OBJECT DETECTION WITH VOICE FEEDBACK

A project report submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

In

Computer Science and Engineering

By

T. LEELAVATHI(N180197)

Under the Esteem Guidance of

Dr.D.V.NAGARJUNA DEVI

Assistant Professor, CSE Dept.

Rajiv Gandhi University of Knowledge Technologies – Nuzvid
Department of Computer Science and Engineering
(A.P. Government Act 18 of 2008)
RGUKT-NUZVID, Eluru Dist - 521202

Rajiv Gandhi University of Knowledge Technologies – Nuzvid
Department of Computer Science and Engineering
(A.P. Government Act 18 of 2008)
RGUKT-NUZVID, Eluru Dist - 521202

CERTIFICATE OF COMPLETION

This is to certify that the project entitled, "**OBJECT DETECTION WITH VOICE FEEDBACK**" is work of "**T.LEELAVATHI (N180197)**" is a recent record of bonafide work carried out by under our guidance and supervision for the partial fulfillment for the degree of Bachelor of Technology in Computer Science and Engineering during the academic session, February2023-July2023 at RGUKT – Nuzvid. To the best of my knowledge, the results embodied in this dissertation work have not been submitted to any university or institute for the award of any degree or diploma.

Dr.D.V.Nagarjuna Devi,
Project Supervisor,
Department of CSE,
RGUKT – Nuzvid.

Examiner,
Department of CSE,
RGUKT – Nuzvid.

Rajiv Gandhi University of Knowledge Technologies – Nuzvid
Department of Computer Science and Engineering
(A.P. Government Act 18 of 2008)
RGUKT-NUZVID, Eluru Dist - 521202

<u>CERTIFICATE OF EXAMINATION</u>

This is to certify that the work entitled, "**OBJECT DETECTION WITH VOICE FEEDBACK**" is work of "**T.LEELAVATHI (N180197)** and here by accord our approval of it as a study carried out and presented in a manner required for its acceptance in third year of Bachelor of Technology for which it has been submitted. This approval does not necessarily endorse or accept every statement made, opinion expressed or conclusion drawn, as recorded in this thesis. It only signifies the acceptance of this thesis for the purpose for which it has been submitted.

----------------------	-----------------------
Dr.D.V.Nagarjuna Devi,	Examiner,
Project Supervisor,	Department of CSE,
Department of CSE,	RGUKT – Nuzvid.
RGUKT – Nuzvid.	

Rajiv Gandhi University of Knowledge Technologies – Nuzvid
Department of Computer Science and Engineering
(A.P. Government Act 18 of 2008)
RGUKT-NUZVID, Eluru Dist - 521202

DECLARATION

I **"T.LEELAVATHI (N180197)"** hereby declare that the project report entitled done by me under the guidance of **Dr.Nagarjuna Devi** is submitted for the partial fulfillment for the award of degree of Bachelor of Technology in Computer Science and Engineering during the academic session **February 2023- July 2023** at RGUKT-Nuzvid.

I also declare that this project is a result of my effort and has not been copied or imitated from any source. Citations from any websites are mentioned in the references. The results embodied in this project report have not been submitted to any other university or institute for the award of any degree or diploma.

Date:13-07-2023
Place:Nuzvid **T.LEELAVATHI(N180197)**

ACKNOWLEDGEMENT

I would like to express my profound gratitude and regards to my guide “ **Dr.D.V. Nagrjuna Devi** “ for her exemplary guidance, monitoring and constant encouragement to us throughout the B.Tech course.I shall always cherish the time spent with her during the course of this work due to the invaluable knowledge gained in the field of Data Science.

I am extremely grateful for the confidence bestowed in me and entrusting my project entitled **““OBJECT DETECTION WITH VOICE FEEDBACK”**. And also I am gratitude to Dr. Sadu Chiranjeevi (HOD of CSE) and other faculty members for being a source of inspiration and constant encouragement which helped me in completing the project successfully.

Finally, yet importantly, I would like to express my heartfelt thanks to our beloved God and parents for their blessings,my friends for their help and wishes for the successful completion of this project.

CONTENTS

ABSTRACT

Object Detection is a field of Computer Vision that detects instances of semantic objects in images/videos (by creating bounding boxes around them in our case). We can then convert the annotated text into voice responses and give the basic positions of the objects in the person/camera's view. In this object detection with voice feedback project , we propose a system that combines real-time object detection using the YOLO algorithm with audio feedback to assist visually impaired individuals in locating and identifying objects in their surroundings.

The object detection with voice feedback project proposes a system that will detect every possible day to day multiple objects on the other hand prompt a voice to alert person about the near as well as farthest objects around them.To get the audio Feedback gTTS (Google Text to Speech), python library used to convert statements into audio speech. Through an in-depth exploration of its methodology, applications, and societal impact, this abstract showcases how the fusion of object detection with voice feedback catalyzes a new era of accessible, interactive, and informative experiences.

1.Introduction

Due to the loss of vision or vision impairment, many people across the globe fail to study an experience the environment. To deal with everyday routine however they tend to develop some alternative approaches, they suffer from navigation complexities and troubles along with awkwardness in society. This object detection with voice feedback project application developed can detect the objects in people surroundings. The object detection with voice feedback model has been proposed which makes the use of smartphone, a common device available to anyone and used technology to make an application which can help the blind user detect objects in his surroundings and help him in navigating from one place to another. To avoid obstacles and to navigate easily visually impaired people generally depend on their sensory information. First variations of YOLO (You Only Look Once) algorithm are compared and then the best one is used according to result we get it by training it on COCO data set.

The aim of the project is an object recognition function with mobile application should be able to to detect certain objects from the camera and return audio output to announce what it is. In order to recognize object, machine learning ha to be involved.

1.1 Purpose

The purpose of object-based detection with voice feedback is to create an inclusive and interactive technology that leverages computer vision and auditory communication. By seamlessly identifying objects and translating visual cues into spoken descriptions, it aims to enhance accessibility for the visually impaired, provide immersive interactions, support education, enrich tourism experiences, revolutionize gaming, facilitate efficient information sharing, aid in emergencies, and drive innovation across various domains.

1.2 Overview

Object-based detection with voice feedback merges computer vision and auditory communication to swiftly identify objects and deliver spoken descriptions, offering immersive interactions and enriching experiences across education, tourism, gaming, and more.

1.3 Scope

The scope of object-based detection with voice feedback has a wide range of possibilities. It helps blind people, makes learning fun, enhances travel experiences, and makes games more exciting.

2 Analysis of Existing and Proposed Systems

2.1 Problem Statement

The problem statement for object-based detection with voice feedback revolves around addressing the limitations of traditional visual - centric technologies in providing inclusive and immersive interactions for all users, particularly the visually impaired. The challenge lies in developing a seamless and efficient system that can accurately identify objects in real time and convert visual information into spoken descriptions, bridging the gap between visual and auditory senses. The objective is to create an accessible and intuitive solution that enhances user engagement, learning, and overall experiences across diverse contexts, while also optimizing efficiency, accuracy, and customization.

2.2 Existed System

Even though there are similar systems out there, this project aims to make object detection with voice feedback even better. It wants to customize the experience, improve how it works, and focus on specific situations where it can be really useful. The goal is to create something more helpful and user-friendly than what's already available..

2.2.1 Disadvantages of Existed System

Accuracy and Reliability:

In some existing projects, there can be difficulties in accurately recognizing different objects in different situations. This can cause the voice feedback to be unreliable or incorrect

Limited Scope:

Certain systems could focus on specific use cases or environments, making them less versatile for diverse applications.

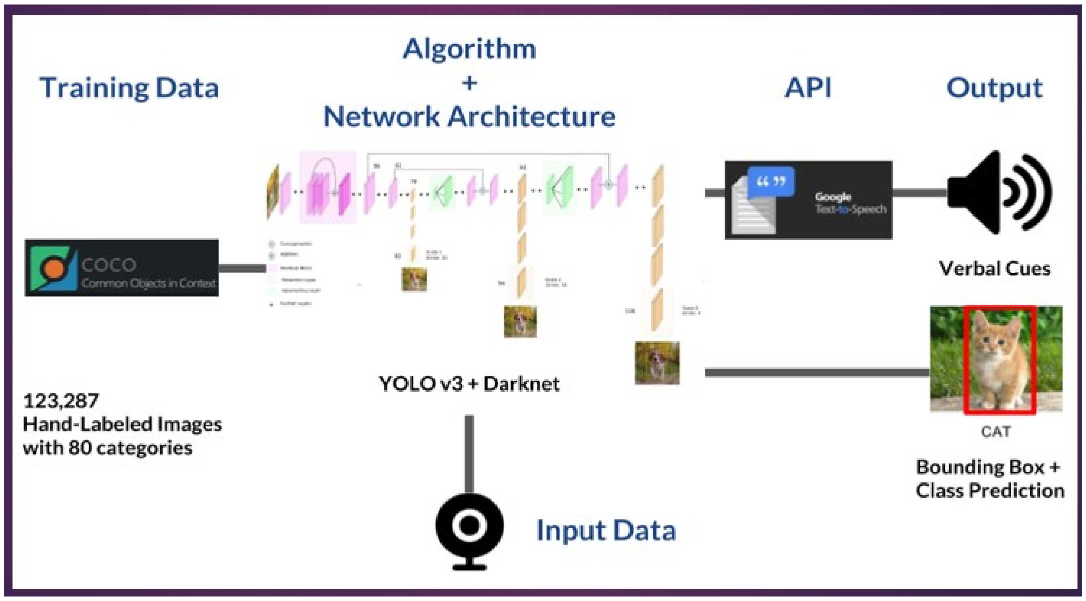
User Interface:

The user interface of some projects might be less intuitive or user-friendly, hindering a seamless and enjoyable user experience.

3. Proposed System

The proposed system wants to make something really useful. It will use smart computer programs to quickly figure out what objects are around us, like trees or cars. Then, it will use a special talking technology to tell us what those objects are, using words we can understand. This talking technology can be changed to sound how we like and say more or less information. The system will work fast and be accurate, and we can use it in different ways, like when we're learning, playing games, or even traveling. This application making it easier for us to understand and enjoy our surroundings.

1. **Capture Video Frames:** Use a camera to capture real-time video frames.
2. **Apply Object Detection:** Utilize the YOLO algorithm to identify objects within the frames.
3. **Convert Labels to Speech:** Translate object labels into spoken text using gTTS (Google Text-to-Speech).
4. **Generate Audio Feedback:** Create an audio segment describing the detected objects using the converted text.
5. **Customize Feedback**: Allow users to personalize the voice feedback system based on their preferences.


