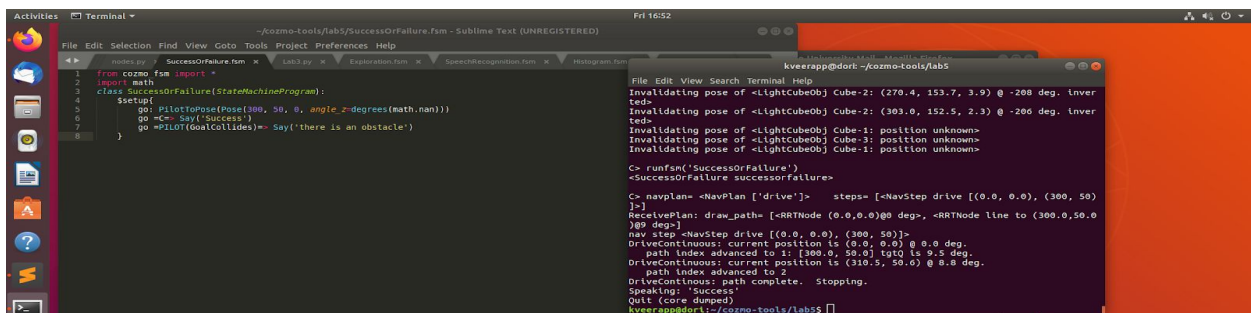


Name: Krithika Veerappan
Lab Partner: Hita Kambhamettu
Lab 5

Success or Failure

For this problem, we used the Pilot. Cozmo was able to tell when there was an object at his end goal such as the cubes, and raised a GoalCollides error because this meant he could not plan a path to get there. When he raised this error, we used Say to make cozmo say that there is an obstacle. On the other hand, if there was no obstacle and Cozmo was able to plan a path, he gets to the location and then says success, please see screenshots of both below.

Success Screenshot:

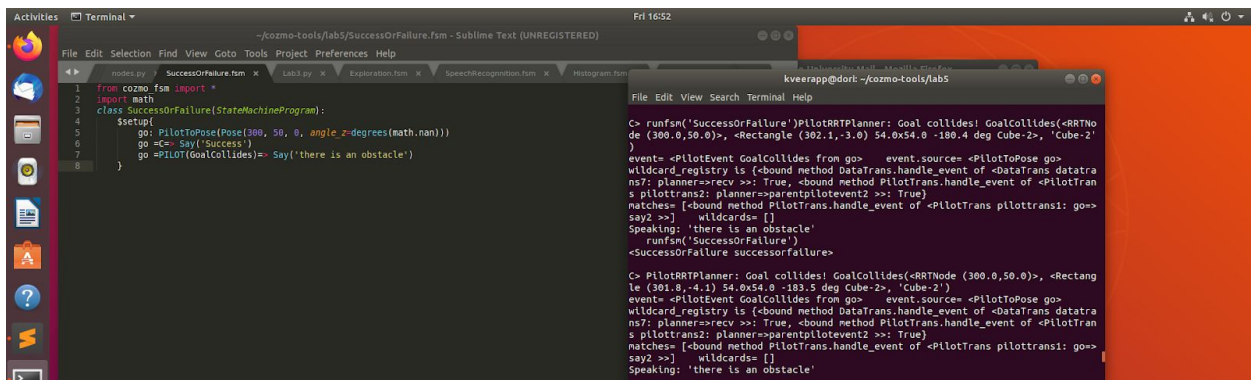


```
File Edit View Search Terminal Help
~cozmo-tools/lab5/SuccessOrFailure.fsm - Sublime Text (UNREGISTERED)
Fri 16:52

1 from cozmo_fsm import *
2 import math
3 class SuccessOrFailure(StateMachineProgram):
4     ssetup{
5         go: PilotToPose(Pose(300, 50, 0, angle_z=degrees(math.nan)))
6         go <=> Say('Success')
7         go =>Pilot(GoalCollides)> Say('there is an obstacle')
8     }

File Edit View Search Terminal Help
kveerapp@dort:~$ kcozmo-tools/lab5
File Edit View Search Terminal Help
Invalidating pose of <LightCubeObj Cube-2: (270.4, 153.7, 3.9) @ -200 deg. Inver
ted>
Invalidating pose of <LightCubeObj Cube-2: (303.0, 152.5, 2.3) @ -200 deg. Inver
ted>
Invalidating pose of <LightCubeObj Cube-1: position unknown>
Invalidating pose of <LightCubeObj Cube-1: position unknown>
Invalidating pose of <LightCubeObj Cube-1: position unknown>
C> runfsm('SuccessOrFailure')
<SuccessOrFailure successorfailure>
C> navPlan <NavPlan ['drive']> steps=<[<NavStep drive [(0.0, 0.0), (300, 50)
3>]
ReceivePlan: draw_path=<[<RRNode (0.0,0.0)@0 deg>, <RRNode line to (300.0,50.0
)@0 deg>]
nav step <NavStep drive [(0.0, 0.0), (300, 50)]>
DriveContinuous: current position is (0.0, 0.0) @ 0.0 deg.
