5m	No	Yes	Yes	0.62
E6	2m	No	Yes	Yes	0.58
E7	5m	No	Yes	No	-
E8	10m	No	Yes	No	-

Figure 36: Range Experiments during daytime.

Presented below are charts of results of the performed experiments. As mentioned above, the system performed the best during night-time and very poor during daytime. However, is worth mentioning that there is a small improvement during daytime when using the blue light filter, see Figure 38.

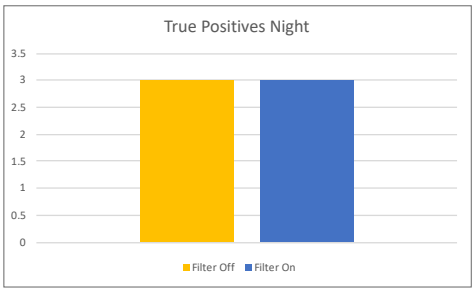


Figure 37: Chamarking theive results during night-time.

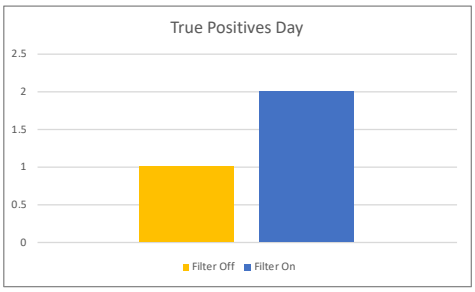


Figure 38: Chart of True Positive results during daytime.

Following experiments have been performed during night-time.

Experiment 1: Distance 0.5m and blue light filter off. Success, a fire was detected.

Figure 39: Image when a fire was detected, before marking the fire.

Figure 40: Image when a fire was detected, after marking the fire.

Experiment 2: Distance 0.5m and blue light filter on. Success, a fire was detected.

Figure 41: Image when a fire was detected, before marking the fire.

Figure 42: Image when a fire was detected, after marking the fire.

Experiment 3: Distance 2m and blue light filter off. Success, a fire was detected.

Figure 43: Image when a fire was detected, before marking the fire.

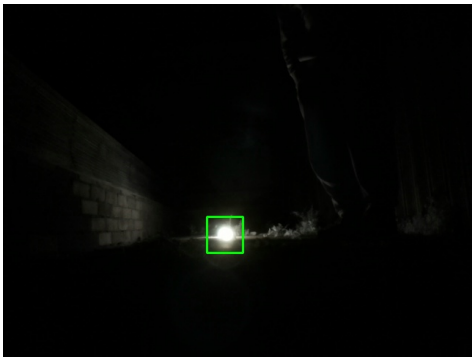


Figure 44: Image when a fire was detected, after marking the fire.

Experiment 4: Distance 2m and blue light filter on. Success, a fire was detected.

Figure 45: Image when a fire was detected, before marking the fire.

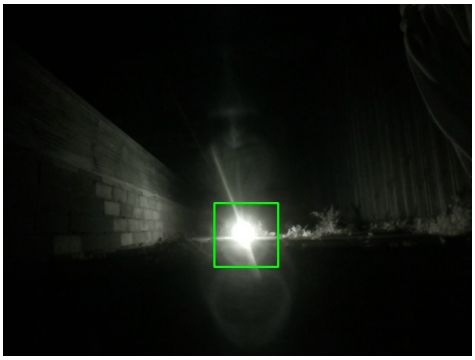


Figure 46: Image when a fire was detected, after marking the fire.

Experiment 5: Distance 5m and blue light filter off. Success, a fire was detected.

Figure 47: Image when a fire was detected, before marking the fire.

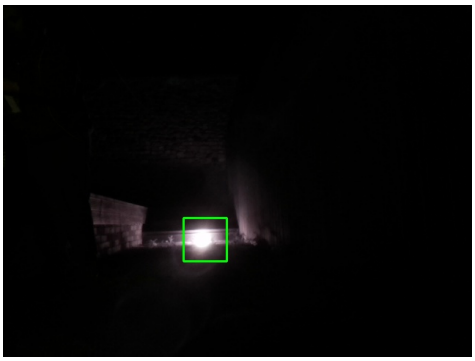


Figure 48: Image when a fire was detected, after marking the fire.

Experiment 6: Distance 5m and blue light filter on. Success, a fire was detected.

