

## Final Project 2

Instructor: Mahya Safavi

In this project we aim to utilize the camera on a Raspberry Pi to detect human heads and substitute them with cat heads. 3-4 people per group.

## Camera Initialization

The initial step involves connecting the camera module to the Raspberry Pi. To learn how to connect the Pi camera, you can refer to a tutorial on YouTube titled "[HOW TO USE the Raspberry Pi camera module - YouTube](#)." Once the camera is installed, execute the following command in the Pi's terminal to verify if the camera is operational:

```
>> libcamera-still -o test.jpg
```

If the camera is operational, it will activate, display a preview, take a photo, and save it in your home directory as "test.jpg". To connect the camera to Python, you can install the Picamera2 library using this command:

```
>> sudo apt-get install python3-picamera2
```

In Python, you can execute certain command lines to verify if the camera is accessible:

```
from picamera2 import Picamera2, Preview
import time
picam2 = Picamera2()
camera_config = picam2.create_preview_configuration()
picam2.configure(camera_config)
picam2.start_preview(Preview.QT)
picam2.start()
picam2.title_fields = ["ExposureTime", "AnalogueGain"]
time.sleep(2)
picam2.capture_file("preview.jpg")
```

This code activates the camera for a preview, captures an image after 2 seconds, and saves it to a directory. To store the captured images in an array, we can use the following command:

```
array = picam2.capture_array("main")
```

In this instance, within the Python terminal, you can view the array, which is a NumPy array. By executing this command, you can determine the shape of the array:

```
>> array.shape
```

```

9 array = picam2.capture_array("main")

```

Running: captureArray.py

```

    [ 81,  75,  82, 255],
    [ 71,  65,  79, 255],
    ...,
    [ 43,  42,   7, 255],
    [ 40,  37,   0, 255],
    [ 36,  30,   0, 255]]], dtype=uint8)
>>> array.shape
(480, 640, 4)

```

The array should have a shape of (Height, Width, 4) for RGBA encoding. For RGB encoding, the shape is (Height, Width, 3). Additional documentation can be found at [The Picamera2 Library](#)

## Detecting Human Heads

We recommend the OpenCV library for image processing in Python. To install the library, enter the following command in the Pi's terminal:

```
>> sudo apt install -y python3-opencv
```

While other libraries such as pyplot or tkinter can be used for processing, OpenCV offers built-in detection algorithms that may be beneficial for this project. To initiate camera preview mode using OpenCV, you can utilize the following code snippets:

```

import cv2
from picamera2 import Picamera2
picam2 = Picamera2()
#//picam2's setup for OpenCV
while True:
    frame = picam2.capture_array()
    cv2.imshow("Preview", frame)

```

OpenCV is limited to reading images in BGR format. To modify the camera's output format, consult the Picamera2 documentation. For head detection using OpenCV, download the file from the [opencv/data/haarcascades at 4.x · opencv/opencv · GitHub](#) and transfer it to your Raspberry Pi. This folder includes a number of built-in classifiers in the \*.xml format. To access them in Python, you can utilize the following command:

```

detectClassifier=cv2.CascadeClassifier('./RaspberryPiDirectory/desiredClassifier.xml')

```

During the preview period, you can locate the desired object using this code:

```

while True:
    frame= picam2.capture_array()
    frameGray=cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)

```

```
detectedRectangular= detectClassifier.detectMultiScale(frameGray,1.3,5)
```

In this code, a gray frame is utilized instead of a BGR frame for object detection, which increases the frames per second. The final variable "detectedRectangular" consists of an array with four integer values: (x, y, width, height). The first two represent the x and y coordinates of the object's top-left corner, while the last two denote the object's width and height.

## Replacing Heads

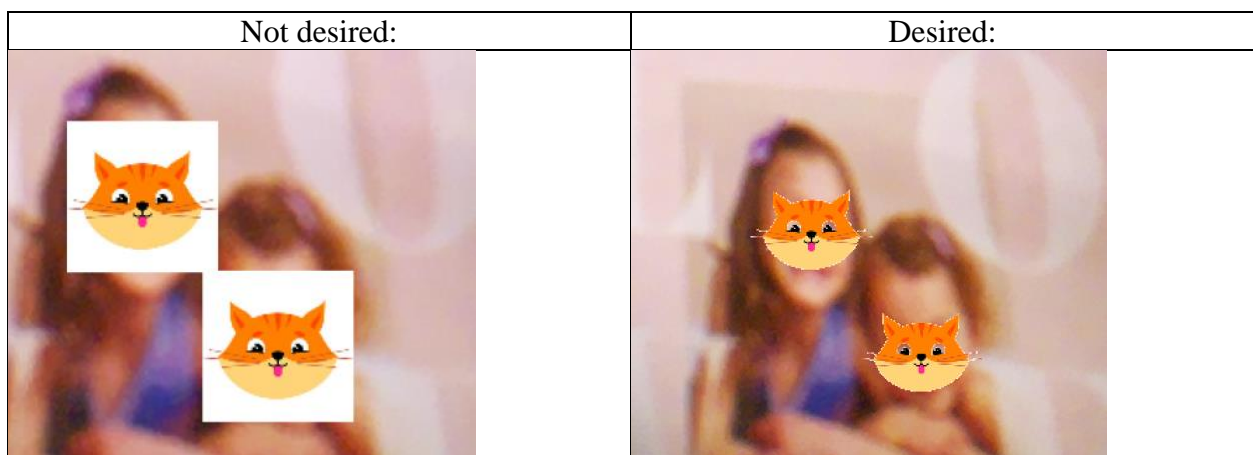
Once the head is identified as a rectangle, it can be substituted with a different image. The recommended image is depicted below.



This code allows you to read the image and access each cat head:

```
img = cv.imread(address[ID],cv.IMREAD_UNCHANGED)
def make_cat(img, number):
    h= 997;
    w = 1000;
    y = [0, int(h/4-60),int(h/2-60),int(3*h/4-80),int(h-80)]
    x = [0, int(w/4),int(w/2),int(3*w/4),int(w)]
    i=number//4;
    j=number%4;
    smaller_img = img[y[j+0]:y[j+1],x[i+0]:x[i+1],:]
    return smaller_img
```

When substituting the head with a cat's head, ensure to eliminate the white space surrounding it.



## Homework problem:

Use the mentioned libraries to detect human heads in Raspberry Pi and replace it with Cat's head