path index advanced to 1: (300.0, 50.0) yaw is 0.5 deg.
DriveContinuous: current position is (310.5, 50.0) @ 0.0 deg.
path index advanced to 2
DriveContinuous: path complete. Stopping.
Speaking: 'Success'
Quit (core dumped)
kveerapp@dort:~$ kcozmo-tools/lab5
```

Failure screenshot:



```
File Edit View Search Terminal Help
~cozmo-tools/lab5/SuccessOrFailure.fsm - Sublime Text (UNREGISTERED)
Fri 16:52

1 from cozmo_fsm import *
2 import math
3 class SuccessOrFailure(StateMachineProgram):
4     ssetup{
5         go: PilotToPose(Pose(300, 50, 0, angle_z=degrees(math.nan)))
6         go <=> Say('Success')
7         go =>Pilot(GoalCollides)> Say('there is an obstacle')
8     }

File Edit View Search Terminal Help
kveerapp@dort:~$ kcozmo-tools/lab5
File Edit View Search Terminal Help
C> runfsm('SuccessOrFailure')PilotRRTPPlanner: Goal collides! GoalCollides(<RRTN
ode (300.0,50.0)>,<Rectangle (302.1,-3.0) 54.0x54.0 -180.4 deg Cube-2>,<'Cube-2'
>)
event=<PilotEvent GoalCollides from go> event.source=<PilotToPose go>
wildcard_registry is <bound method DataTrans.handle_event of <DataTrans datatra
