Monitoring Face Mask and Social Distancing Regulations using Drones for COVID-19

Zachary Walker-Liang[‡]

June 2, 2022

Abstract

Our goal was to create a system on a drone that can overlook a group of people and report if they are following COVID-19 guidelines. We used computer object detection to distinguish if people are wearing a mask, not wearing a mask, or wearing a mask incorrectly (below one's nose or mouth). We then took this software and implemented it onto a drone, so that the drone can patrol autonomously around an area and determine who is following guidelines.

1 Introduction

The COVID-19 pandemic took the world by surprise, causing places everywhere to go into lockdown. During lockdown there are rules and regulations, such as wearing a face mask, avoiding large groups, and staying 6 feet away from any person. Our goal was to create systems that can monitor how well COVID guidelines are being followed in an area. We used a drone as our means to capture data of the population regarding COVID-19 guidelines. We chose to use drones because they are nimble, versatile, and adaptable to many different areas / climates.

[*]Samantha Wong
[†]RSL Lab

2 Mask Detection using YOLOv3

We want our drone to be able to determine whether someone is wearing a mask, regardless of mask color or type. We also want our drone to be able to determine whether someone is wearing their mask incorrectly. We used YOLOv3, a real-time object detection system, in order to process images taken from the drone. YOLOv3 accepts a 2D image (pixel and colors) and applies a single neural network to the whole image. This allows for a global context in the image, as it looks at the whole image at test time. YOLOv3 neural network divides the image into regions, and predicts the bounding boxes and probabilities for each region. This predicts an "objectness score" for each bounding box using logistic regression. The higher the objectness score is, the more likely the object in question is within that box.

Figure 1: Mask Detection, IPhone photo

In Figure 1, YOLOv3 detected my mask with an 100% objectness score. The YOLOv3 neural network works by us "training" it. We found many images with people online wearing masks, then annotated and labeled these images by drawing boxes around the mask. If someone was wearing their mask incorrectly, we would draw a mask around the area and label it as an incorrectly worn mask. To make these annotations, we used Microsoft;s Virtual Object Tagging Tool, then exported that data to a CSV file. Then, we feed this data into the neural network. YOLOv3 automatically splits our data into two different categories, **Training** and **Testing**. The **Training** data will be the images that the neural network use to develop their sense of what a mask looks like. The **Testing** data will be images that the neural network uses to test it's accuracy while training, these images are unseen to

the network while it is learning. When our neural network is done training, it produces files called weights. These weights are used by the computer to determine what objects are in an image.

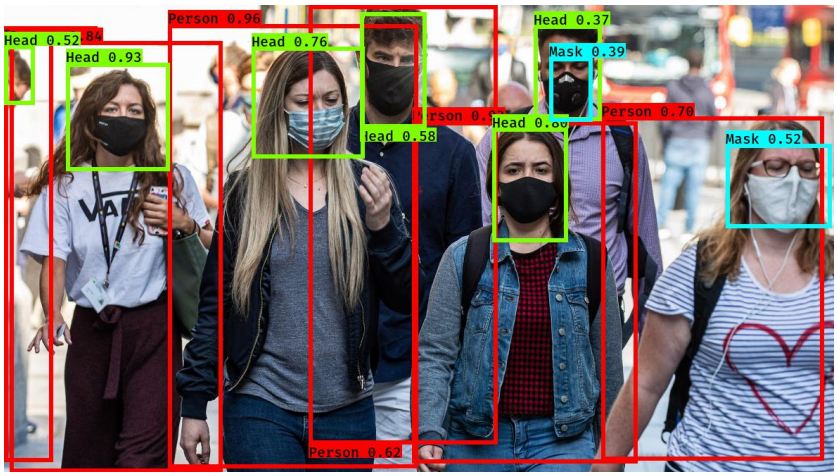

Figure 2: Example of our weights executed on an image

Our trained weights are used to detect objects from images, as seen in Figure 2 and Figure 1. We trained our weights to detect four different classes: Mask, NotMask, Head, and Person. The class Mask indicates a mask on a face of a person. The class NotMask indicates when someone is wearing their mask incorrectly (beneath the nose or mouth). The classes Person and Head will be important later on when centering people in frame and determining if they are social distancing.