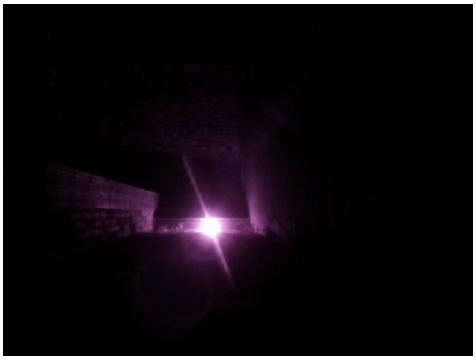


Figure 49: Image when a fire was detected, before marking the fire.

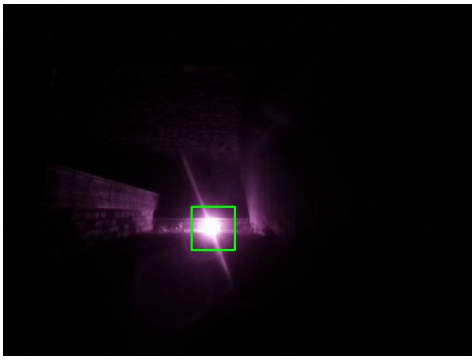


Figure 50: Image when a fire was detected, after marking the fire.

Experiment 7 and 8: Distance 10m and blue light filter off, Figure 51 and on, Figure 52. Fail, a fire was not detected.

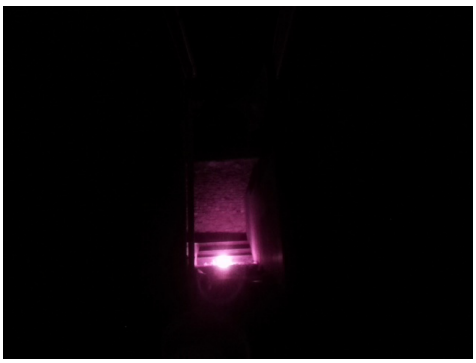


Figure 51: Snapshot of frame, blue light filter off.

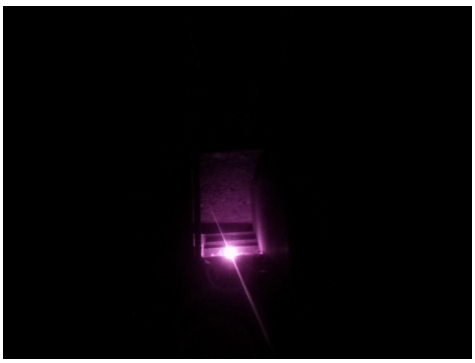


Figure 52: Snapshot of frame, blue light filter on.

Same experiments have performed during daytime. However, I will present experimental results of succesful experiments only.

Experiment 1: Distance 0.5m and blue light filter off. Success, a fire was detected.

Figure 53: Image when a fire was detected, before marking the fire.

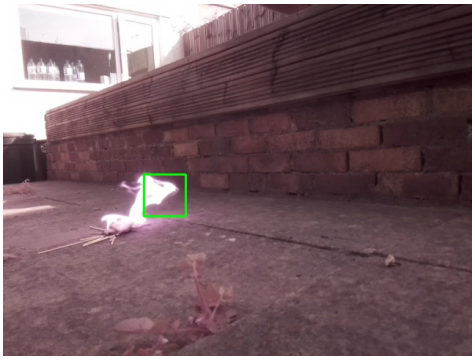


Figure 54: Image when a fire was detected, after marking the fire.

Experiment 2: Distance 0.5m and blue light filter on. Success, a fire was detected.

Figure 55: Image when a fire was detected, before marking the fire.

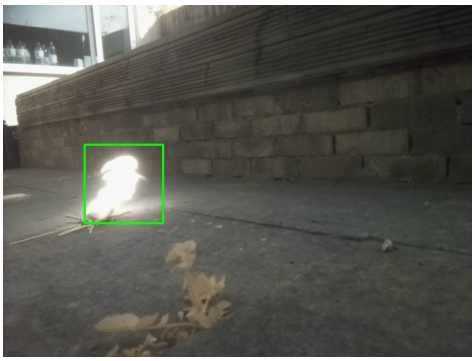


Figure 56: Image when a fire was detected, after marking the fire.

Experiment 3: Distance 2m and blue light filter on. Success, a fire was detected.

Figure 57: Image when a fire was detected, before marking the fire.

Figure 58: Image when a fire was detected, after marking the fire.

The system failed to detect the presented fire at 2m without the blue light filter, 5m and 10m during daytime and at 10m during night-time. The system seems to present higher accuracy during night-time.