ns?> planner>recv>> True, <bound method PilotTrans.handle_event of <PilotTran
s pilottrans2: planner>parentPilotEvent2>> True)
matches=<[bound method PilotTrans.handle_event of <PilotTrans pilottrans1: go>
say2>>] wildcard=<[]>
Speaking: 'there is an obstacle'
runfsm('SuccessOrFailure')
<SuccessOrFailure successorfailure>
C> PilotRRTPPlanner: Goal collides! GoalCollides(<RRNode (300.0,50.0)>,<Rectang
le (301.8,-4.3) 54.0x54.0 -182.5 deg cube-2>,<'Cube-2'>)
event=<PilotEvent GoalCollides from go> event.source=<PilotToPose go>
wildcard_registry is <bound method DataTrans.handle_event of <DataTrans datatra
ns?> planner>recv>> True, <bound method PilotTrans.handle_event of <PilotTran
s pilottrans2: planner>parentPilotEvent2>> True)
matches=<[bound method PilotTrans.handle_event of <PilotTrans pilottrans1: go>
say2>>] wildcard=<[]>
Speaking: 'there is an obstacle'
```

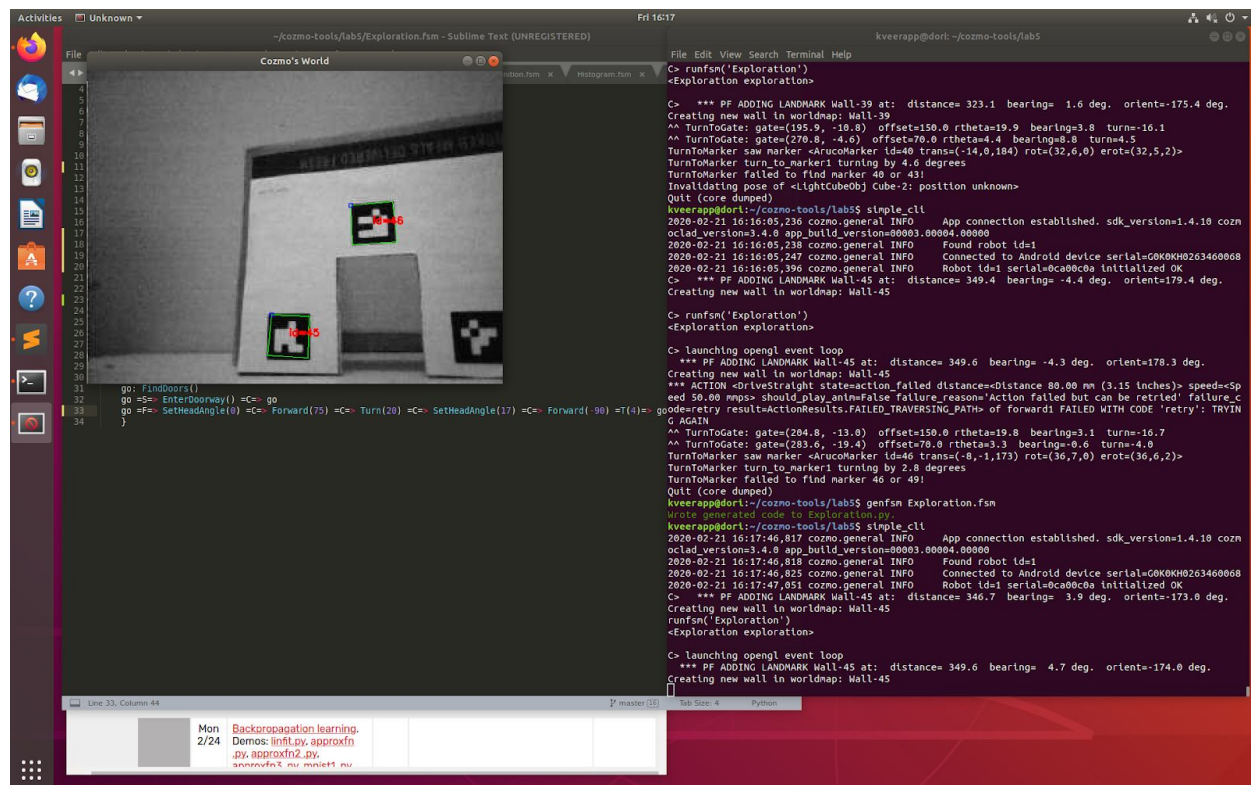
Exploration

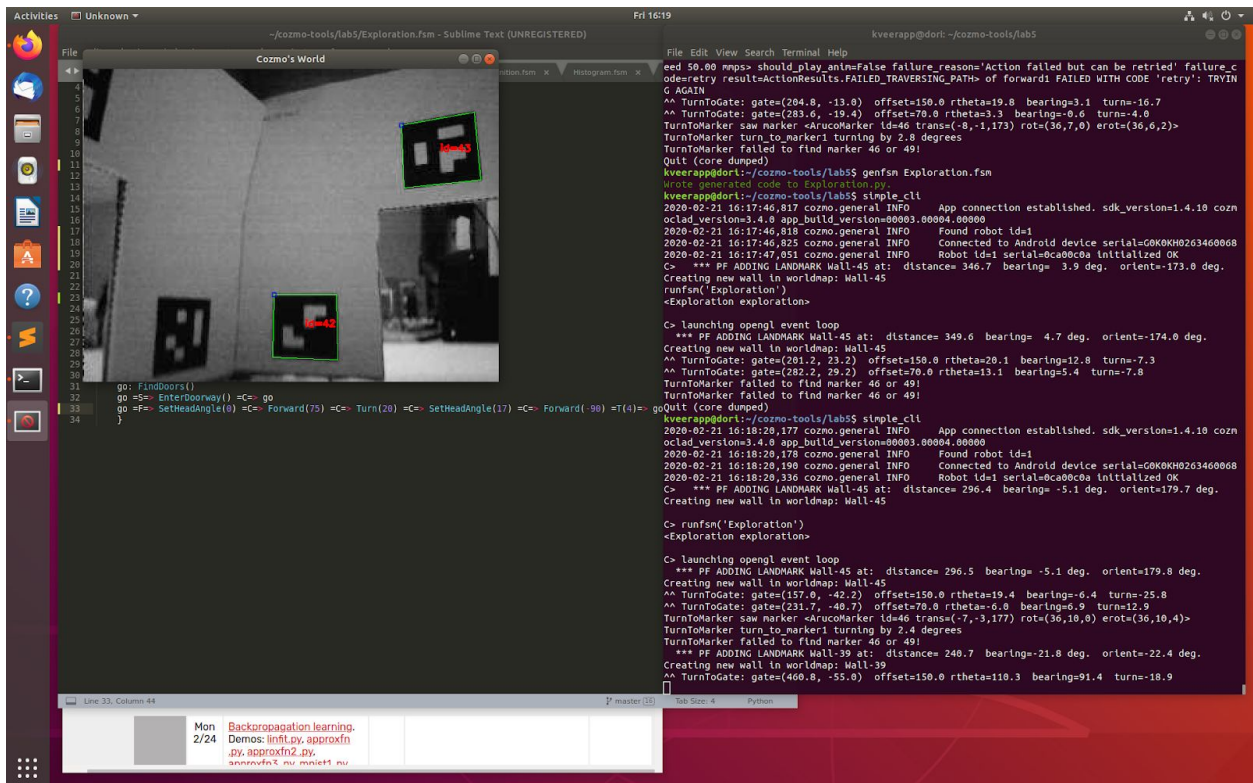
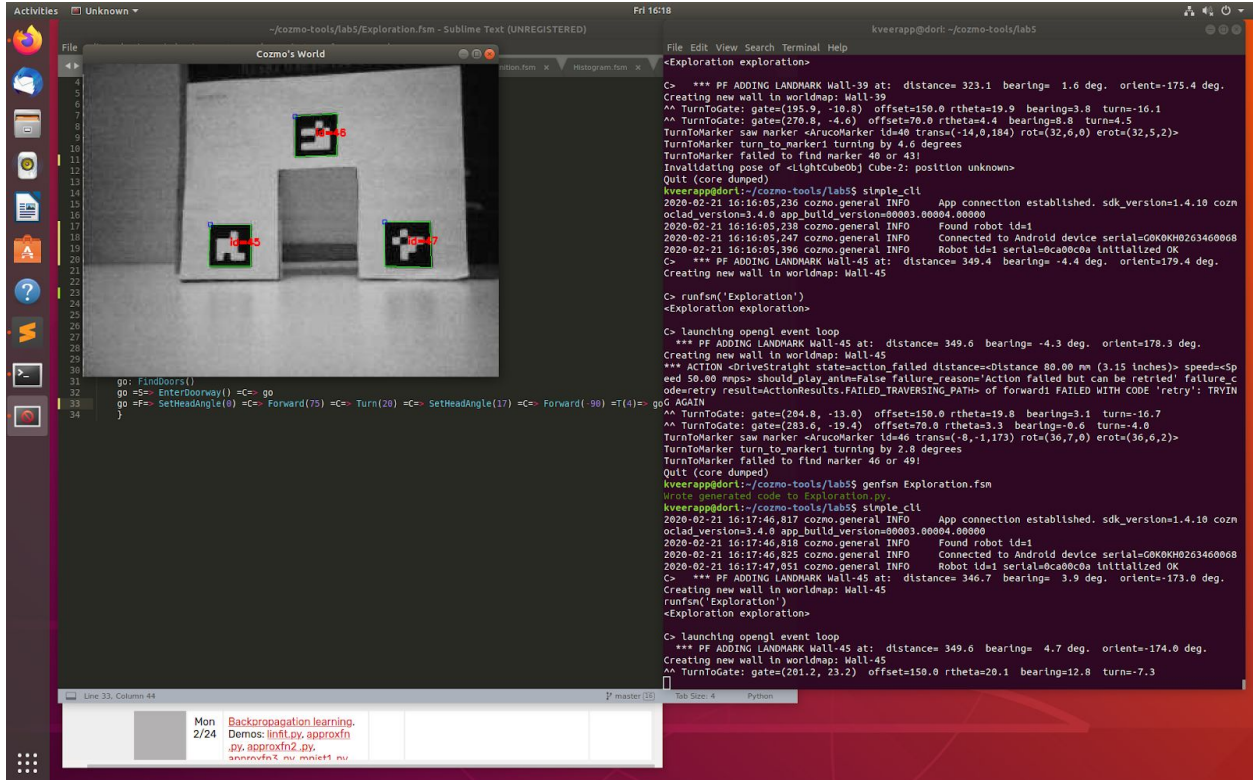
For this problem, we have an fsm that calls on a FindDoors state node, an EnterDoorway node which inherits from DoorPass, and