3 Drone Following Objects in 3D space

Our drone software allows the drone to follow people around in 3D space. We localize the person in relation to the drone by getting the center of the persons' bounding box. We then adjust the drone accordingly, by either flying up, down, or rotating. If the drone decides that a persons bounding box is too small, that means the drone is too far away. The drone will then move forward to adjust accordingly. The drone's adjustment is constant for forward/backward and up/down movement. The drone has one smaller rotation option and one bigger rotation option when deciding if a person is

too far left/right, depending on how far off the person is. We control our drone (Parrot Mambo) using a library called pyparrot.

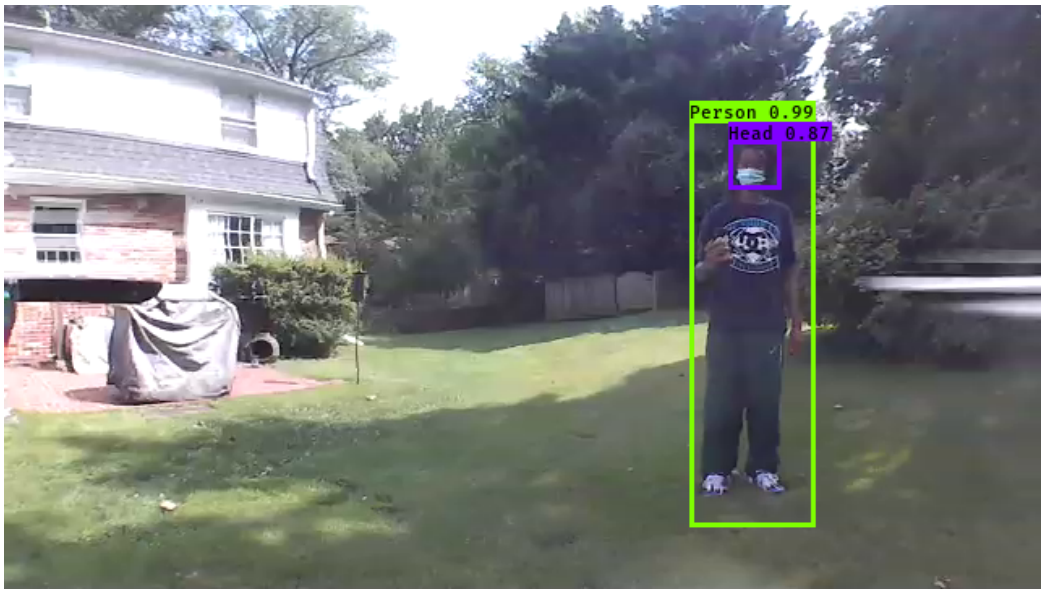

Figure 3: Drone following me in 3D space

In the case of Figure 3, drone would rotate clockwise and move closer to person. The drone would not need to adjust vertically, as the person is already in the centered in the frame vertically.

4 Drone Mask Detection Technique

We want our drone to survey a population of people and report back who is wearing a mask or not. The drone uses Figure 4 to decide its actions.

The goal this algorithm is to focus on one person at a time, find out if they are wearing a mask, and then move on to the next person. During this search, the drone will also be checking if people are social distancing.

Our algorithm first checks if *focus* is null. *focus* variable holds the bounding box of a suspected unmasked person. If *focus* is null, we take the current image from the drone, and run our YOLOv3 weights on it. This will give us a CSV file, containing the bounding boxes of our objects of interest. Then, we use these results to check if people are social distancing (see next Section). We then iterate through all people in the scene, and try to find if

```
while(true)
    if focus is null
        run weights on new image
        run SocialDistancingCheck
        for each person in scene
            if no mask box in person box
                focus = person
        if focus is null
            drone rotate small
    else
        run weights on new image
        person set to closest distance to focus
        if mask box in person box
            focus = null
        else if badmask box in person box
            focus = null
            // add code to save last image, or increment variable
        else
            get closer to person
            focus = person
```

Figure 4: Main Control Algorithm

there is a mask in anybody's bounding box. If there isn't a mask on someones bounding box, they become the *focus*. If there is nobody in the scene, or if everybody is detected with a mask in the scene, *focus* will still be null, and drone will rotate a small amount to seek other individuals.