Training Data
Algorithm + Network Architecture
API
Output
COCO Common Objects in Context
Google Text-to-Speech
Verbal Cues
YOLO v3 + Darknet
123,287 Hand-Labeled Images with 80 categories
CAT
Bounding Box + Class Prediction
Input Data

3.1 Advantages of Proposed System

i. Instantly recognizing objects as soon as they appear.
ii. Easy to use and understand for all types of users.
iii. Enhancing travel experiences with informative descriptions of landmarks and attractions.
iv. Making information and technology more available to everyone including those with disabilities.
v. Adapting the technology to suit individual preferences and needs.

4. Requirements

4.1 Hardware Requirements

Operating System : 64 bit windows
RAM : 4 GB(approx)
Disk : 500GB
Camera : Web Camera 10 Megapixels
Processor : I5

4.2 Software Requirements

Operating System : 64 bit windows
Code Editor : Jupyter Notebook, Google colab
Programming Language : Python

4.3 Technologies

4.3.1 Python

It is Commonly used for developing websites and software,task automation,data analysis and data visualization. In python we used different kinds of libraries.

4.3.1.1 Numpy

It is very useful for fundamental scientific computations in Machine Learning.NumPy is a very popular python library with the help of a large collection of high-level mathematical functions.

4.3.1.2 gTTS (Google Text-to-Speech)

The gTTS library allows you to convert text into speech using Google's text-to-speech API. It's used to generate voice feedback based on the detected objects' labels and descriptions.

4.3.1.3 Matplotlib

Matplotlib is a versatile and widely used visualization library in Python, allowing you to generate a variety of graphs and plots that vividly represent object detection results.

4.3.1.4 OpenCV

OpenCV is a Python library that allows you to perform image processing and computer vision tasks. It provides a wide range of features, including object detection, face recognition, and tracking. In this OpenCV Tutorial in Python, we'll be learning more about the library.

4.3.1.5 IPython

IPython, short for "Interactive Python," is an enhanced interactive shell and Jupyter Notebook interface for Python programming. It provides features like code autocompletion, history management, rich media display, and the ability to combine code, text, and visualizations in a single document, enhancing the interactive and exploratory coding experience.

4.3.1.6 IO

Python, io stands for "input/output" and is a module that provides tools for working with input and output streams, such as reading from and writing to files, strings, and other data sources. It offers a unified interface to handle various types of data streams, making it easier to manage data input and output operations.

4.3.1.7 Python Imaging Library (PIL)

The Python Imaging Library (PIL), now Pillow, is a powerful Python library for working with images, offering functions for tasks like opening, editing, and manipulating various image formats. It's commonly used for tasks such as resizing, cropping, and applying filters to images.

4.3.1.8 YOLO (Darknet or YOLOv4-TF):

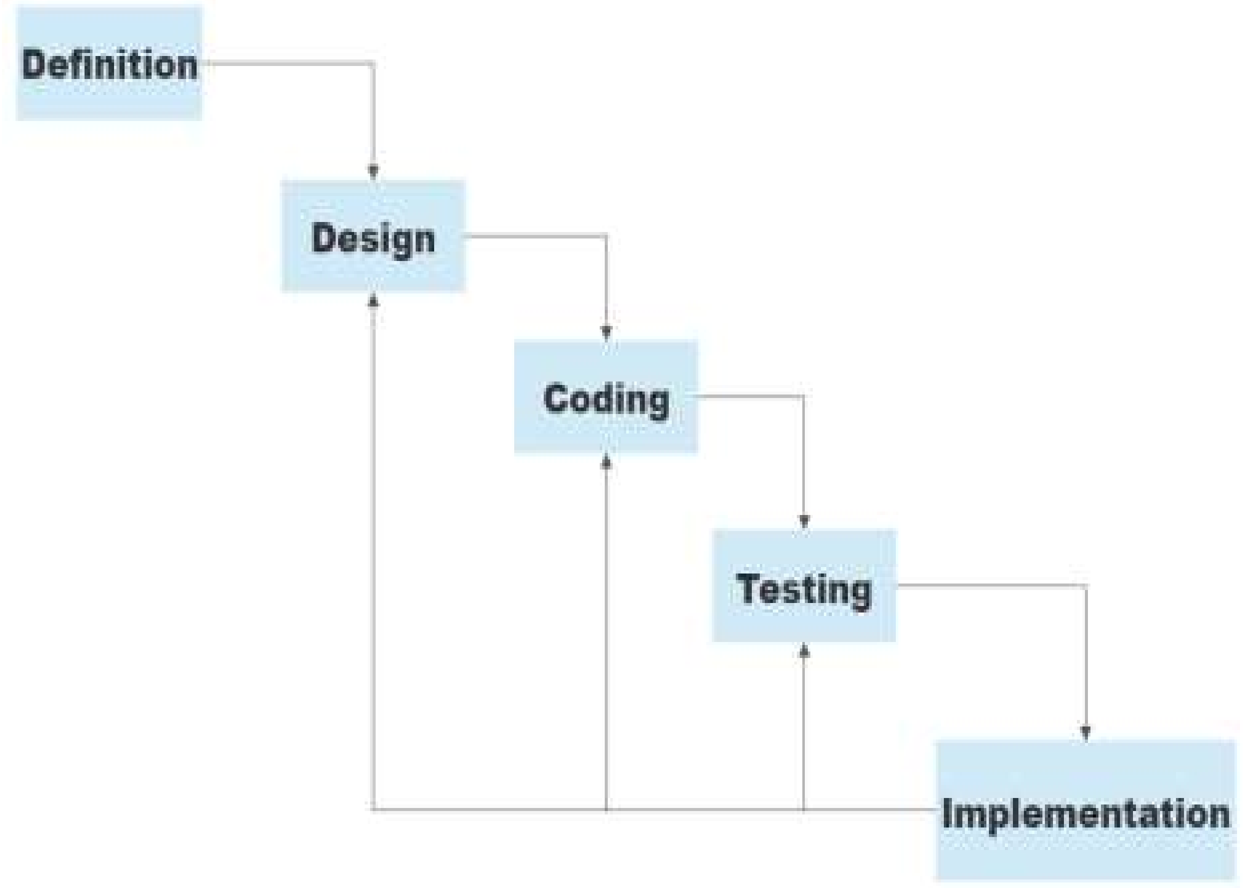
YOLOv4, for instance, can be implemented using Darknet or TensorFlow. YOLO is crucial for object detection, and the respective libraries provide pre-trained models and functions for object detection tasks.

5.Methodology

The methodology for object-based detection with voice feedback involves integrating YOLO object detection, text-to-speech conversion, and user customization to create a seamless system that accurately identifies objects in real-time and converts visual information into spoken descriptions.

5.1 Procedure Identification

The procedure for object-based detection with voice feedback encompasses several interconnected stages, the object-based detection with voice feedback system, from capturing camera input to providing informative audio feedback based on detected objects.



5.1.1 Definition Stage

The first phase of this project is to try to capture what the system will do (its requirements) based on milestones. This is to ensure the system will be on track. Here, comprehensive research has been done to study the basic concept of the system itself. During this stage, the problem definition and problem statement of the system have been started.

5.1.2 Design Stage

The second stage determines how it will be designed. All the functionalities required are designed in this phase.

5.1.3 Coding Stage

In the middle of the stage the actual programming started. Here, the appropriate software and hardware tools as mentioned below are being applied in order to facilitate the project development process.

5.1.4 Testing Phase

The fourth phase is the full system testing where the stages of testing like unit testing, integration testing, system testing and user acceptance testing are involved in order to ensure quality of the system.

5.1.5 Implementation Stage

The final phase is focused on implementation tasks such as go-live, training, and documentation. Here, the documentation is being prepared to conclude on the overall research and experiment, which is basically to know which method can be applied.

6.Main Functionality

The main functionalities in this project, "Object Detection with Voice Feedback," involve the integration of computer vision techniques and audio processing to create an interactive system that can detect objects in real-time camera feeds and provide auditory descriptions of those objects.

6.1 Real-Time Object Detection:

Real-Time Object Detection involves using the YOLO (You Only Look Once) algorithm to accurately and swiftly identify objects within live camera frames, enabling instant and dynamic recognition of various items or entities in the environment..

6.2 Voice Feedback:

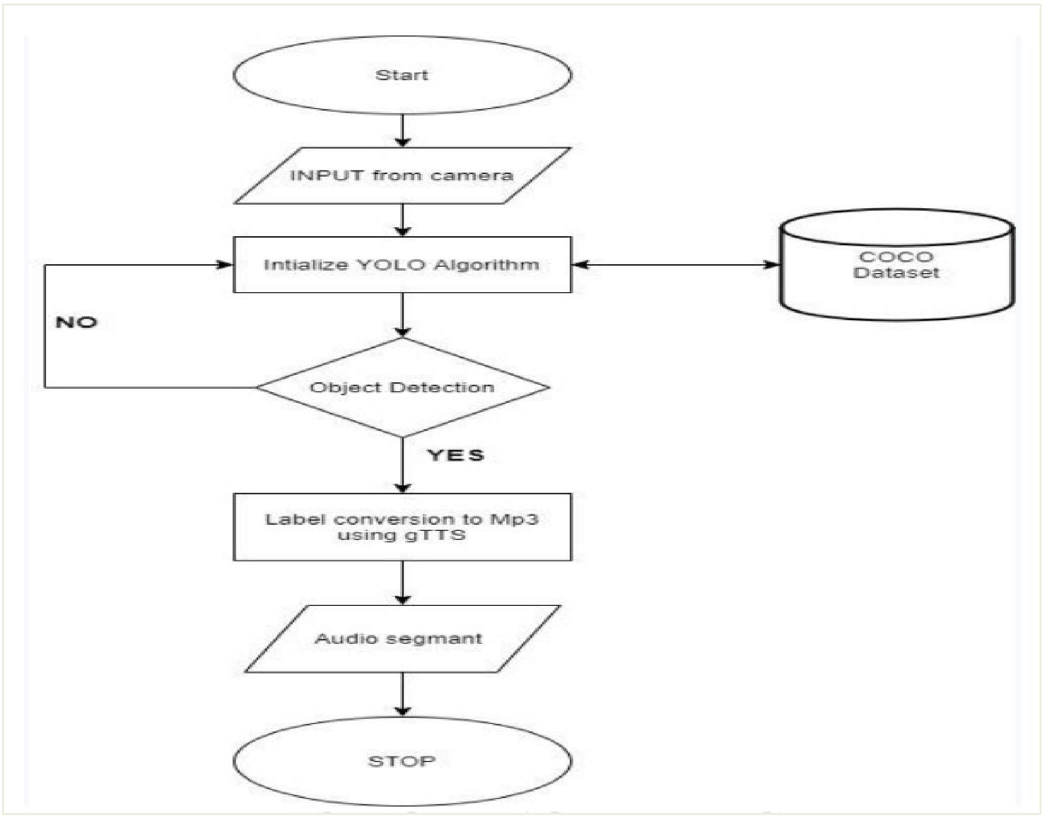
Voice Feedback entails transforming the labels of detected objects into spoken words using the gTTS (Google Text-to-Speech) technology. This process enables the system to deliver auditory information about the recognized objects, enhancing accessibility and providing valuable insights to users, especially those who are visually impaired.