7.3 Identify Close and Distant Objects

For this section, I attempted to determine whether the system would be able to differentiate between close and far objects. From a visual point of view, objects appear to be smaller the further away they are. So to answer the question, not entirely the system would be able to say if an object is far away or close based on its size, but no, there is no such ability of telling the distance to an object. A depth camera would be needed in order to be able to calculate distances.

8 Discussion

The system shows very promising results in fire detection, see Figure 26 and 59. It tends to have a higher accuracy of fire detection during night-time, which makes sense as there are not as many objects to reflect sunlight and emit heat.

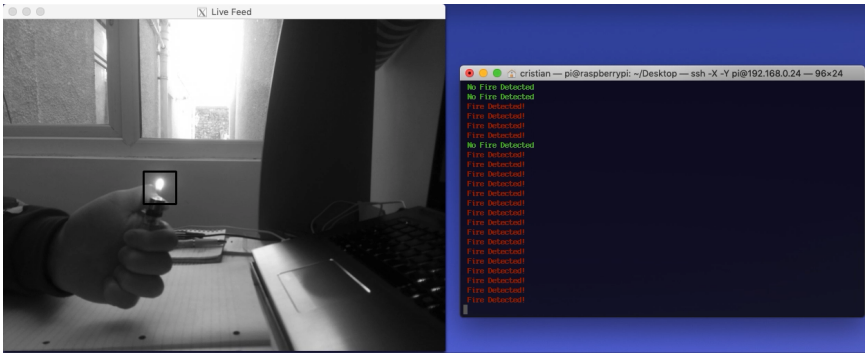


Figure 59: Fire detected during daytime: live feed and console feedback

However, the system is far from perfect and the rate of false positives is a bit high, see Figure 33. There are moments when it would not detect a fire at all, for example, Figure 60. Other live beings that have a visual perception of the world process what they see as a whole. For example, in Figure 60 a human would see a hand holding a lit lighter. And this is not done by looking at each pixel individually top to bottom but as a whole. On the other hand, computers examine pixels, top to bottom and a flame is just a cluster of pixels. If that cluster of pixels make up a shape that the computer knows that is a flame/fire, it would report it, otherwise is just a cluster of pixels. It is way more difficult to teach a computer what an object represents in a frame. In Figure 60 system does not see a fire, therefore there is no rectangle drawn around the lighter flame in the given frame.

Figure 60: Fire not detected during daytime.

The classifier can be further trained and taught of other shapes a fire can have. However, this would approach an accuracy of 99% maximum. There is always going to be the 1% of shapes that have not been seen before. A better approach would be to use a true Convolutional Neural Network. It would give the flexibility of learning a few shapes and being able to predict other unseen shapes. Even though Convolutional Neural Networks are much better that Haar-feature based Cascade Classifiers, the former is much slower as all the learning is done without from scratch. There is plenty of room for improvement and future work.

8.1 Strengths and Weakness of the System

The system can detect fire during any time of the day, with higher accuracy at night. Seems stable and does not crash. However, it thinks that flashlights and shiny objects are fires. Sometimes, it assumes that bright lights/reflections are fires.

The system is fully wireless, and other live feedback over a Wi-Fi network, hotspot. Two types of feedback is provided, visual, what the system is observing and text, whether it has detected fire or not. The visual feedback is sometimes slow, there is interference in the communication channel from outside objects, just like with regular Wi-Fi. Also, this feature comes with a price, it needs a power source. Because the system is expected to be in the air attached to a drone, adding a battery would increase the weight making the drone's job of carrying the system more difficult. More weight requires more power. Furthermore, batteries are an exhaustible, rechargeable, resource and do not last forever.

The system can quickly detect a fire. As seen in Experimental Results it happens in under a second. This means that it does very quick image processing. Even though the system has very low specifications, lower than most smartphones, it is able to perform very heavy tasks in a small amount of time.

8.2 What Have I Learnt

At first, I thought it might be impossible to achieve the set task. The little Raspberry Pi computer on a chip seemed slow and not very capable of much in comparison with bigger more powerful desktop computers. However, while working on this project I have learnt a lot about how computer vision works. I discovered the limits of the Raspberry Pi. Even though it is a small computer, it can achieve a lot.

As I was progressing through the project I learnt how to deal with the system's limitations and make use of all the available resources and get it to work better. It is fascinating how we have much more advanced systems doing less than the computer that can fit in your pocket.

I have learnt that no matter how powerless a tool, the Raspberry Pi, in this case, may seem, it is the person using in that makes it great. Surely, it is the aspect of processing power, where the system's capabilities are of high importance because the human can make it do what it is not able to do.