If *focus* is not null, we run weights on current image again. We update the *focus* to the Person in the scene closest to it. We then check if they are wearing a mask. If they are detected with a mask on, the new *focus* will be set to null. If they are not detected with mask on, the new *focus* is kept the same, and drone adjusts to center camera to person.

We need our drone to center person in the scene because masks can be hard to see. It is much easier to detect a person from far away than a mask from far away; getting closer to the person will ensure good mask detection.

5 Drone Social Distancing Technique

After focusing on a person to find a mask, our drone will scan the area for social distancing violations. The software behind the social distancing check was done by my research partner Samantha Wong.

Figure 5: Before Social Distancing Detector

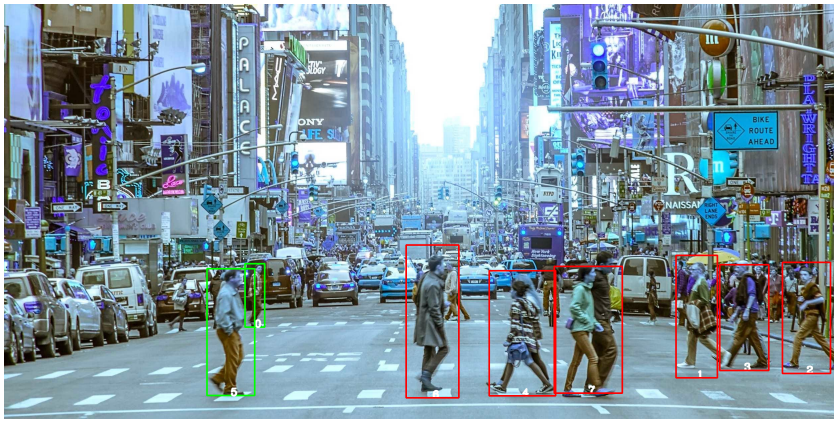

Figure 6: After Social Distancing Detector

The red boxes (as shown in in Figure 6) notifies us that this person is not social distancing. A green box indicates that this person has a 6ft radius around them with no people detected. When the drone detects that someone

is not social distancing, it will store that image. Social distancing violations will be stored on your computer inside of a specified folder.

6 Project Improvements / Follow Ups

6.1 SLAM / Area Patrol

Currently, the drone has no way of avoiding objects during flight. If SLAM(Simultaneous Localization and Mapping) is adapted to our software, the drone could navigate throughout scenes without bumping into objects. The drone we used for our project, Parrot Mambo, has a normal 720P RGB camera. ORBSLAM2/3 was experimented with near the end of this project.

Currently, the drone rotates until it finds a person, closes into them and decides if they are wearing a mask, and then looks for the next person to evaluate. With SLAM installed, there could be code written for the drone to patrol an area. The user could specify and area, and the drone could patrol across that area in a pattern.

6.2 Better Distance Detection

Currently, our software commands the drone to get closer to a person if a mask is not seen. The drone has a minimum distance of how close it can get to people, or else the software will tell it to back up. Currently, this minimum distance is calculated with the area of the person box. This could be improved, as people who are shorter and skinnier will have the drone get closer than someone who is not short and skinny. This method also is not precise when people are in awkward positions, (such as hands outstretched - increases area of box).

6.3 Person Facing Backwards

Our software cannot yet determine whether someone is facing fowards or backwards. This causes the drone to freeze up when backs are turned to it. This could fixed by training another class for the YOLOv3 weights that determines the front of a person.

7 Conclusion

We created software for a drone that can enforce COVID-19 guidelines through object detection. Our software commands a drone to look for a target, center to the target with the appropriate distance away, and determine whether they are wearing a mask or not. Our software will than command the drone to look for the next target. The drone will periodically check the scene to make sure people are social distancing.

Even after the COVID-19 pandemic, this technology could be useful in stopping other respiratory spread diseases in the future. Even with the science out, many people around the world failed to take the correct precautions to keep people safe during a pandemic. We believe this technology could be great to collect data on how many people are following pandemic guidelines in an area.

Tools Used

1. YOLOv3, https://pjreddie.com/darknet/yolo/
2. pyparrot Library, https://pjreddie.com/darknet/yolo/