6.3 Customizable Voice Output:

Customizable Voice Output involves giving users the ability to tailor the voice feedback according to their individual preferences. This feature empowers users to adjust the tone, speed, and style of the voice, enhancing the user experience and making the system more adaptable to their specific needs and comfort.

7. Architecture

7.1 Architecture diagram



7.1.1 Starting Point

The Start shape is the initial point from which the flowcharts steps and actions will follow.

7.1.2 Input from camera

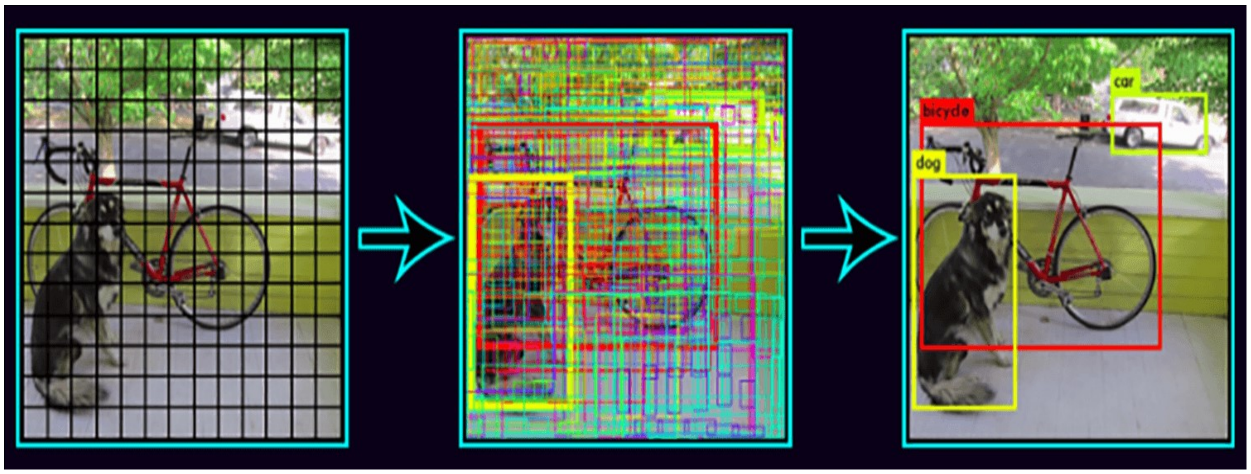
This step involves accessing the camera feed to capture individual frames. Process each captured frame as input for the object detection system.

7.1.3 COCO dataset

COCO (Common Objects in Context) is a large-scale dataset containing images with objects in complex scenes.It is used for evaluating and benchmarking object detection, instance segmentation, and other vision tasks.The dataset consists of a vast collection of images, each annotated with object labels, bounding box coordinates, and more.

7.1.4 Initialize YOLO algorithm

Set up and configure the pre-trained YOLO model for object detection. Initializing the YOLO (You Only Look Once) algorithm involves setting up and configuring the pre-trained YOLO model to perform object detection on input images or frames from a camera.



7.1.5 Object Detection :

After initializing the YOLO algorithm, apply the model to the captured frames from the camera.YOLO divides the input image into a grid and makes predictions for bounding boxes and class probabilities for objects within each grid cell.

```
img_height, img_width, _ = img.shape
width_ratio = img_width/width
height_ratio = img_height/height
```

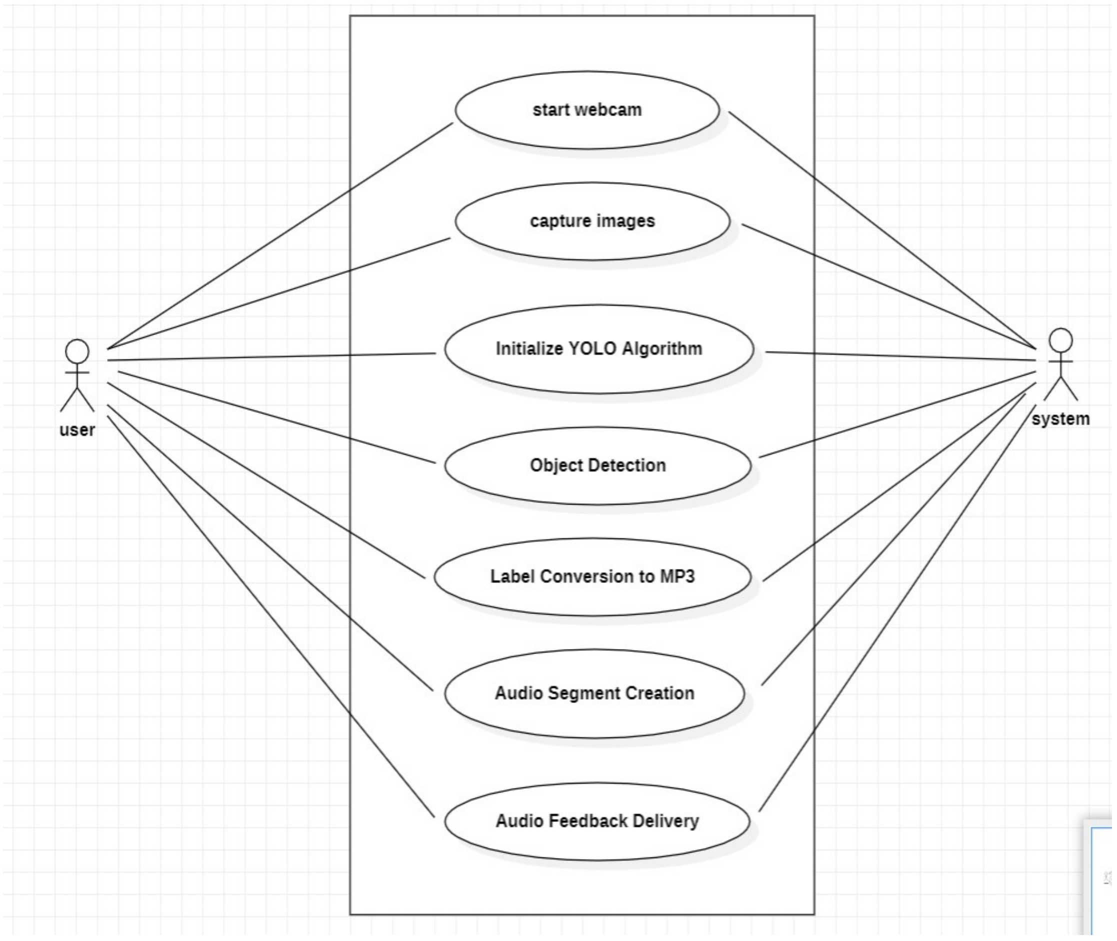
The output of this step includes a list of detected objects along with their class labels and accurate bounding box coordinates.

7.1.6 Generate Audio Segment:

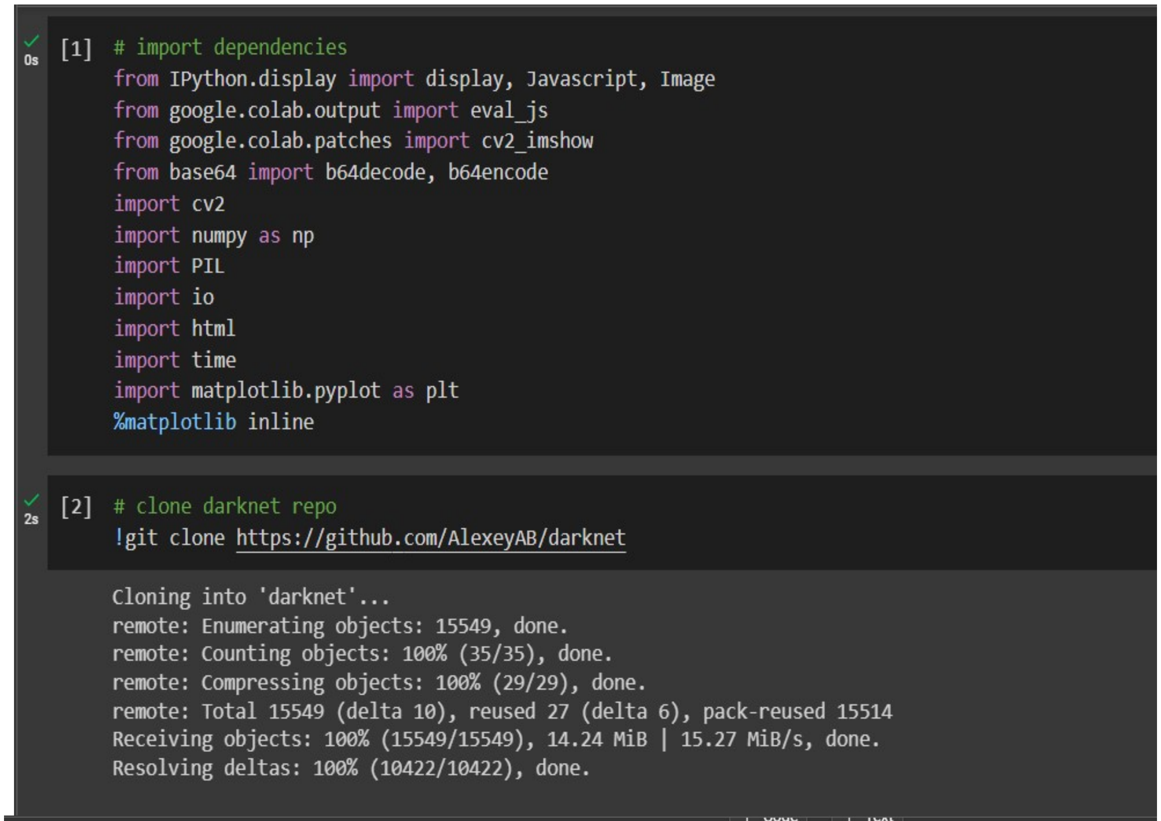
Use the converted text to generate an audio segment describing the detected objects.

