Assignment 5- Video and real-time analysis

Part 1- Video read/write

We start with reading a video in python and applying some basic tools on it. The video chosen for this part is from *pixabay.com*, a resource of free video/image and sound clips.

a) Read the file "Bangkok.mp4" file (located in your assignment in Blackboard) using OpenCV. Find the relevant commands to find the video's metadata and print out following information about the video: frame per second (fps), total number of frames, height, and width of the video.
b) There are many ways to play this file inside Google Colab. If you just use OpenCV, it will be shown frame by frame. Another approach is to use an embedded version using the HTML library. For the rest of the steps, it would be easier to use the html version for playing the video.
c) Let's apply some filters on the video file. Use adaptive_theshold_mean and threshold_binary_inv, to create a sketch version of the video. You can use the Videowriter command to save the output, but it takes quite some time. Instead, only print 3 frames of the video.

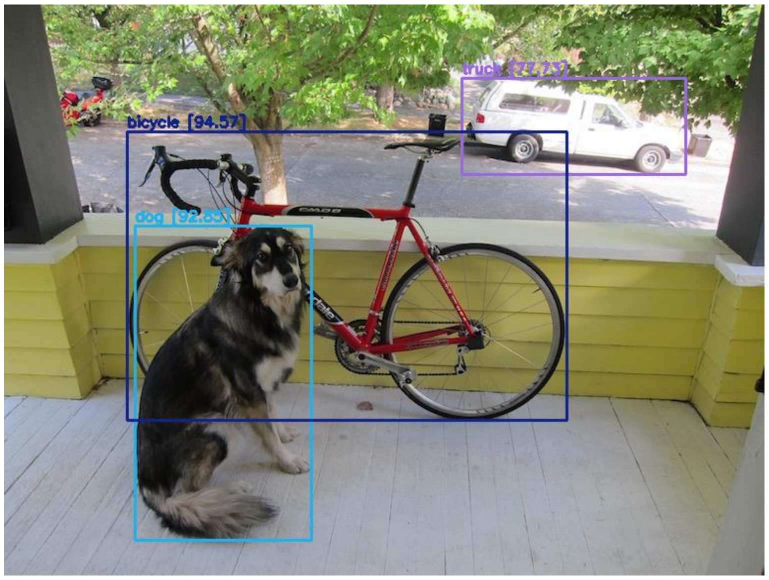
Part 2- Pose estimation in video

Return to the functions that were provided in previous assignments to estimate the pose in the images. This time, we want to experience it in a video file. For that, we will follow the approach in https://github.com/tensorflow/hub/tree/master/examples and estimate the pose of the girl in the video. The format of the video this time is gif.

Part 3- Using webcam video

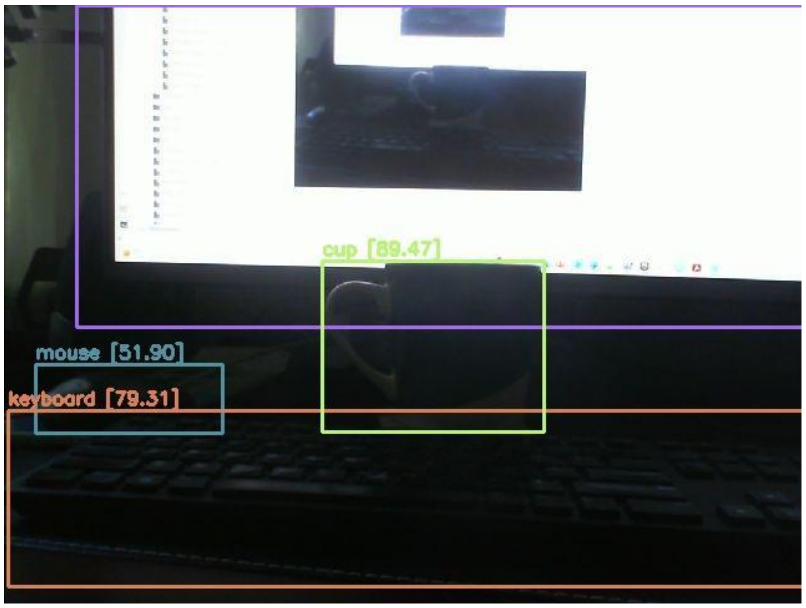
Now, we want to use a webcam and make object detection in real-time. As you have seen in previous modules, the first step is to have a pre-trained model and use the weights for future classifications. For this part, we will use the Darknet model.

a) Setup the model from the following GitHub: https://github.com/AlexeyAB/darknet. If you have any issues with setting up the model, feel free to reach out to your instructor. Use: *from darknet import ** to bring in all the required packages. You'll need to copy some functions from the repository too. For example, bring in the *darknet_helper* function. To test your model, you can read an image from the data folder and see how accurately your model can detect the objects. Here is an example dog.jpg:



b) Now, we want to connect our webcam to python to make a live object detection application. We will be using the code snippet for Camera Capture which runs JavaScript code to utilize your computer's webcam. The code snippet will take a webcam photo, which we will then pass into our YOLOv4 model for object detection. Please refer to the previous assignment to copy the code here.

Now, call the take_photo function, and see how accurately your model can detect the objects.



c) For the last part of the work, a real-time video processing is evaluated. Create a procedure to see the images through your camera and detect the objects in a real-time video.