COMP6441

SOMETHING AWESOME

CHARLEY WONG

[Introduction 4](#_Toc515103837)

[Problems Encountered 5](#_Toc515103838)

[Solving Problems 6](#_Toc515103839)

[Progress Log 7](#_Toc515103840)

[Conclusion 8](#_Toc515103841)

## Introduction

This report details the process of developing an application which uses the command line to capture images using an inbuilt webcam (currently only tested on MAC OS X). In this day and age, having your photo taken without your consent is a breach of privacy even more so if it’s captured on your own device. Often, films associate such a breach with blackmail and threats. Media aims to warn us about the dangers of hackers and how they can take control of our lives, manipulating us into doing what they want.

The program written will capture an image from command line and save the image wherever the program is saved. At this point in time, when the image is captured a light will still be activated to show that it is active and the image will be saved under “temp[0-9]+” in the same directory the program is saved in. The report will discuss the challenges faced and how they were overcome.

## Problems Encountered

A lot of problems were encountered when undertaking this project and they will be discussed as well as analysed in this section. The problems encountered will be discussed in a sequential format and how they were overcome will be discussed in the next section.

1. Lack of Experience with Objective-C

When doing research for this topic and how to approach programming it became clear the knowing how to program in Objective-C was required. I had no knowledge of specific Objective-C and didn’t realise just how different it was from C or C++. Though the thought process was generally the same, there was a different way of calling functions. For example, to call a function from another class:

ret = [sampleClass max:a andNum2:b]

This roughly reads, ‘ret’ is calculated by parsing in ‘a’ and ‘b’ into a function ‘max’ that’s unique to a ‘sampleClass’ object. This of course is different to say C, where we would be doing:

Ret = sampleClass.max(a,b);

1. Being unfamiliar with Objective-C’s built-in libraries and how to use them

The crux of this program fell to being able to use the inbuilt libraries to give me access. It was down to doing enough research and understanding what each thing did so that I could actually make use of the device’s hardware. This was something I’d never really attempted so it was something different for me.

1. Major bug – Images are never saved

One of the biggest bugs of my program was that the light for the webcam flashed – indicating that it was in use, but no image was ever created. A lot of my time was spent on this issue and trying to fix in. In the end, no concrete solution was ever found but a ‘hack’ was utilised to generate the images. Research was conducted and after testing the program I wrote, I realised that the function used to actually capture the image:  
“captureStillImageAsynchronouslyFromConnection:”  
is a built-in async function which runs parallel to the main. This function never actually finishes writing data to file and the program ends prematurely so the image data captured never saved.

1. Images produced are dark

After figuring out how to write image data to fie, the next problem encountered was that the image produced was extremely dark. The image produced looked similar to when a lens isn’t open long enough and thus cannot capture enough light. Another explanation could be that the webcam itself isn’t able to focus itself properly or adjust white balance as needed.

Solving Problems

Below, I detail how I overcame the problems I encountered and discuss what specifically fixed the problem

1. This problem was easily solvable just by doing enough tutorials for Objective-C and making sure that I wasn’t just reading code and that I was actually practicing. Due to a lack of time, I had to follow quick tutorials that helped me understand the basic structure of it. By doing tutorials, I understood the process of function calling in Objective-C, which deviated the most from C or C++.
2. I found a tutorial online – documentation by Apple – which clearly outlined what I needed to do, which libraries I needed and how I could use them to do what I wanted. Whilst the tutorial told me what I needed to do, its descriptions were not very succinct and simply described the bare minimum. I found myself googling the required functions and what they actually did. This is reflected in the comments I have in my program.
3. After discovering that the program ended before the image data could ever be written to file, I started focusing on prolonging the program. Initially, I used ‘sleep(*time)*’ to try to solve this and it worked, an image was being written to file, but it was extremely dark. I wasn’t sure what the cause was so I decided to put everything into a loop and call the async function a large number of times. This resulted in a number of images being saved with each of them becoming increasingly brighter. This meant that I had finally achieved my initial goal. However, the first image was always extremely dark and no matter what I did, this was something that would never change. This leads me to my next issue:
4. The consistent darkness in the first image suggested that each image captured was building upon the previous one. Since the async function I used a buffer to store image data, I assumed that the buffer was never cleared after the function completed itself and that each image write was using the buffer in the previous call. There was nothing I could do about this and using sleep instead of repeatedly calling the function just meant that the image produced would always be dark. In the end, I used a combination of both, and instead of calling the same function multiple times, I decided to call it a total of twice and in each iteration of the loop, I’d call sleep(1) so that the function would finish itself. This also meant that I’d be creating two images, with the second image’s quality building on the last.

## Progress Log

**Week 5**

* Started researching into what was needed to complete what I needed to complete including webcam architecture and language libraries
* Started completing Objective-C tutorials

**Midsem break, Week 6**

* Nothing much done apart from light tutorials on objective-c
* Plan written up for what I need to do in terms of programming (programming logic-wise)

**Week 7**

* Setup basic input and output destinations for the program
* Need to write image data to file

Week 8

* Completed program but major bug encountered
  + No image is saved to directory
* Discovered why the bug was happening and what steps I could take to fix it
* Implemented steps outlined and finalised program despite it not being a solid solution

Week 9

* Still no *concrete* solution found
* Researching a concrete solution

Week 10

* More research into a more concrete solution
* Implement, use and test out different data structures to support program

Week 11

* Stuck with loop from week 8
* Took time off from Something Awesome to complete other subject’s assignments

Week 12

* Completed program. There’s a workaround, it’s not solid but much better than previous versions.
* Writing report