7.17 Stop point: This step involves defining conditions or criteria that determine when the entire process should stop or come to an end, such as user input or reaching the end of the camera feed**.**

7.2 Use case diagram



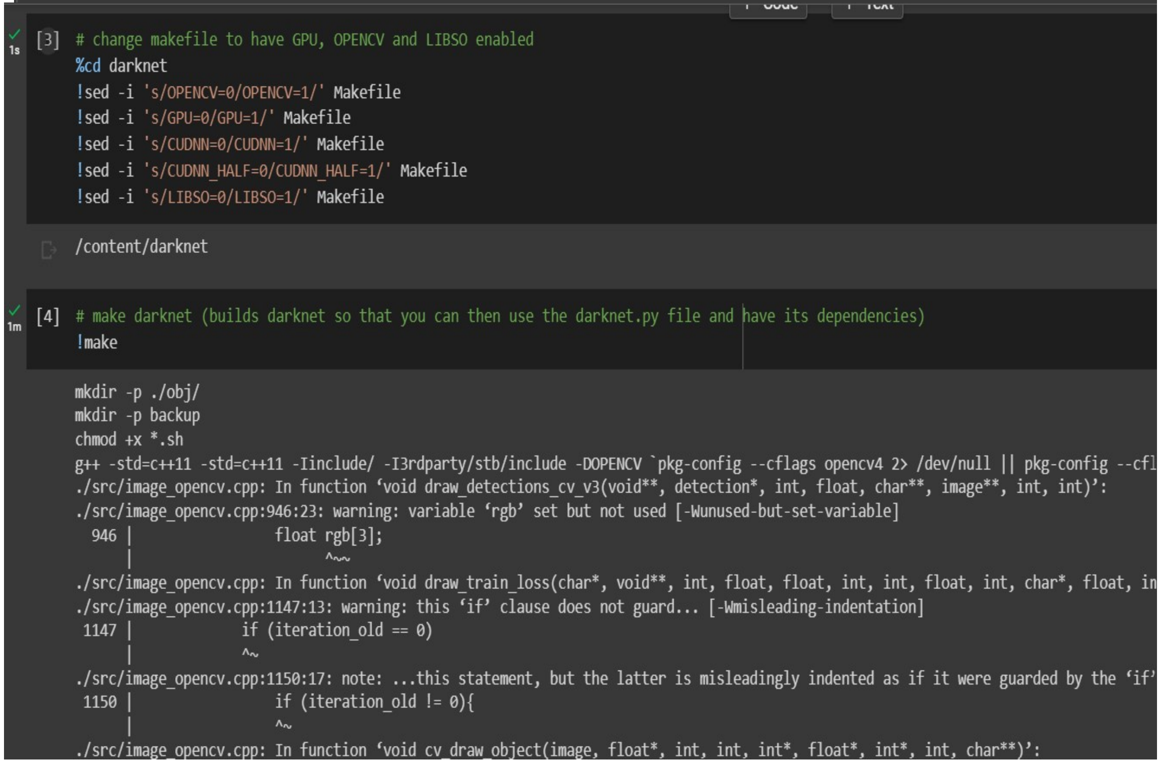
8.Implementation code:



```
[1] # import dependencies
    from IPython.display import display, Javascript, Image
    from google.colab.output import eval_js
    from google.colab.patches import cv2_imshow
    from base64 import b64decode, b64encode
    import cv2
    import numpy as np
    import PIL
    import io
    import html
    import time
    import matplotlib.pyplot as plt
    %matplotlib inline
```

```
[2] # clone darknet repo
    !git clone https://github.com/AlexeyAB/darknet
```

```
Cloning into 'darknet'...
remote: Enumerating objects: 15549, done.
remote: Counting objects: 100% (35/35), done.
remote: Compressing objects: 100% (29/29), done.
remote: Total 15549 (delta 10), reused 27 (delta 6), pack-reused 15514
Receiving objects: 100% (15549/15549), 14.24 MiB | 15.27 MiB/s, done.
Resolving deltas: 100% (10422/10422), done.
```

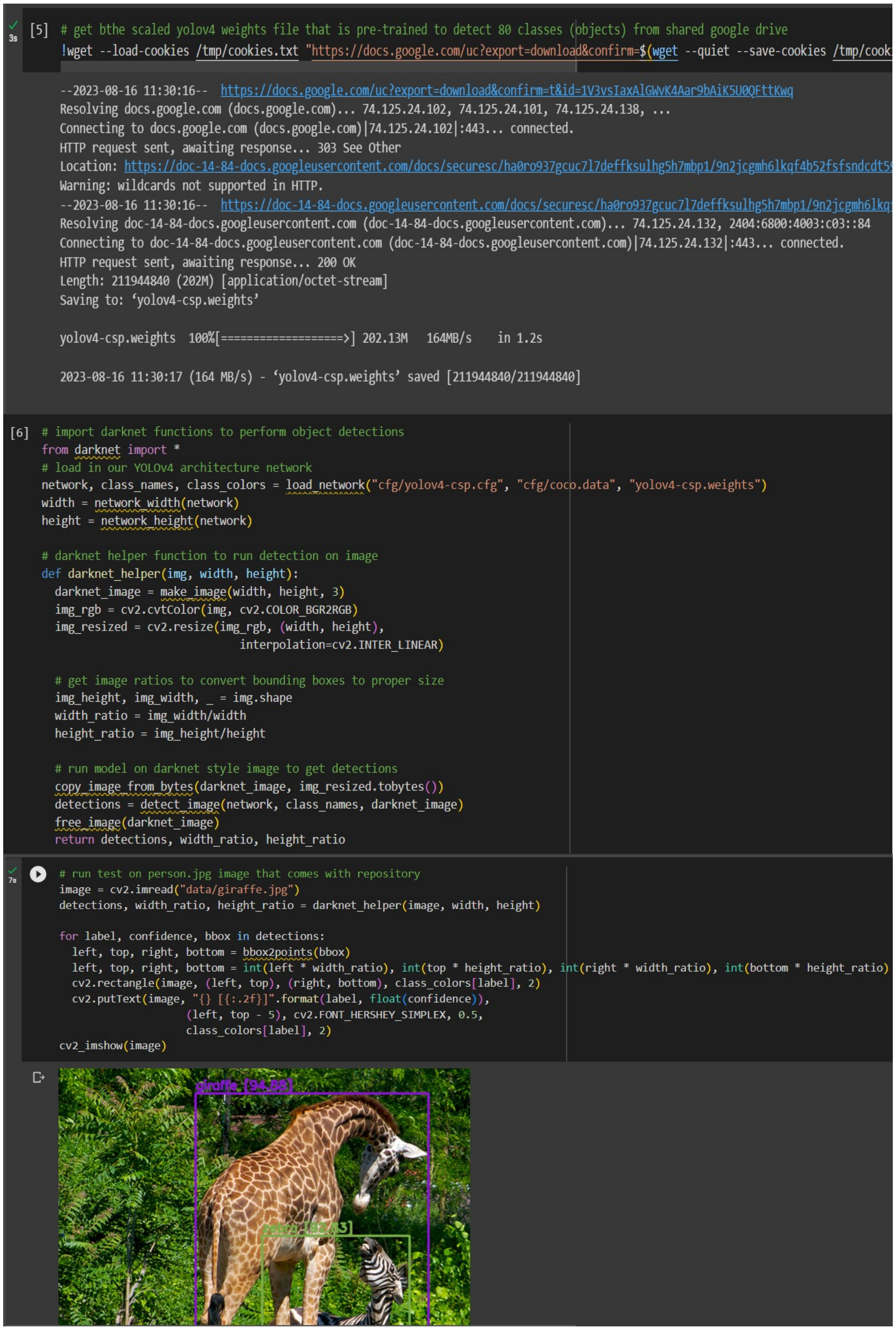


```
[3] # change makefile to have GPU, OPENCV and LIBSO enabled
    %cd darknet
    !sed -i 's/OPENCV=0/OPENCV=1/' Makefile
    !sed -i 's/GPU=0/GPU=1/' Makefile
    !sed -i 's/CUDNN=0/CUDNN=1/' Makefile
    !sed -i 's/CUDNN_HALF=0/CUDNN_HALF=1/' Makefile
    !sed -i 's/LIBSO=0/LIBSO=1/' Makefile
```

```
/content/darknet
```

```
[4] # make darknet (builds darknet so that you can then use the darknet.py file and have its dependencies)
    !make
```

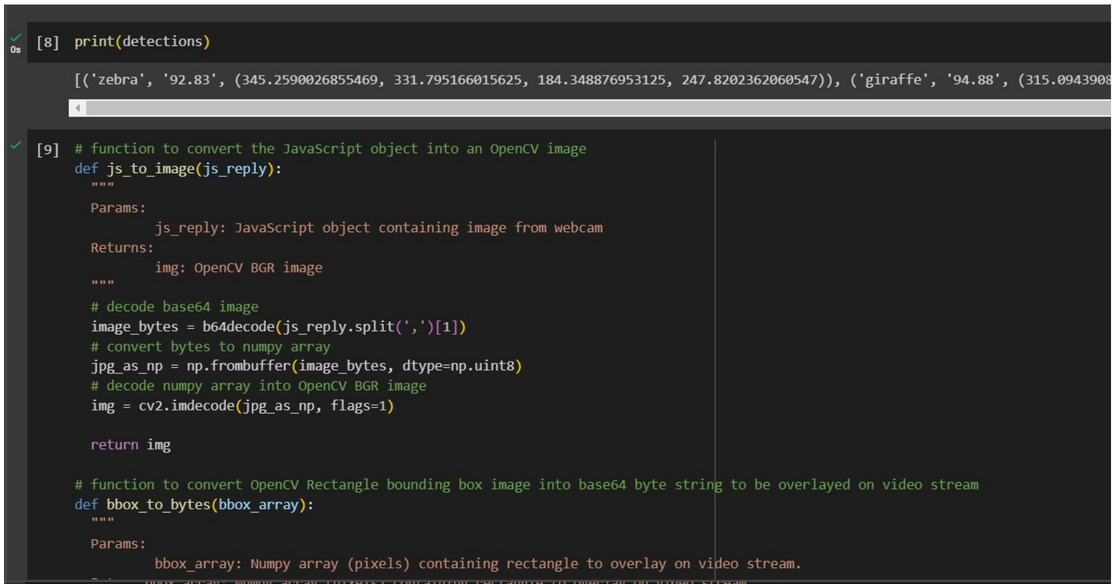
```
mkdir -p ./obj/
mkdir -p backup
chmod +x *.sh
g++ -std=c++11 -std=c++11 -Iinclude/ -I3rdparty/stb/include -DOPENCV `pkg-config --cflags opencv4 2> /dev/null || pkg-config --cfl
./src/image_opencv.cpp: In function ‘void draw_detections_cv_v3(void**, detection*, int, float, char**, image**, int, int)’:
./src/image_opencv.cpp:946:23: warning: variable ‘rgb’ set but not used [-Wunused-but-set-variable]
  946 |                 float rgb[3];
      |                       ^~~
./src/image_opencv.cpp: In function ‘void draw_train_loss(char*, void**, int, float, float, int, int, float, int, char*, float, in
./src/image_opencv.cpp:1147:13: warning: this ‘if’ clause does not guard... [-Wmisleading-indentation]
 1147 |             if (iteration_old == 0)
      |             ^~
./src/image_opencv.cpp:1150:17: note: ...this statement, but the latter is misleadingly indented as if it were guarded by the ‘if’
 1150 |                 if (iteration_old != 0){
      |                 ^~
./src/image_opencv.cpp: In function ‘void cv_draw_object(image, float*, int, int, int*, float*, int*, int, char**)’:
```



```
[5] # get bthe scaled yolov4 weights file that is pre-trained to detect 80 classes (objects) from shared google drive
    !wget --load-cookies /tmp/cookies.txt "https://docs.google.com/uc?export=download&confirm=$(wget --quiet --save-cookies /tmp/cook

    --2023-08-16 11:30:16--  https://docs.google.com/uc?export=download&confirm=t&id=1V3vsIaxAlGWvK4Aar9bAiK5U0QFttKwq
    Resolving docs.google.com (docs.google.com)... 74.125.24.102, 74.125.24.101, 74.125.24.138, ...
    Connecting to docs.google.com (docs.google.com)|74.125.24.102|:443... connected.
    HTTP request sent, awaiting response... 303 See Other
    Location: https://doc-14-84-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc7l7deffksulhg5h7mbp1/9n2jcgmh6lkqf4b52fsfsndcdt5
    Warning: wildcards not supported in HTTP.
    --2023-08-16 11:30:16--  https://doc-14-84-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc7l7deffksulhg5h7mbp1/9n2jcgmh6lkq
    Resolving doc-14-84-docs.googleusercontent.com (doc-14-84-docs.googleusercontent.com)... 74.125.24.132, 2404:6800:4003:c03::84
    Connecting to doc-14-84-docs.googleusercontent.com (doc-14-84-docs.googleusercontent.com)|74.125.24.132|:443... connected.
    HTTP request sent, awaiting response... 200 OK
    Length: 211944840 (202M) [application/octet-stream]
    Saving to: 'yolov4-csp.weights'

    yolov4-csp.weights  100%[==================>] 202.13M   164MB/s    in 1.2s

    2023-08-16 11:30:17 (164 MB/s) - 'yolov4-csp.weights' saved [211944840/211944840]
```

