

UNIVERSITY OF
Waterloo



SE 101 Introduction to Methods of Software Engineering

December 2, 2019

Course Project Final Report

Group Name: Group

Team Member Full Name	userID	UserName
Frank Chen	20829078	f74chen
Matthew Vidov	20832667	mvidov
Willard Ma	20829201	w57ma
Patrick Wang	20837029	p238wang

1. Introduction

Forest fires are a natural disaster that devastate geographical regions and the environment. Once started, they grow at alarming rates, and cannot be effectively contained. However, their starting bush fires, if addressed timely, are more manageable in size.

We aimed to create a mobile rover that takes pictures periodically to identify potential bush fires. Then, the rover alerts the user to invoke a response. We identified 4 main components to our project: the physical design, the motor navigation, the rover-user communication and the fire identification.

Our rover is intended to be small and cost-effective, so large forests can deploy many affordably. As such, a significant challenge is to achieve the best functionality with limited equipment - dramatically increasing the difficulty and complexity of the project. Despite our success, further testing and extensions are recommended for the future.

2. Background Research

Fire detection tools are already available to the general public. However, traditional smoke alarms are not useful in the forest: They are limited in area coverage, and smoke expands into the open air. Rather than use costly heat sensors on our rover, we found that images can also be used to identify fires (using simple cameras). In his paper, *"Fast and Efficient Method for Fire Detection Using Image Processing,"* Celik outlines various such methods. Specifically, the method proposed by Chen and others uses red, green and blue pigment values as input - intuitive for our situation. These are subsequently converted to 'L*', 'a*' and 'b*' indicators through matrix multiplication (weighting the red, green and blue pigment values (RGB) by a Markov stochastic matrix), and then we individually analyze a sample to determine benchmark values for detecting a fire. The algorithm evaluates segmented regions of an image, and improves in success rate as the analyzed segments diminish in size.

Celik also presents the use of moving pixels as another tool to use in conjunction. However, a caveat is that the camera module must be stationary, or we need to take into account the movement of the module, and thus would only be considered as an extra extension.

Chen's team provides a simple and effective method, but the RGB model can be affected by luminescence and skew results. In *"Fire detection in a still image using colour information,"* Giwa and Benkrid present an alternative method that takes into account the intensity of the light. However, this model then uses a non-linear model.

To understand how to use RGB in java, we consulted an online tutorial page "How to get and set pixel value in Java". This tutorial helped us import the necessary libraries and get started on getting pixel values, and image sizes.

Research for the hardware focused around learning the use of our components: The Arduino breadboard, the camera module and the motors. We consulted online tutorials and used their diagrams to connect the wires in order. We used the ArduCAM Mini 2MP camera module and imported the arduino libraries from the website tutorial.

3. Implementation

First, we assembled the hardware components of our project. We soldered all of the wires to the proper boards, using online tutorials to ensure that we had the correct arrangement of wires for each component. We needed a stable base for the mobile rover, so we cut and used wooden pieces in the lab. This process was done twice, once for the prototype and once for the final version of the robot. Our rover runs on the Arduino Uno R3 system, which controls the camera module for image processing, the motors for mobility, as well as two bluetooth modules for communication. The robot is powered by 8 AA batteries split into two battery packs - one for the motors and one for the rest of the components.

To navigate, we control the motors through a computer where the rover moves based on keyboard controls. Each keystroke by the user activates the motors in a specific combination. This combination is relayed to the Arduino via the bluetooth module dedicated to receiving commands. Additionally, users can draw a path on a map to have the rover patrol an area by itself, along the pathway. This piece of software was made using Processing, which is a coding language based off of Java that is oriented towards graphical uses. The program turns the lines drawn by the user into "line objects", each of which possesses information about the line that it represents, such as its length and its angle. All these lines are then stored in an array that the program goes through in order to send movement commands to the robot.

Then, the attached front-facing camera module is programmed to periodically take images and sends them back to the computer via the bluetooth module. During this time, the computer will be running a Java program that is waiting for any type of changes to a specific folder. As soon as a new image is sent to the folder, the program then provides it to another class responsible for the image analysis.

For the fire identification software, we used Java once again since we were more familiar with it and it is more user-friendly than C. We integrated the research conducted and decided to analyze the weighted averages of the pigments for each pixel and then compare them to experimentally determined values we found by manually analyzing a sample of images with and without fire.