some forward/turn directions. The fsm first calls on the FindDoors node, which goes through the list of objects in the world map viewer and searches for the doorways with the key, value in that dictionary. It adds the doorways to a global listOfDoors if the doorway is not already in the list. If a new door is added, the FindDoors node

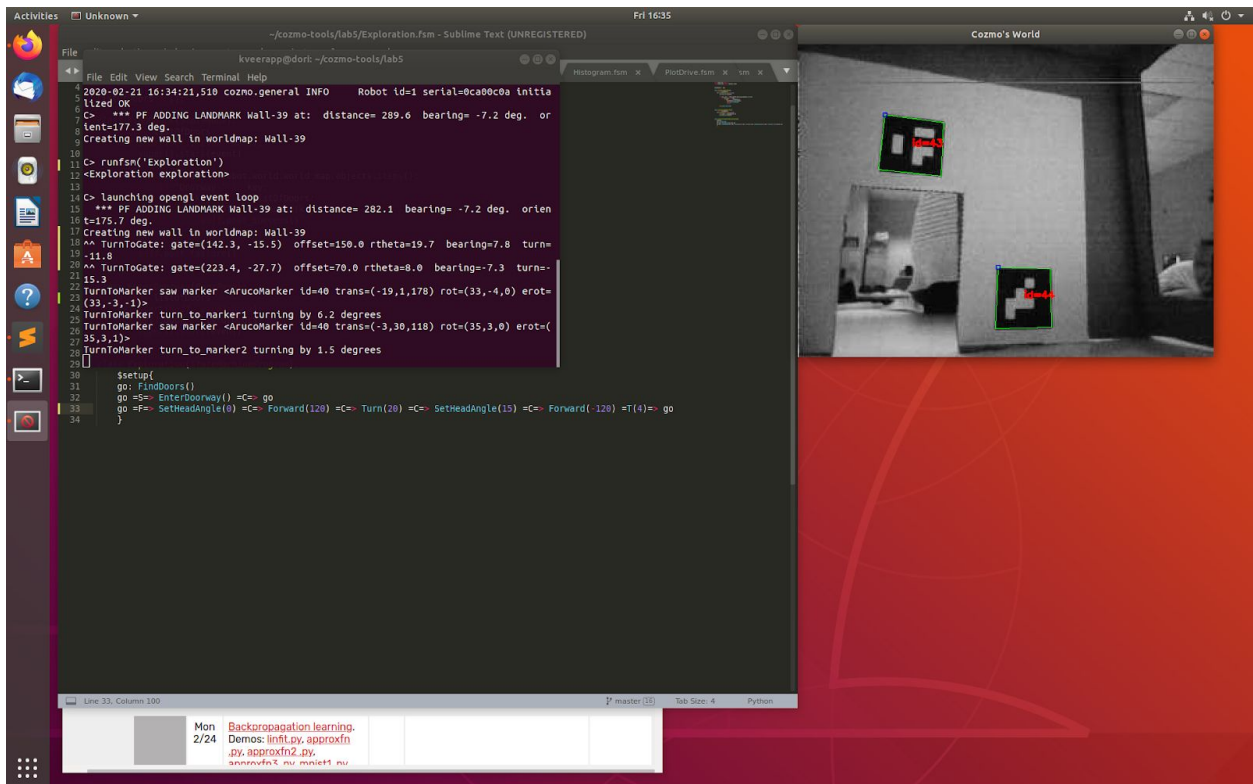
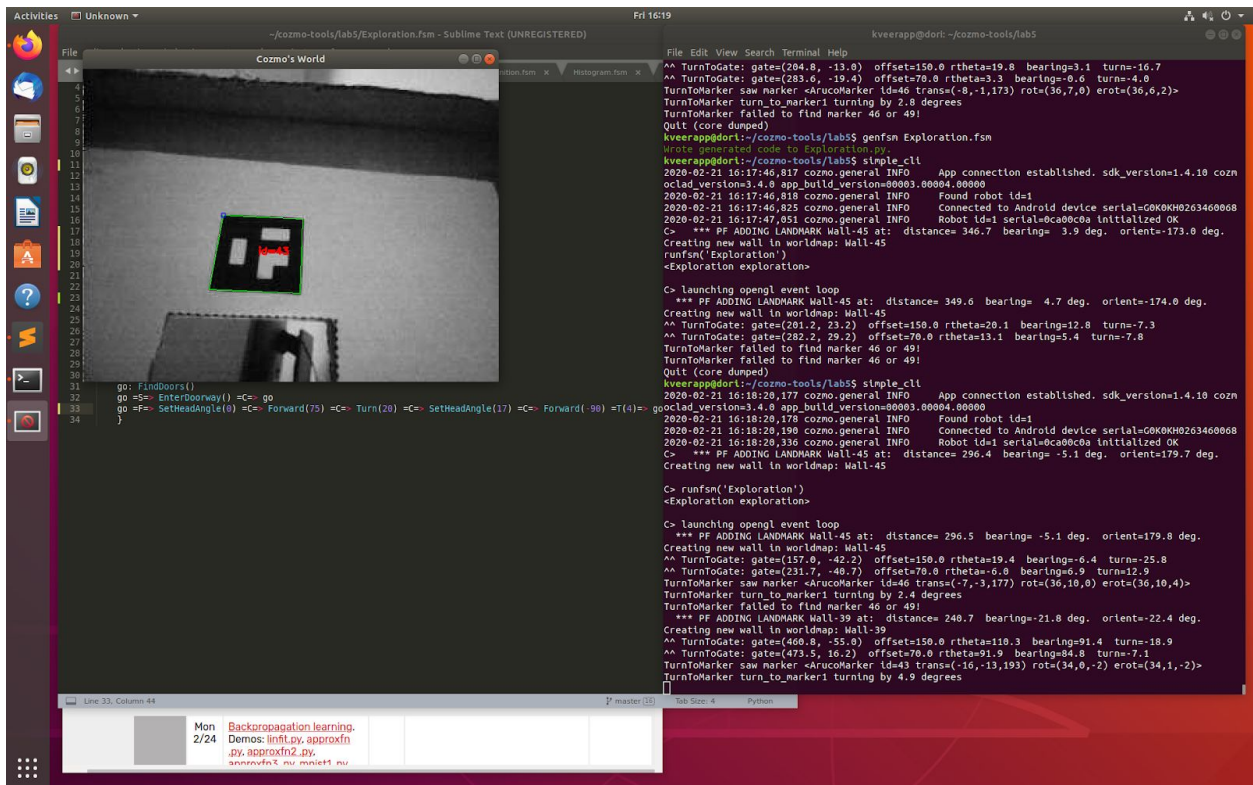
posts a success and then in the fsm this triggers the EnterDoorway node. The EnterDoorway node sets the door equal to the last door added in the