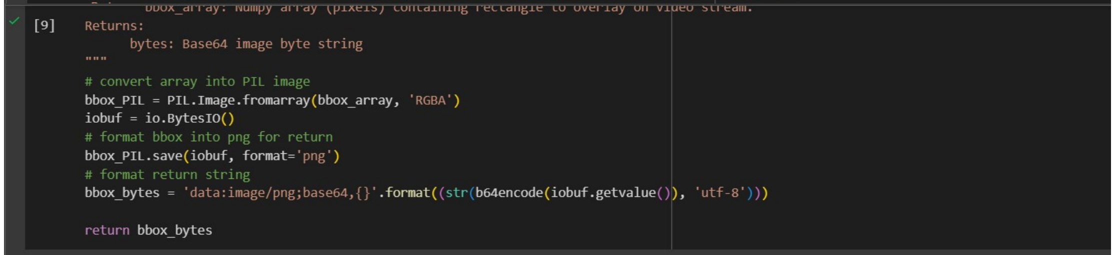
```
[6] # import darknet functions to perform object detections
    from darknet import *
    # load in our YOLOv4 architecture network
    network, class_names, class_colors = load_network("cfg/yolov4-csp.cfg", "cfg/coco.data", "yolov4-csp.weights")
    width = network_width(network)
    height = network_height(network)

    # darknet helper function to run detection on image
    def darknet_helper(img, width, height):
      darknet_image = make_image(width, height, 3)
      img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
      img_resized = cv2.resize(img_rgb, (width, height),
                                  interpolation=cv2.INTER_LINEAR)

      # get image ratios to convert bounding boxes to proper size
      img_height, img_width, _ = img.shape
      width_ratio = img_width/width
      height_ratio = img_height/height

      # run model on darknet style image to get detections
      copy_image_from_bytes(darknet_image, img_resized.tobytes())
      detections = detect_image(network, class_names, darknet_image)
      free_image(darknet_image)
      return detections, width_ratio, height_ratio
```

```
# run test on person.jpg image that comes with repository
image = cv2.imread("data/giraffe.jpg")
detections, width_ratio, height_ratio = darknet_helper(image, width, height)

for label, confidence, bbox in detections:
  left, top, right, bottom = bbox2points(bbox)
  left, top, right, bottom = int(left * width_ratio), int(top * height_ratio), int(right * width_ratio), int(bottom * height_ratio)
  cv2.rectangle(image, (left, top), (right, bottom), class_colors[label], 2)
  cv2.putText(image, "{} [{:.2f}]".format(label, float(confidence)),
                    (left, top - 5), cv2.FONT_HERSHEY_SIMPLEX, 0.5,
                    class_colors[label], 2)
cv2_imshow(image)
```



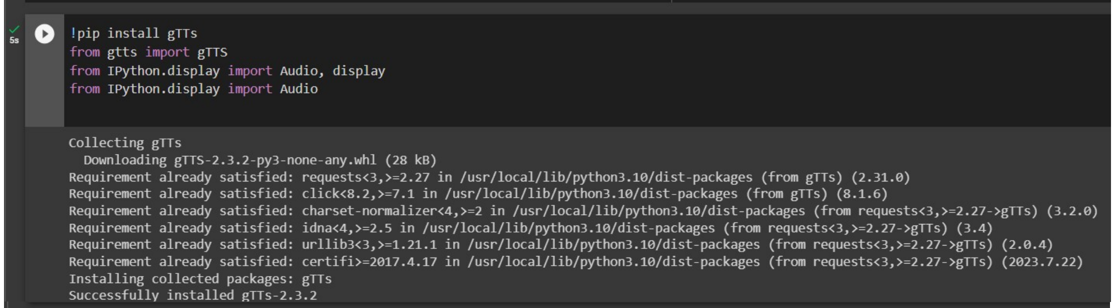
```
[8]  print(detections)

     [('zebra', '92.83', (345.2590026855469, 331.795166015625, 184.348876953125, 247.8202362060547)), ('giraffe', '94.88', (315.0943908
```

```
[9]  # function to convert the JavaScript object into an OpenCV image
     def js_to_image(js_reply):
       """
       Params:
               js_reply: JavaScript object containing image from webcam
       Returns:
               img: OpenCV BGR image
       """
       # decode base64 image
       image_bytes = b64decode(js_reply.split(',')[1])
       # convert bytes to numpy array
       jpg_as_np = np.frombuffer(image_bytes, dtype=np.uint8)
       # decode numpy array into OpenCV BGR image
       img = cv2.imdecode(jpg_as_np, flags=1)

       return img

     # function to convert OpenCV Rectangle bounding box image into base64 byte string to be overlayed on video stream
     def bbox_to_bytes(bbox_array):
       """
       Params:
               bbox_array: Numpy array (pixels) containing rectangle to overlay on video stream.
```

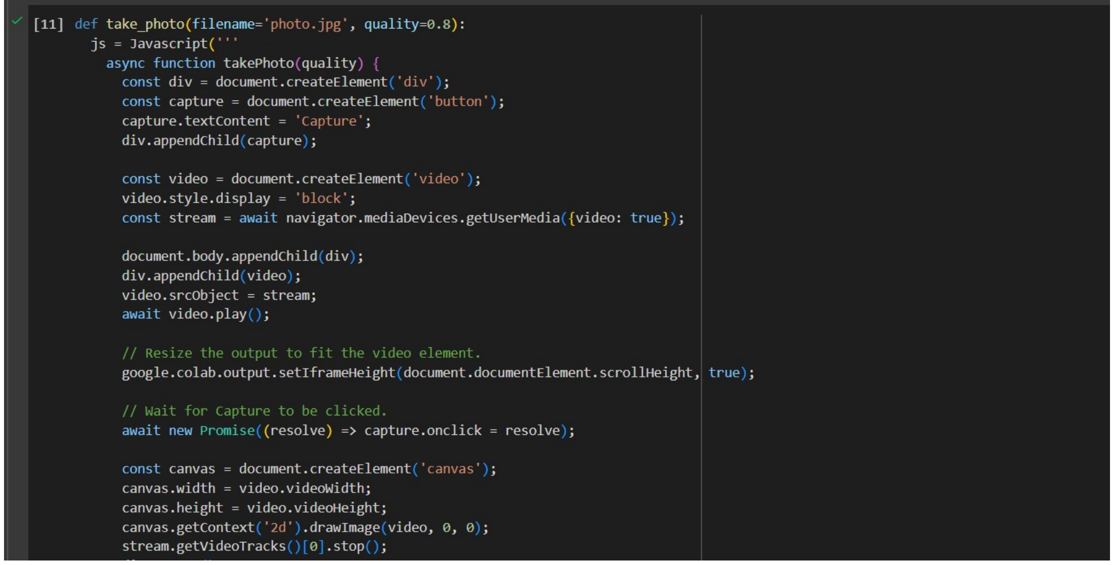


```
               bbox_array: Numpy array (pixels) containing rectangle to overlay on video stream.
```

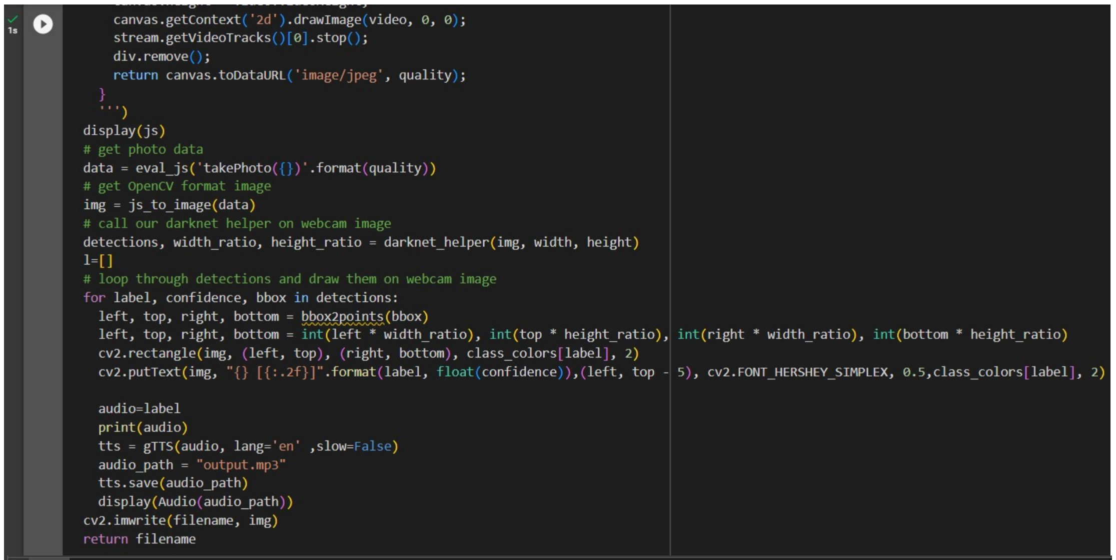
```
[9]    Returns:
             bytes: Base64 image byte string
       """
       # convert array into PIL image
       bbox_PIL = PIL.Image.fromarray(bbox_array, 'RGBA')
       iobuf = io.BytesIO()
       # format bbox into png for return
       bbox_PIL.save(iobuf, format='png')
       # format return string
       bbox_bytes = 'data:image/png;base64,{}'.format((str(b64encode(iobuf.getvalue()), 'utf-8')))

       return bbox_bytes
```