To analyze the images, we partition them into small segments and loop through each segment. Using the the matrix class in Java, we take averages for red, green and blue pigment values in each small segment and then transform them linearly into the "YCbCr" color space. The standard matrix for this linear transformation was obtained in our

background research. For the purpose of our project, we considered only the Cr component (red-difference chroma) component as one of our main indicators for fire, since we need to statistically analyze our indicators and manually identify fires in a sample of images, and would not be able to do this for too many indicators.

Our program “counts” the number of pixels that meet certain thresholds for Cr, as well as additional considerations on the original red, green and blue and saves this value. These considerations and thresholds were determined by running our program on 25 fire images and 25 non-fire images. At this point in the implementation, 21 fire images (84%) and 18 non-fire images (72%) could be “fitted” into thresholds in order for our evaluation to be correct (78% overall).

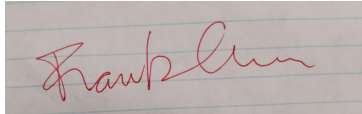
Too many false-positives occurred with one indicator, so we decided to add another. For the next indicator, our program goes through the image again and analyzes the difference between pixels and their surrounding edges using the Sobel operator. We noticed that colour differences within a fire could also be used as an indicator in conjunction with the Cr component. Counting the number of pixels that pass a threshold (determined experimentally), we combine this count with our first count and now consider this value from the combined indicators. Finally, we run the image through several case-related checks. This helps eliminate several common cases of false positives. For example, we found that the red clouds observed shortly after the sun sets below the horizon could potentially have a colour distribution very similar to that of a fire. However, we noticed that these clouds are often surrounded by blue or purple skies, colours which would not usually appear around a fire. As a result, we were able to implement an additional check for blue pigments around the “fire” and successfully allow the algorithm to correctly detect no presence of a fire in these instances. Now, 24 fire images (96%) and 20 non-fire images (80%) could be “fitted” into thresholds in order for our evaluation to be correct (88% overall).

The same Java program used to detect pictures that were sent from the rover is also responsible for displaying the results to the user. The results are displayed in the form of a Graphical User Interface (GUI). This was created from scratch using the AWT, ImageIO, and File libraries in Java. In addition to the colour, size, and location of each visual component, the animated Charmander had to also be implemented from start to finish. Using an online sprite editing app called Piskel, each of the 14 sprites used for animating the movement of Charmander had to be drawn individually. Then, the actual animation of Charmander was done using a loop and some time-based conditional statements. The animation serves two purposes: the first is to allow the user to tell if the program is frozen, since it is otherwise static while waiting for an input. The second is to add some life to a relatively sparse GUI that fits the theme of the project, making it more satisfying to use. After the algorithm decides whether or not an image contains a fire, the program then displays the picture that was sent by the rover in addition to a message corresponding to the result of the algorithm.

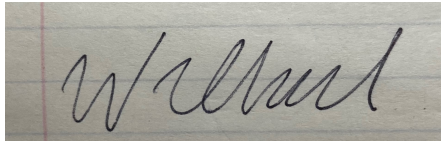
4. Group members' contribution

Each group member listed a breakdown of their tasks and an estimated time commitment for each task (in hours). The signature follows each respective group member:

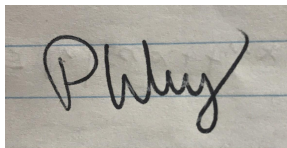
Frank: Contributed to planning and outlining the project proposal (0.5) and drafting proposal plan (1), contributed to prototype physical structure (2), made prototype presentation powerpoint (2) and organized presentation content (1), summarised and organized research (6), wrote final report (6).



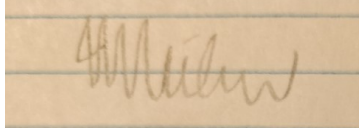
Willard: Built initial prototype of robot and experimented with various components, including soldering (6). Researched info for and developed program to wait for and receive images from robot via Bluetooth (2). Designed and developed Graphical User Interface (GUI) to link the image input software with the fire detection software as well as display the result for the user (4). Made the graphics for the GUI, including sprites of the Charmander animation (3). Tested and improved fire detection algorithm by suggesting more parameters to test as well as experimenting with the threshold values (4). Contributed to making the video presentation (4).



Patrick: Researched hardware components (1). Built prototype and final version of robot, soldered wire connections, and experimented with various components (6). Researched and developed software that allowed for bluetooth communication between robot and computer (2). Developed program for receiving user inputs, which includes both keystrokes and path drawings, relaying inputs to robot, displaying robot position and movement (6). Animated, filmed and edited the video presentation (6).



Matthew: Helped plan project and write proposal (0.5), compiled list of necessary components to purchase. Researched topic and developed the algorithm for fire detection (8), implemented the code and tested against sample images (9), and worked on assembling the project as a whole and larger scale implementation (2).