9 Future Work

There is a great variety of computers on a chip, just like the Raspberry Pi. These computers differ in specifications to accommodate different needs, more or less powerful, bigger or smaller. There are some fascinating computers on a chip which offer impressive performance.

A notable example is the NVIDIA Jetson Nano, manufacturing by the leading expert in computer graphics, Nvidia. The Jetson Nano offers a dedicated graphics chip which can bring image processing to another level. It also offers great specifications in terms of memory and processing power. It is ideal for AI and machine learning. If this project was to use Convolutional Neural Networks at some point in the future, this board would be ideal. However, in the industry of computers power equals money, so more power is more money. The Jetson Nano is not an exclusion, the tinny computer is highly priced and does not make exactly affordable for small projects.

As mentioned above, a better fire detection algorithm can be developed using Convolutional Neural Networks. These neural networks can be trained to detect fires based already seen fires and predictions from training data. This idea has already been explored in an experimental by professors at Durham University, UK. Their work illustrates high fire detection accuracy, maximal accuracy of 0.93 for whole image binary fire detection, with 0.89 accuracy within their superpixel localization framework. [23]

From the beginning, the system was intended to be deployed on a drone. Deploying it on a drone is just a small step towards the end goal of fire-watching forests. In its current state, the system can be attached to any third-party quadcopter drone. In the future, a drone with the system attached to it can be sent to explore a small part of a forest. Later on, the assembly can be sent to a forest that is burning for field testing.

However, wireless communication is not the best at the moment. The range is limited to the capabilities of the Wi-Fi wireless chip. And the signal is not great strength either, at times it lags because of external interferences. To conquer the range issue as a better wireless chip can be attached to the computer, for example, an HC-12 module HC-12 SI4463 Wireless Serial Port Module [21]. The HC-12 wireless module can communicate from distances of up to 1km.

10 Conclusions

In this paper, I have determined that building and deploying this technique of proactive fire detection is very much feasible and with least expenses. I have researched available technology to build the system, such as processing unit and cameras. It was concluded from research that a Raspberry Pi 3 Model B and a daylight camera with the infrared filter removed were a good fit.

The process started with a few failed attempts until the right approach, Final Approach: Raspberry Pi 3 Model B as the Main Board, has been found. Several experiments have been conducted and their results have documented in this paper, see Experimental Results. The system shows very promising results with relatively high fire detection accuracy at distances of up to 5m. Furthermore, the system is quite fast at processing live video, approximately 5 frames per second.

The goal of building a wireless fire detection system has been achieved. I believe this work will contribute to the common cause of dealing with wildfire issues.

Acknowledgements: Thanks to my supervisor, Dr Deepak Sahoo, for helping me figure out the specifics of the project. As well, thanks to everyone who shared their views and opinions on this project.

11 References

[1] P. Moore, J. Hardesty, S. Kelleher, S. Maginnis, and R. Myers, "Forests and wildfires: Fixing the future by avoiding the past," 2003, xII World Forestry Congress. [Online]. Available: http://www.fao.org/3/xii/0829-b3.htm

[2] NOAA's-National-Weather-Service, "Wildfire information and safety rules." [Online]. Available: https://www.weather.gov/otx/Wildfires

[3] A. Berg, "Detection and tracking in thermal infrared imagery," Swedish Licentiate's Thesis, Linköping University, Computer Vision Laboratory Department of Electrical Engineering Linköping University SE-581 83 Linköping Sweden, 2016. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:918038/FULLTEXT01.pdf

[4] Z. Wang, H. Song, H. Xiao, W. He, J. Gu, and K. Yuan, "A real-time small moving object detection system based on infrared image," in *2014 IEEE International Conference on Mechatronics and Automation*, Aug 2014, pp. 1149–1154. [Online]. Available: https://doi.org/10.1109/ICMA.2014.6885860

[5] C. Graham, "Ces 2016: Intel takes to skies with smarter drone," *telegraph.co.uk*, January 2016. [Online]. Available: https://www.telegraph.co.uk/technology/ces/12083983/CES-2016-Intel-takes-to-skies-with-smarter-drone.html

[6] M. Locatelli, E. Pugliese, M. Paturzo, V. Bianco, A. Finizio, A. Pelagotti, P. Poggi, L. Miccio, R. Meucci, and P. Ferraro, "Imaging live humans through smoke and flames using far-infrared digital holography," *Opt. Express*, vol. 21, no. 5, pp. 5379–5390, Mar 2013. [Online]. Available: http://www.opticsexpress.org/abstract.cfm?URI=oe-21-5-5379