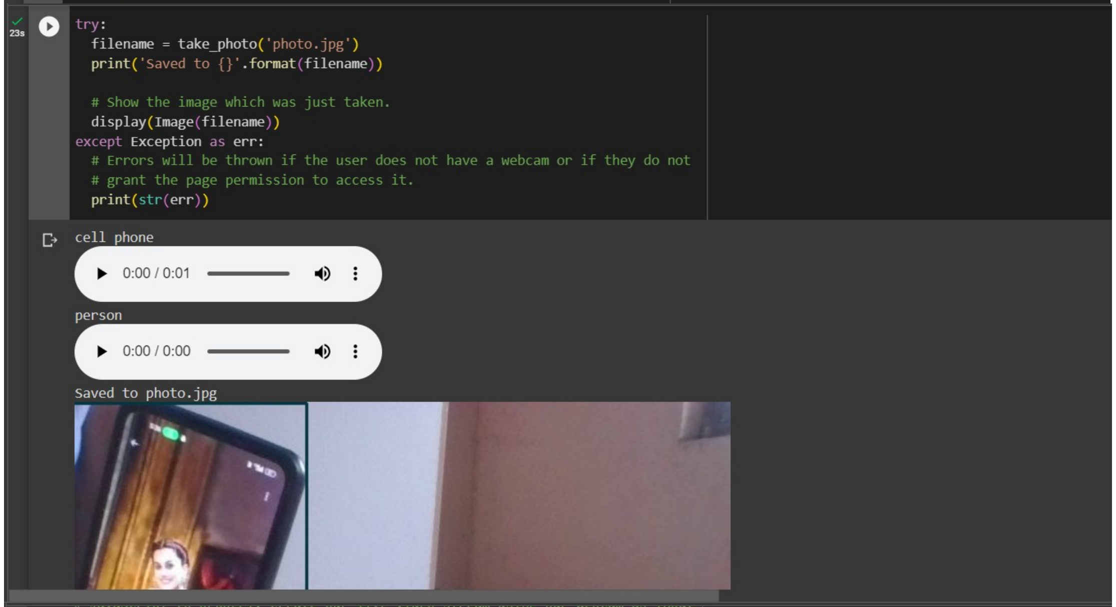
```
!pip install gTTs
from gtts import gTTS
from IPython.display import Audio, display
from IPython.display import Audio

Collecting gTTs
  Downloading gTTS-2.3.2-py3-none-any.whl (28 kB)
Requirement already satisfied: requests<3,>=2.27 in /usr/local/lib/python3.10/dist-packages (from gTTs) (2.31.0)
Requirement already satisfied: click<8.2,>=7.1 in /usr/local/lib/python3.10/dist-packages (from gTTs) (8.1.6)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.27->gTTs) (3.2.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.27->gTTs) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.27->gTTs) (2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.27->gTTs) (2023.7.22)
Installing collected packages: gTTs
Successfully installed gTTs-2.3.2
```



```
[11] def take_photo(filename='photo.jpg', quality=0.8):
       js = Javascript('''
         async function takePhoto(quality) {
           const div = document.createElement('div');
           const capture = document.createElement('button');
           capture.textContent = 'Capture';
           div.appendChild(capture);

           const video = document.createElement('video');
           video.style.display = 'block';
           const stream = await navigator.mediaDevices.getUserMedia({video: true});

           document.body.appendChild(div);
           div.appendChild(video);
           video.srcObject = stream;
           await video.play();

           // Resize the output to fit the video element.
           google.colab.output.setIframeHeight(document.documentElement.scrollHeight, true);

           // Wait for Capture to be clicked.
           await new Promise((resolve) => capture.onclick = resolve);

           const canvas = document.createElement('canvas');
           canvas.width = video.videoWidth;
           canvas.height = video.videoHeight;
           canvas.getContext('2d').drawImage(video, 0, 0);
           stream.getVideoTracks()[0].stop();
```

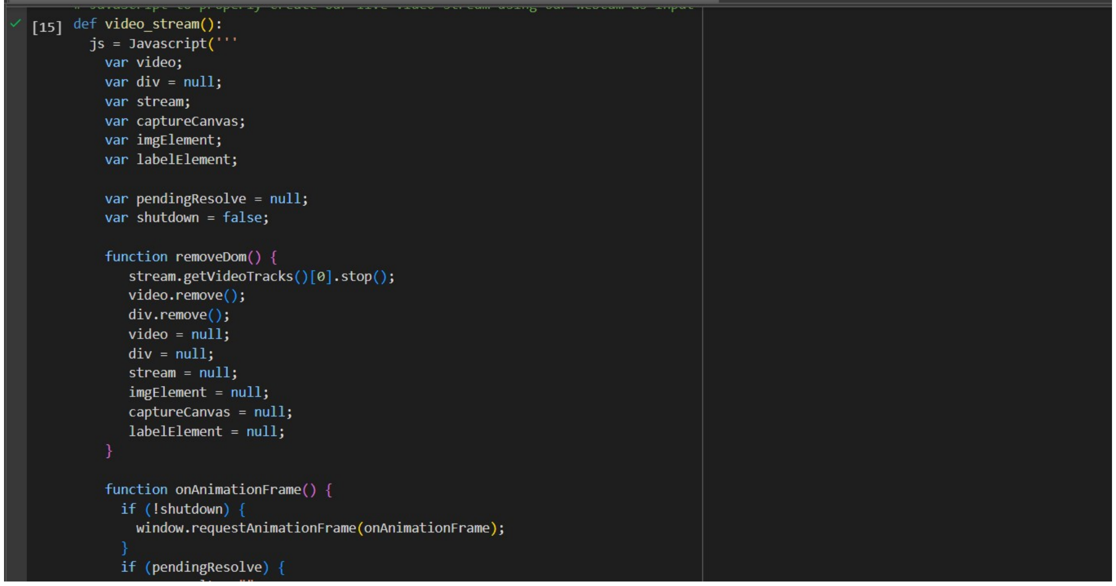


```
      canvas.getContext('2d').drawImage(video, 0, 0);
      stream.getVideoTracks()[0].stop();
      div.remove();
      return canvas.toDataURL('image/jpeg', quality);
    }
    ''')
  display(js)
  # get photo data
  data = eval_js('takePhoto({})'.format(quality))
  # get OpenCV format image
  img = js_to_image(data)
  # call our darknet helper on webcam image
  detections, width_ratio, height_ratio = darknet_helper(img, width, height)
  l=[]
  # loop through detections and draw them on webcam image
  for label, confidence, bbox in detections:
    left, top, right, bottom = bbox2points(bbox)
    left, top, right, bottom = int(left * width_ratio), int(top * height_ratio), int(right * width_ratio), int(bottom * height_ratio)
    cv2.rectangle(img, (left, top), (right, bottom), class_colors[label], 2)
    cv2.putText(img, "{} [{:.2f}]".format(label, float(confidence)),(left, top - 5), cv2.FONT_HERSHEY_SIMPLEX, 0.5,class_colors[label], 2)

    audio=label
    print(audio)
    tts = gTTS(audio, lang='en' ,slow=False)
    audio_path = "output.mp3"
    tts.save(audio_path)
    display(Audio(audio_path))
  cv2.imwrite(filename, img)
  return filename
```

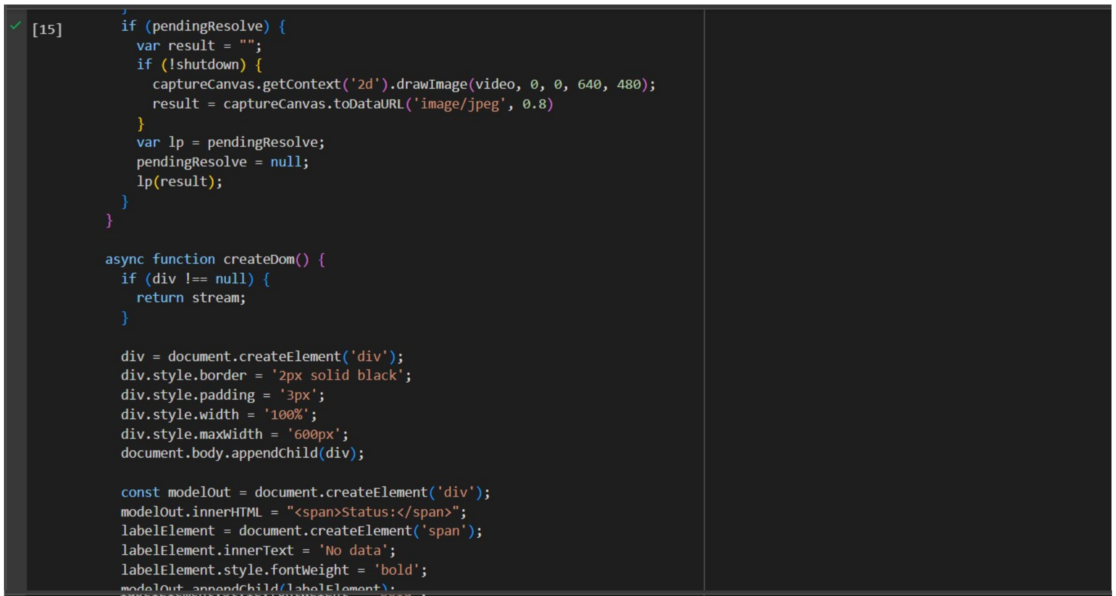


```
try:
  filename = take_photo('photo.jpg')
  print('Saved to {}'.format(filename))

  # Show the image which was just taken.
  display(Image(filename))
except Exception as err:
  # Errors will be thrown if the user does not have a webcam or if they do not
  # grant the page permission to access it.
  print(str(err))
```

```
cell phone
```
0:00 / 0:01
```
person
```
0:00 / 0:00
```
Saved to photo.jpg
```