5. Final Product Evaluation

Our project satisfies the goals from our plan! We used cost-effective materials to create a mobile rover that detects fires. Due to our limited resources, our current model does not work for our ultimate goal however: our bluetooth device is very limited in range and much more testing is required for higher fire identification accuracy rates.

Additionally, we implemented the patrol feature extension - increasing the functionality of our rover and reducing the man-power required on the part of forest patrols to control the rovers.

Therefore, our final product is very successful and useful for our goal.

6. Design Trade-offs

At the beginning of the design process, one of the first questions that we were faced with was whether to control the hardware with an Arduino or a Raspberry Pi. Both are viable options for projects with a small budget, and either would work for us. The Raspberry Pi is actually more powerful in terms of functionality, but we decided that the extra simplicity provided by the Arduino was more beneficial for us in this particular project.

Another hardware design choice we had to make was what kind of sensor we wanted on the robot. Of course, a better funded project could implement all the relevant sensors, but our group could not afford to do so. Besides a camera, the other viable option was a temperature sensor. Since the only thing that could reach the temperature of a fire in a setting like the rainforest is a fire, this would allow for a method that is 100% accurate in its detection with no possible false positives. However, this would mean that the robot would have to get close enough to the fire to actually sense its heat, meaning that its range of detection is really small compared to that of a camera. Therefore, we decided that a camera would actually be more effective in detecting fires overall.

From our research, we found that luminescence could skew the results of images. Our group decided that this could be considered in the future but the luminescence on the forest floor should be generally consistent and would not have too large an impact. Therefore we concluded that the increase in resources required to consider this aspect outweighs the potential benefits.

For our software, we chose threshold values that sought to identify more fires, throwing false negatives rather than missing potential positives. The best statistical thresholds would have yielded 22/25 fire images (88%) and 25/25 non-fire images (100%) for an improved overall success rate of (94%). However, we decided instead to classify more images as potential fires - better to be safe than sorry!

7. Future Work

Our current model allows individual forest rangers to control multiple rovers and navigate through the forest terrain, in simple terrain. Future work would reduce the man-power needed to navigate and increase the flexibility of the rover to navigate harsher landscapes.

First, we would create a user-friendly application that lets forest rangers control and design pathways easily for multiple rovers. This app can have mobile extensions so the ranger can obtain information while patrolling.

Second, the physical design of the rover needs to be modified - the outer casing, the wheels, the motors, etc to allow for climbs and drops as well as defense against obstacles and wildlife. The camera module also needs modification: the camera should hold steady to transfer relevant images. In addition, using solar panels instead of a battery pack would be a significant improvement to the design since this means the robot can last for much longer while on patrol.

Furthermore, rovers could be equipped with small 'fire extinguishers' to combat the problem at the source immediately.

Finally, as with any statistical model, we can improve the accuracy by increasing the sample size. For our rover to be actually viable, significantly more testing would be needed against a much larger sample size.

8. References

- ArduCam, “ArduCAM Mini Cameras Tutorial,”
<https://www.arducam.com/knowledge-base/mini-tutorial/>
- A. Benkrid, O. Giwa, “Fire detection in a still image using colour information.” University of Portsmouth, 2017., <https://arxiv.org/ftp/arxiv/papers/1803/1803.03828.pdf>
- T. Celik, “Fast and Efficient Method for Fire Detection Using Image Processing .” *ETRI Journal*, vol. 32, no. 6, Dec. 2010, pp. 881–890.,
<https://pdfs.semanticscholar.org/e0d0/daa8c39dcecac9e3bb8afc4384d7534c93f1.pdf>.
- T. Chen, P. Wu, and Y. Chiou, “An Early Fire-Detection Method Based on Image Processing,” *Proc. IEEE Int. Image Process.*, 2004, pp. 1707-1710. [10.1109/ICIP.2004.1421401](https://doi.org/10.1109/ICIP.2004.1421401)
- Dylan Classroom, “How to get and set pixel value in Java,”
<https://www.dyclassroom.com/image-processing-project/how-to-get-and-set-pixel-value-in-java>
- How to Mechatronics, “Arduino and HC-05 Bluetooth Module Tutorial | Android Smartphone & Laptop Control,” Feb. 29, 2016, <https://www.youtube.com/watch?v=E-1w7dL3Cps>
- Sparkfun, “TB6612FNG Hookup Guide,”
<https://learn.sparkfun.com/tutorials/tb6612fng-hookup-guide/all>