Fall Term Retrospective

Group 09: Handheld AR Device

Christopher Miyasako

Thomas Lane

Jacob Sowanick

Section 1: Project Goals

Our device gives users a new and intuitive way to use Augmented Reality. Since everyone knows how to use flashlights, operation of this device would be as simple as turning it on and pointing. The digital shapes are also shown as a projection instead of through a head mounted screen, making our images shareable. One person can operate the AR device and anyone standing around them will be able to see the same thing that they are seeing. The real life applications for this device extends farther than just fun demonstrations. It could be used effectively during construction as a way for the employees to scan a certain section of wall to see where any studs, pipes, or wires are. It could also be used in an education setting, such as a museum, where guests can illuminate certain displays to see more information in an interesting and interactive way.

For the project, the scope will be centered around the implementation of these features as the minimal deliverable scope and refining these features. If time allows, the next features to be implemented will be to add portability by placing these features into the form of a flashlight. Based on the time and resources available to us, a working prototype should be completed by term 3 of the year that will be refined and ready for continued improvement by future groups working on this project.

Section 2: Project Status

For Sprint 1, the goals of the project were for the Raspberry Pi to be able to take input from the camera module and then recognizing a QR code or barcode from the image to show the viability of Computer Vision for this project.

Take in Input from Camera Module	100%
Recognize QR Code from Image	50%

The next goals of this project would be to focus on implementing computer vision functionality for the project to allow the device to recognize its surroundings based on the input from the camera. Over winter break, the goal will be to stay in contact with the group and the project partner while preparing for the next sprint by ensuring that the OpenCV library that will be used for the computer vision portion of the project is correctly installed and correctly working on the Raspberry Pi. Due to the large scope of the project, it is recommended to decrease the project scope to focus primarily on implementing computer vision functionality by recognizing QR codes and being able to pull up relevant data depending on the QR code.

Section 3: Problems

The project's current problems exist with both getting certain libraries, like OpenCV, to work on a raspberry pi and in connecting the two parts that we have already completed.

While a QR code can be recognized with code run on a windows desktop, the transition from that desktop to the pi caused issues that should be quickly solvable. Additionally, that code will need to have the image taken by the camera module passed into it in order to detect the QR code in the image.

Once we have a program that can capture an image, detect a QR code in the image, and then read that QR code, we will be done with our first portion of the project. This will allow us to move on to the next portion of the project, which will be to detect a QR code in a video feed and to keep track of the QR code's position.

Section 4: Project highlights

The following bits of code demonstrate how the functionality to take in input from the camera module of the Raspberry Pi has been implemented and the Raspberry Pi is capable of recognizing a barcode in the image and outputting something back.

```
# Reads in an image after showing a short preview
# and saves it as a file with current date and time
def capture(self):
    self.camera.start_preview(fullscreen=False, window=(100,20,640,480))
    sleep(3)

    # This is all to just get the file name
    # which is the current date and time
    d = datetime.now()
    timename = "/home/pi/Pictures/"
    timename = timename + str(d.year) + str(d.month) + str(d.day) + "_"
    timename = timename + str(d.hour) + str(d.minute) + str(d.second)
    timename = timename + str(d.microsecond) + ".jpg"

    # If there is an error, it will handle it, that way
    # we can still stop the preview
    try:
        self.camera.capture(timename)
        print("Capture Saved to \"" + timename + "\"")
    except:
        print("There was an error capturing \"" + timename + "\"!")

    self.camera.stop_preview()
    return
```

```
Cam = CameraModule()
assert (Cam != None),"Camera Module not initialized"
print("Camera Module Initialized")


print("Say Cheese!")
Cam.capture()
```

Figure 4.1. This code takes an image from the command line and reads a QR code

```
# import the necessary packages
from pyzbar import pyzbar
import argparse
import cv2

# construct the argument parser and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-i", "--image", required=True,
    help="path to input image")
args = vars(ap.parse_args())

# load the input image
image = cv2.imread(args["image"])

# find the barcodes in the image and decode each of the barcodes
barcodes = pyzbar.decode(image)

# loop over the detected barcodes
for barcode in barcodes:
    # extract the bounding box location of the barcode and draw the
    # bounding box surrounding the barcode on the image
    (x, y, w, h) = barcode.rect
    cv2.rectangle(image, (x, y), (x + w, y + h), (0, 0, 255), 2)
    # the barcode data is a bytes object so if we want to draw it on
    # our output image we need to convert it to a string first
    barcodeData = barcode.data.decode("utf-8")
    barcodeType = barcode.type
    # draw the barcode data and barcode type on the image
    text = "{} ({})".format(barcodeData, barcodeType)
    cv2.putText(image, text, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX,
        0.5, (0, 0, 255), 2)
    # print the barcode type and data to the terminal
    print("[INFO] Found {} barcode: {}".format(barcodeType, barcodeData))
```

Figure 4.2. This is code that uses computer vision to recognize a barcode in an image

Section 5: Design Review Feedback

We got one piece of feedback from Julian Fortune that was very well written. The main point of concern from the feedback centered around the scope of the project and that it was unachievable in the limited amount of time that we have. The feedback suggested to narrow down the scope in order to focus on what we can do. There was also concern about the team's unfamiliarity with designing and building hardware as it

was originally supposed to be a large portion of the project. From the feedback that was received, there were groups with similarities in project goals and tools that could be helpful in working with, such as group 06, that utilized computer vision in part of their project.