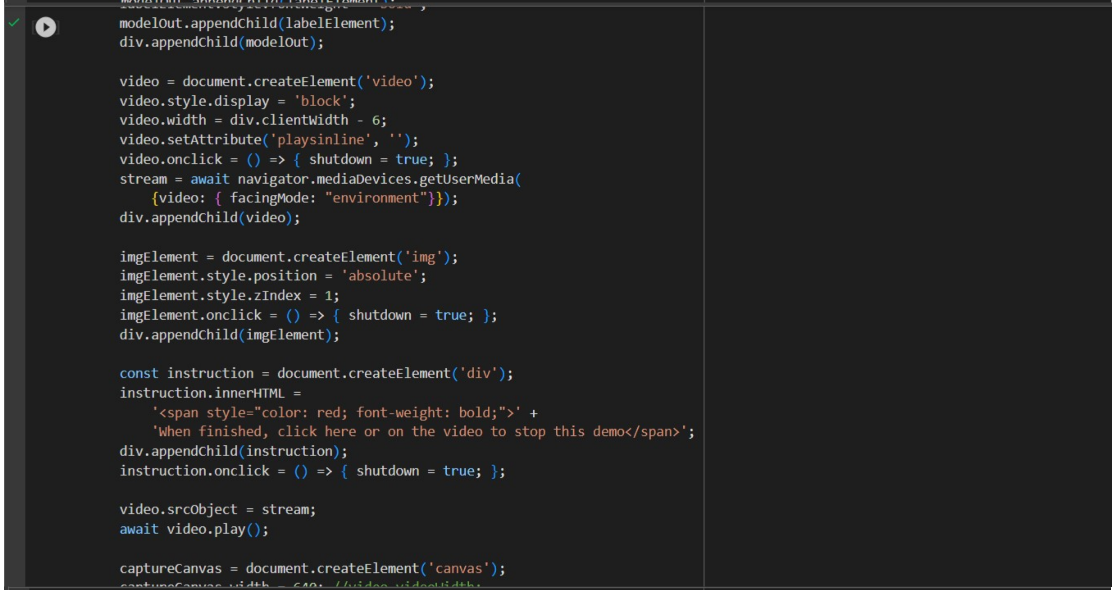


```
[15] def video_stream():
       js = Javascript('''
         var video;
         var div = null;
         var stream;
         var captureCanvas;
         var imgElement;
         var labelElement;

         var pendingResolve = null;
         var shutdown = false;

         function removeDom() {
            stream.getVideoTracks()[0].stop();
            video.remove();
            div.remove();
            video = null;
            div = null;
            stream = null;
            imgElement = null;
            captureCanvas = null;
            labelElement = null;
         }

         function onAnimationFrame() {
           if (!shutdown) {
             window.requestAnimationFrame(onAnimationFrame);
           }
           if (pendingResolve) {
```

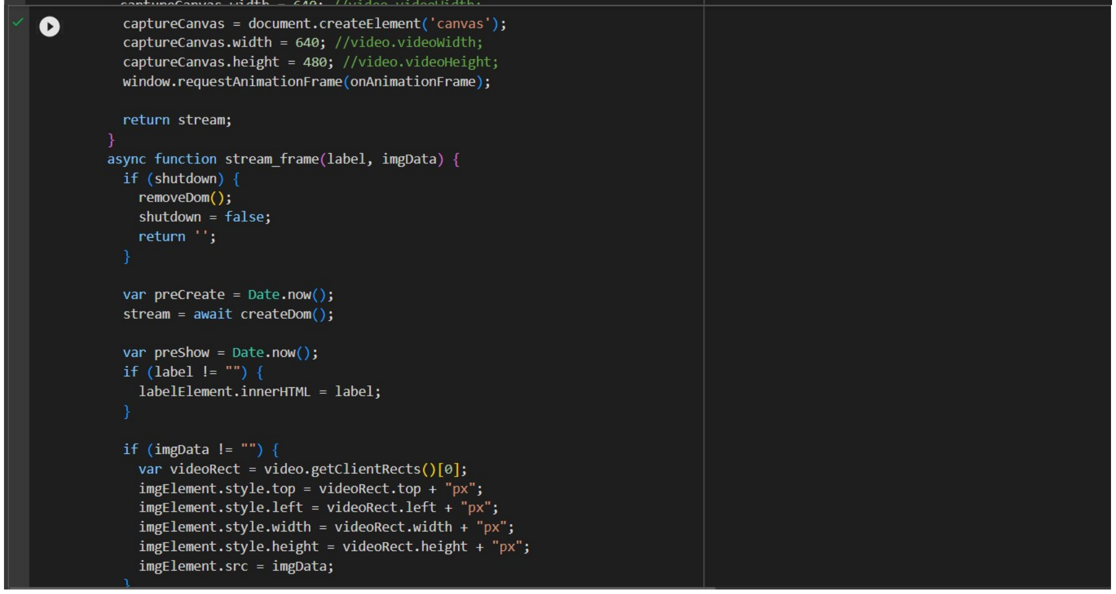


[15]

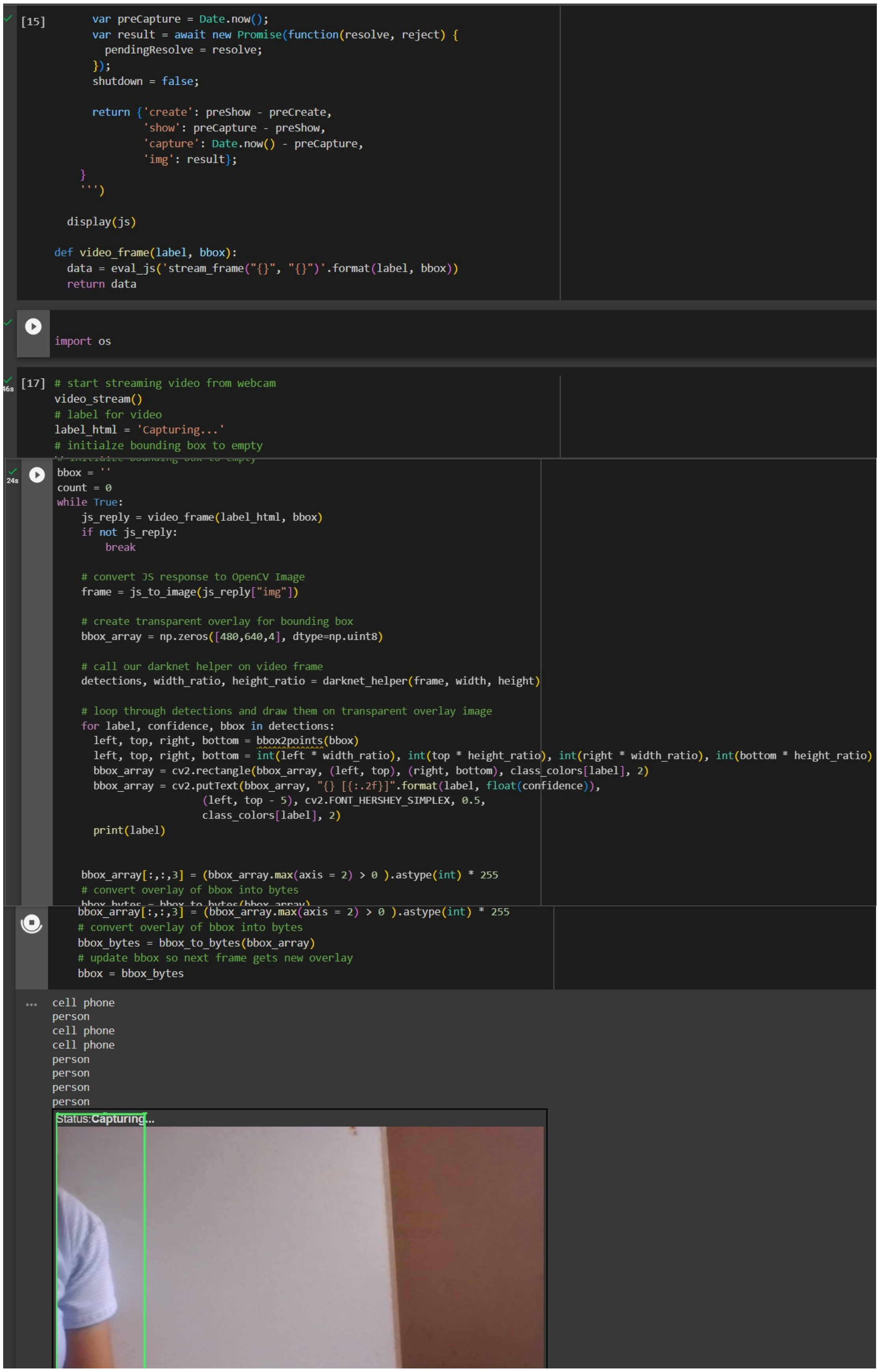
```
    if (pendingResolve) {
      var result = "";
      if (!shutdown) {
        captureCanvas.getContext('2d').drawImage(video, 0, 0, 640, 480);
        result = captureCanvas.toDataURL('image/jpeg', 0.8)
      }
      var lp = pendingResolve;
      pendingResolve = null;
      lp(result);
    }
  }

  async function createDom() {
    if (div !== null) {
      return stream;
    }

    div = document.createElement('div');
    div.style.border = '2px solid black';
    div.style.padding = '3px';
    div.style.width = '100%';
    div.style.maxWidth = '600px';
    document.body.appendChild(div);

    const modelOut = document.createElement('div');
    modelOut.innerHTML = "<span>Status:</span>";
    labelElement = document.createElement('span');
    labelElement.innerText = 'No data';
    labelElement.style.fontWeight = 'bold';
```



```
    modelOut.appendChild(labelElement);
    div.appendChild(modelOut);

    video = document.createElement('video');
    video.style.display = 'block';
    video.width = div.clientWidth - 6;
    video.setAttribute('playsinline', '');
    video.onclick = () => { shutdown = true; };
    stream = await navigator.mediaDevices.getUserMedia(
        {video: { facingMode: "environment"}});
    div.appendChild(video);

    imgElement = document.createElement('img');
    imgElement.style.position = 'absolute';
    imgElement.style.zIndex = 1;
    imgElement.onclick = () => { shutdown = true; };
    div.appendChild(imgElement);

    const instruction = document.createElement('div');
    instruction.innerHTML =
        '<span style="color: red; font-weight: bold;">' +
        'When finished, click here or on the video to stop this demo</span>';
    div.appendChild(instruction);
    instruction.onclick = () => { shutdown = true; };

    video.srcObject = stream;
    await video.play();

    captureCanvas = document.createElement('canvas');
```



```
    captureCanvas = document.createElement('canvas');
    captureCanvas.width = 640; //video.videoWidth;
    captureCanvas.height = 480; //video.videoHeight;
    window.requestAnimationFrame(onAnimationFrame);

    return stream;
  }
  async function stream_frame(label, imgData) {
    if (shutdown) {
      removeDom();
      shutdown = false;
      return '';
    }

    var preCreate = Date.now();
    stream = await createDom();

    var preShow = Date.now();
    if (label != "") {
      labelElement.innerHTML = label;
    }

    if (imgData != "") {
      var videoRect = video.getClientRects()[0];
      imgElement.style.top = videoRect.top + "px";
      imgElement.style.left = videoRect.left + "px";
      imgElement.style.width = videoRect.width + "px";
      imgElement.style.height = videoRect.height + "px";
      imgElement.src = imgData;
```



```
[15]      var preCapture = Date.now();
          var result = await new Promise(function(resolve, reject) {
            pendingResolve = resolve;
          });
          shutdown = false;

          return {'create': preShow - preCreate,
                  'show': preCapture - preShow,
                  'capture': Date.now() - preCapture,
                  'img': result};
        }
        ''')

    display(js)

  def video_frame(label, bbox):
    data = eval_js('stream_frame("{}", "{}")'.format(label, bbox))
    return data
```

```
import os
```

