LAB 3: Adding Functionality

Due: Tuesday, March 12th during lab session

The purpose of lab 3 is to further explore Cozmo's capabilities and establish a good code base for future labs. In this lab you will implement continuous driving, finite state machine behaviors, and AR marker reading on top of your lab 2 code.

Continuous driving: If you have not already, implement Cozmo motion using the `drive_wheels` command. This is necessary for the ability to drive and turn in place which is essential for the second part of the lab.
We will look for the use of this command in the code.

FSM: Adapt your code to include the capabilities of a finite state machine. There are many ways of doing this but it must be object oriented not a series of `if`, `elif` statements. We require at least three states to be demonstrated during the demo, use of the Cozmo screen to display the current state, an auditory signal for a state change, and a print statement on your terminal to show state changes. To summarize when your code runs, we need to be able to see the current state on the Cozmo screen, hear a beep when the current state is changed, and see on your terminal the exiting state and the entering state.
We look for the implementation of the FSM in the code. Also, you should add a diagram of your designed FSM (as a PDF file) along the code. The TA will grade this part based on both criteria.

Navigate towards the cube by reading AR markers: One cube will be placed in the Arena with your robot. Your robot must navigate to a pre-designated face of this cube and then face it. Note that in this step, you use a customized AR marker on the cube.
Before demoing and recoding, you should point at the face your Cozmo will go to and your TA will take this into account when grading. To generate a customized marker, use one of the following options:

- Pre-defined custom markers (recommended)
  https://data.bit-bots.de/cozmo_sdk_doc/cozmosdk.anki.com/docs/generated/cozmo.objects.html#cozmo.objects.CustomObjectMarkers

- ArUco Markers + OpenCV

  https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html

  https://pyimagesearch.com/2020/12/21/detecting-aruco-markers-with-opencv-and-python/

You will need two different markers for this lab.

Follow moving target: Before running the code, place the robot and the first cube in the arena. After your robot has reached the first cube, it should switch to a searching and tracking state for the second marker on the second cube (the markers must have different patterns). At this point place the cube in the arena (do not remove the first cube). While searching and tracking, you should move the second cube in front of the robot and Cozmo must follow it. If the cube leaves the robot's sight, it should begin its search pattern in the direction it was last seen.

Evaluation: The point breakdown for this lab is as follows:

The robot uses `drive_wheels` for its motion and is able to demonstrate this.	20 pts
The robot displays the current state on its face.	10 pts
The robot beeps to indicate a state change.	5 pts
Your terminal displays the exiting state and the entering state.	5 pts
Your code makes use of at least three distinct states (bring a drawing of your FSM).	10 pts
Your robot successfully navigates to the designated face of the cube.	25 pts
The robot tracks a moving cube and is able to move toward it.	15 pts
The robot begins its search pattern in the direction in which the cube exited it's view	10 pts

This lab does not make use of the color tracking algorithm that you developed during labs 1 and 2. So you should replace the color markers with the AR markers instead.

Your demo will be graded during the lab session.

Submission: Submit a zip file containing all code used for this assignment **in one python file and a PDF of your designed FSM** with the file name following the form `Group_xx.zip` where you substitute your group number for '`xx`'. Only one partner should submit to Blackboard.