listOfDoors and passes through this new door. On the other hand, if FindDoors can not find a new doorway to add, it posts a failure and so cozmo will keep looping through the forward and turn directions until he finds a new door (this way he will not reenter previously visited doors). The edge cases we tested:

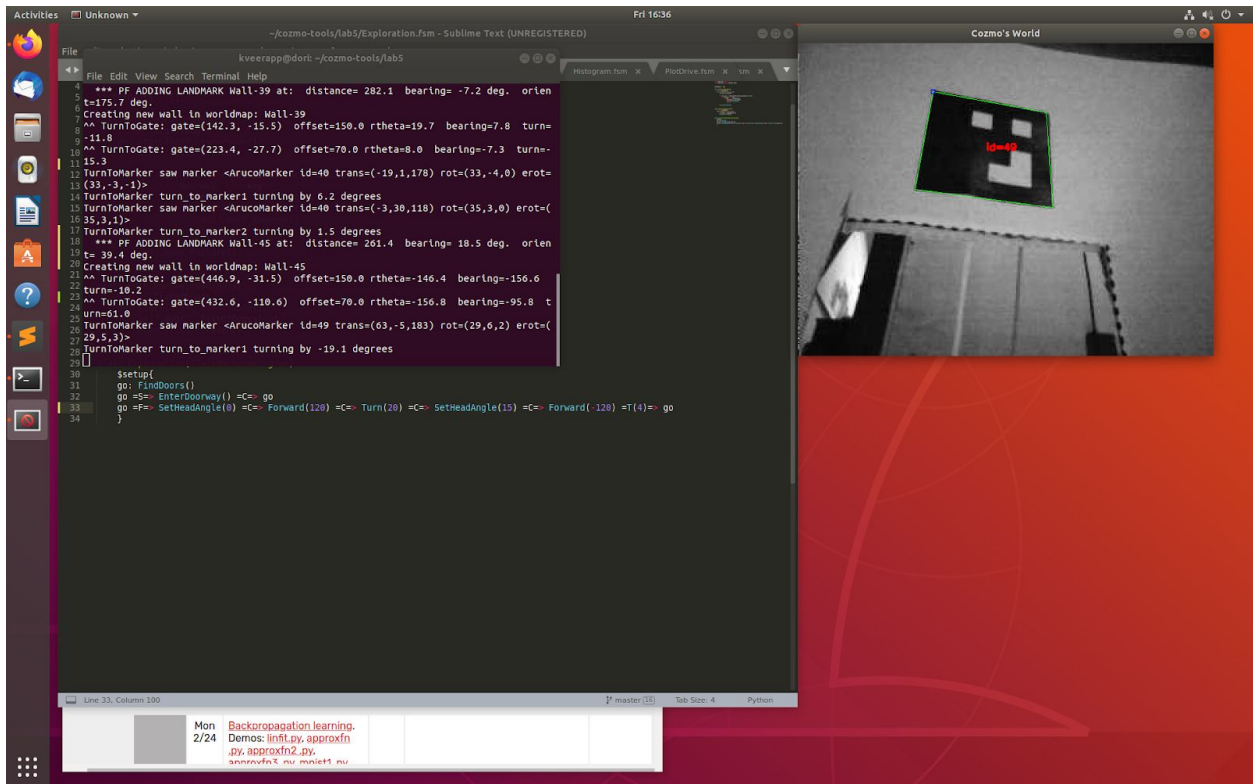
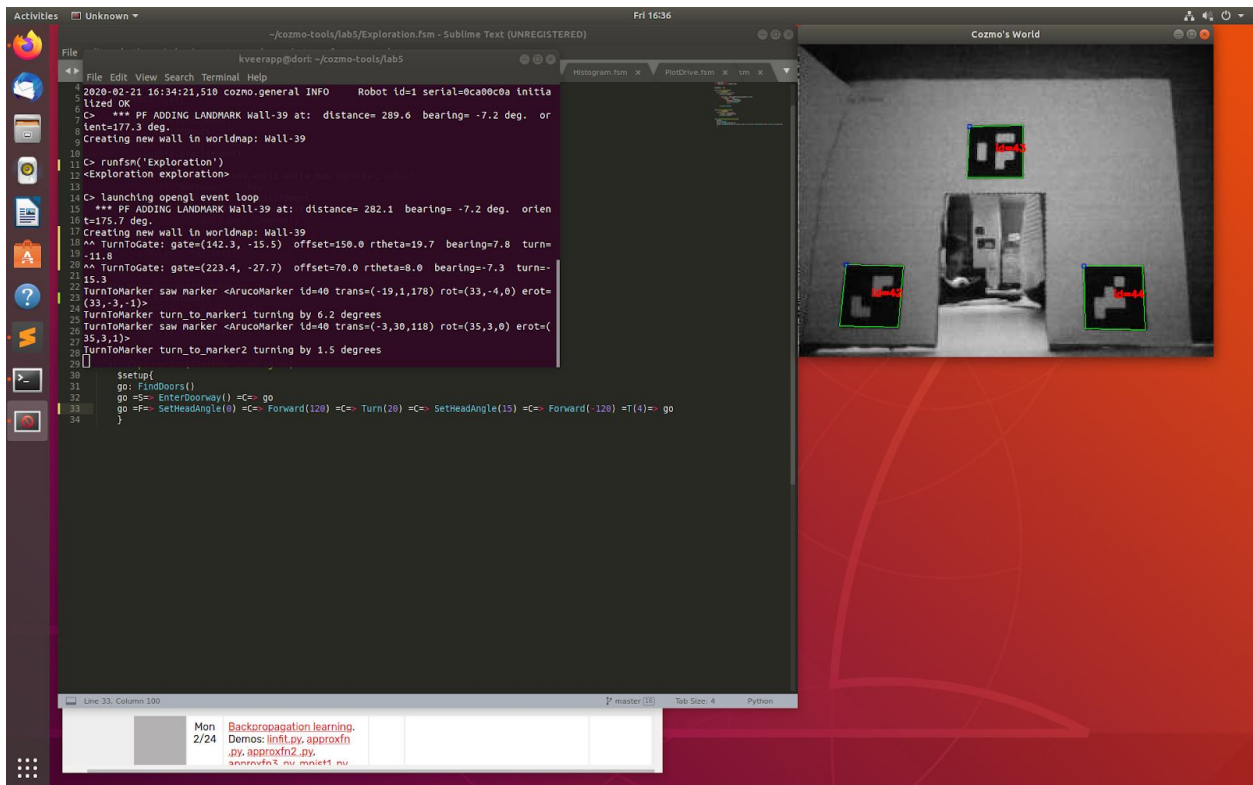
- Cozmo does not enter the same door twice
- Cozmo is able to enter the second door regardless of which door he enters first (from the long wall or the short wall).