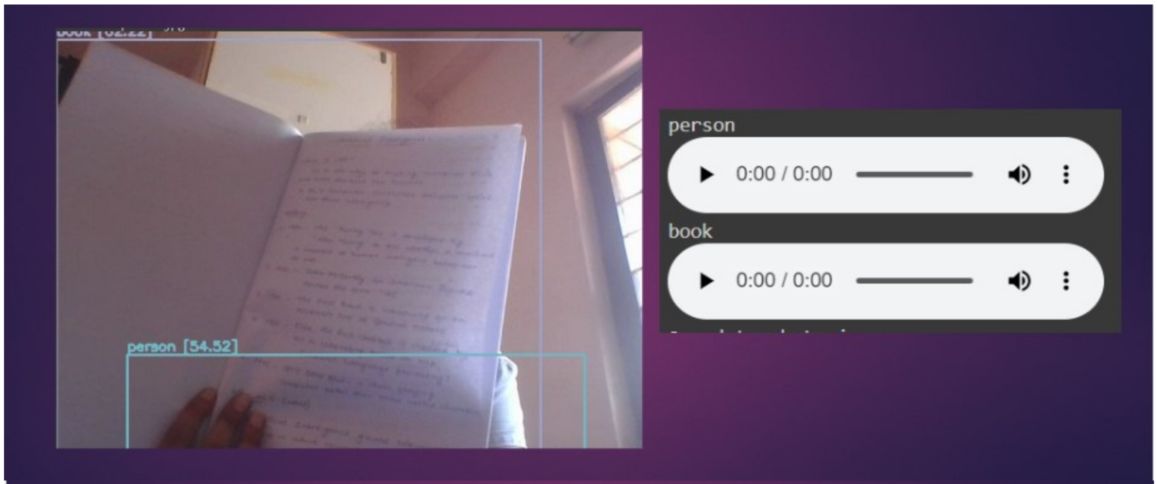
```
[17] # start streaming video from webcam
     video_stream()
     # label for video
     label_html = 'Capturing...'
     # initialze bounding box to empty
```

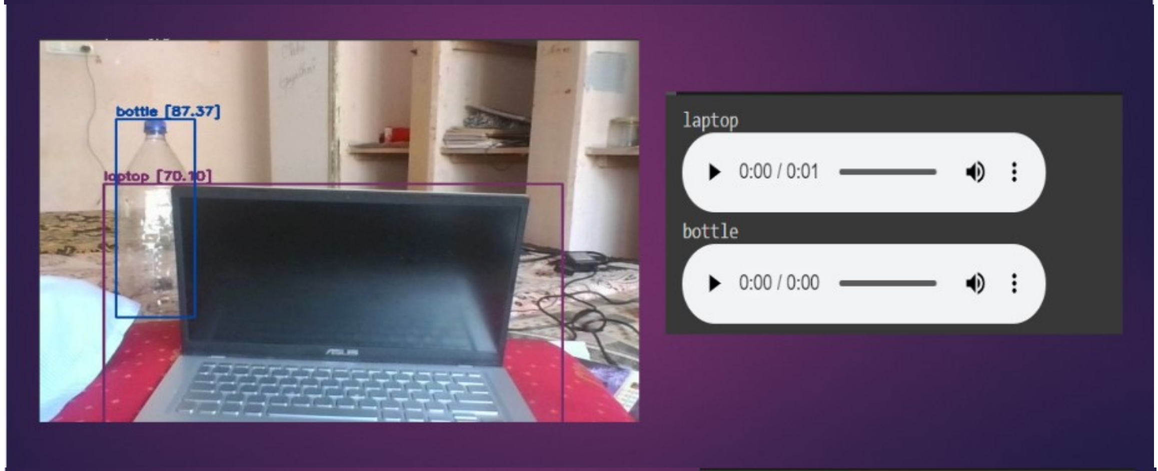
```
bbox = ''
count = 0
while True:
    js_reply = video_frame(label_html, bbox)
    if not js_reply:
        break

    # convert JS response to OpenCV Image
    frame = js_to_image(js_reply["img"])

    # create transparent overlay for bounding box
    bbox_array = np.zeros([480,640,4], dtype=np.uint8)

    # call our darknet helper on video frame
    detections, width_ratio, height_ratio = darknet_helper(frame, width, height)

    # loop through detections and draw them on transparent overlay image
    for label, confidence, bbox in detections:
      left, top, right, bottom = bbox2points(bbox)
      left, top, right, bottom = int(left * width_ratio), int(top * height_ratio), int(right * width_ratio), int(bottom * height_ratio)
      bbox_array = cv2.rectangle(bbox_array, (left, top), (right, bottom), class_colors[label], 2)
      bbox_array = cv2.putText(bbox_array, "{} [{:.2f}]".format(label, float(confidence)),
                        (left, top - 5), cv2.FONT_HERSHEY_SIMPLEX, 0.5,
                        class_colors[label], 2)
      print(label)


    bbox_array[:,:,3] = (bbox_array.max(axis = 2) > 0 ).astype(int) * 255
    # convert overlay of bbox into bytes
```

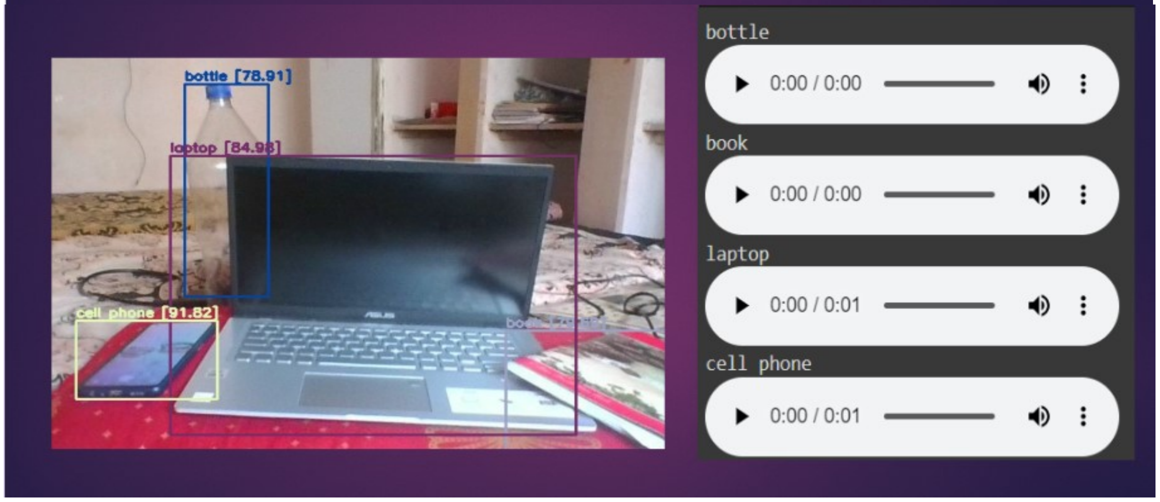
```
    bbox_array[:,:,3] = (bbox_array.max(axis = 2) > 0 ).astype(int) * 255
    # convert overlay of bbox into bytes
    bbox_bytes = bbox_to_bytes(bbox_array)
    # update bbox so next frame gets new overlay
    bbox = bbox_bytes
```

```
cell phone
person
cell phone
cell phone
person
person
person
person
```

9.Output


person [54.52]
person
0:00 / 0:00
book
0:00 / 0:00


bottle [87.37]
laptop [70.10]
laptop
0:00 / 0:01
bottle
0:00 / 0:00


bottle [78.91]
laptop [84.98]
cell phone [91.82]
bottle
0:00 / 0:00
book
0:00 / 0:00
laptop
0:00 / 0:01
cell phone
0:00 / 0:01

Results comparison with other approaches:

1. Faster R-CNN (Region Convolutional Neural Network):

Faster R-CNN provides high accuracy by using region proposal networks, making it suitable for precise object detection tasks. However, it might be slower compared to YOLO in real-time applications.

Drawbacks

- **Complex Architecture:** The intricate architecture of Faster R-CNN, including the region proposal network (RPN) and object detection network, can be daunting for beginners to comprehend, implement, and optimize.

- **Slower Inference**: Despite being faster than some previous methods, Faster R-CNN might still not meet real-time processing requirements for some applications, especially those needing ultra-low latency.

2. Haar Cascade Classifiers:

YOLO offers significantly higher accuracy and the ability to detect a broader range of object classes compared to Haar Cascade classifiers, making it a more robust and modern choice.

Drawbacks

- **False Positives/Negatives:** Achieving a balance between avoiding false positives and detecting all instances of an object can be challenging, impacting overall performance.

- **Performance on Small Objects:** Detecting small objects with Haar cascades might be challenging due to limitations in feature size and resolution.

10. Conclusion

The project of implementing object detection with voice feedback presents a dynamic and interactive solution that merges computer vision and audio synthesis technologies. This object detection with voice feedback project not only showcases the power of machine learning and computer vision but also highlights the potential for enhancing user experience through voice-based interactions.

For children, the system transforms learning into an interactive and engaging adventure, turning everyday objects into sources of discovery and knowledge. By hearing and seeing the objects around them, children can learn new words, explore their environment.By providing real-time descriptions of their surroundings, the system empowers blind individuals to navigate confidently and make informed decisions.This project serves as a dual-purpose solution: a playful learning companion for children and an empowering guide for the blind.

11.References

- https://github.com/theAIGuysCode/YOLOv4-Cloud-Tutorial
- https://ieeexplore.ieee.org/document/9377064
- https://www.v7labs.com/blog/coco-dataset-guide