[7] B. Mesnik, "Detection, recognition, and identification – thermal vs. optical ip camera," *kintronics.com*, June 2016. [Online]. Available: https://kintronics.com/detection-recognition-and-identification-using-thermal-imaging-vs-optical-ip-camera/?wpamp

[8] Opgal, "The use of johnson's criteria for thermal camera and systems performance," *Opgal*, August 2017. [Online]. Available: https://www.opgal.com/blog/thermal-cameras/johnsons-criteria-for-thermal-camera-and-systems-performance/

[9] Wikipedia, "Electromagnetic spectrum," *Wikipedia*, 2019, [Online; accessed 03-May-2019]. [Online]. Available: https://en.wikipedia.org/wiki/Electromagnetic_spectrum

[10] A. Robert, J. Joshua, C. Gregory, and J. Sion, "Airborne optical and thermal remote sensing for wildfire detection and monitoring," *Sensors*, vol. 16, no. 8, p. 1310, Aug 2016. [Online]. Available: http://dx.doi.org/10.3390/s16081310

[11] "Pi noir camera v2 specifications," Online. [Online]. Available: https://www.raspberrypi.org/products/pi-noir-camera-v2/

[12] OpenCV, *Open Source Computer Vision Object Detection*, 3rd ed., OpenCV. [Online]. Available: https://docs.opencv.org/3.4.3/d5/d54/group__objdetect.html

[13] KwonJH, "Fire detection." [Online]. Available: https://github.com/hojak99/fire-detection

[14] F. S. Leira, "Infrared object detection & tracking in uavs," Master's thesis, Norwegian University of Science and Technology, June 2013. [Online]. Available: https://pdfs.semanticscholar.org/83d3/db3fc6861f2b043aa368204a9f987c3c1652.pdf

[15] I. Bosch, S. Gómez, and L. Vergara, "A ground system for early forest fire detection based on infrared signal processing," *International Journal of Remote Sensing*, vol. 32, pp. 4857–4870, 09 2011. [Online]. Available: https://doi.org/10.1080/01431161.2010.490245

[16] M. N. H. Khan and C. Neustaedter, "An exploratory study of the use of drones for assisting firefighters during emergency situations," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, ser. CHI '19. New York, NY, USA: ACM, 2019, pp. 272:1–272:14. [Online]. Available: http://doi.acm.org/10.1145/3290605.3300502

[17] M. Prasad, G. Jaya Sree, K. Gnanendra, P. Kishore, and D. Anil Kumar, "Fire detection using computer vision models in surveillance videos," *arpn journal of engineering and applied sciences*, vol. 12, 10 2017. [Online]. Available: https://www.researchgate.net/publication/320555688_FIRE_DETECTION_USING_COMPUTER_VISION_MODELS_IN_SURVEILLANCE_VIDEOS

[18] M. ŚRUTEK and T. ANDRYSIAK, "Flame detection based on infrared images," *Problemy Eksploatacji*, vol. nr 2, pp. 197–208, 2013. [Online]. Available: http://yadda.icm.edu.pl/yadda/element/bwmeta1.element.baztech-1365ddc1-0637-47d1-a739-115f1ee2f14d/c/Srutek.pdf

[19] V. Di Biase and G. Laneve, "Geostationary sensor based forest fire detection and monitoring: An improved version of the sfide algorithm," *Remote Sensing*, vol. 10, p. 741, 05 2018. [Online]. Available: https://doi.org/10.3390/rs10050741

[20] "Bluetooth 4.0 ble module," Online. [Online]. Available: ftp://imall.iteadstudio.com/Modules/IM130614001_Serial_Port_BLE_Module_Master_Slave_HM-10/DS_IM130614001_Serial_Port_BLE_Module_Master_Slave_HM-10.pdf

[21] R. Rozee, *HC-12 Wireless Serial Port Communication Module*, 2nd ed., January 2016. [Online]. Available: http://arduinolab.pw/wp-content/uploads/2016/06/2016-01-14_122335_HC-12_v2.3B.pdf

[22] Glenn, "Fire detection with computer vision," 10 2017. [Online]. Available: https://www.codemade.io/fire-detection-with-computer-vision/

[23] A. Dunnings and T. Breckon, "Experimentally defined convolutional neural network architecture variants for non-temporal real-time fire detection," in *Proc. International Conference on Image Processing*. IEEE, September 2018, pp. 1558–1562. [Online]. Available: http://community.dur.ac.uk/toby.breckon/publications/papers/dunnings18fire.pdf