- Cozmo is eventually able to find his way through the second door regardless of which perpendicular side it is on.

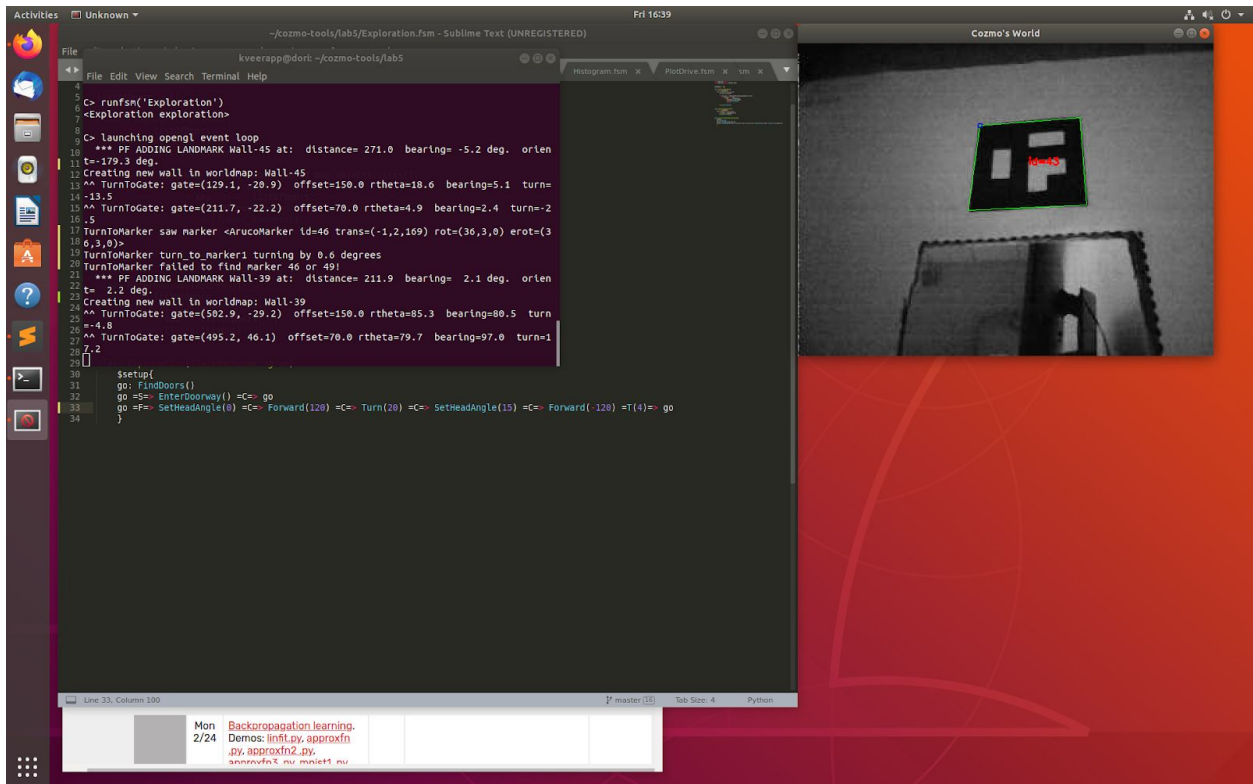
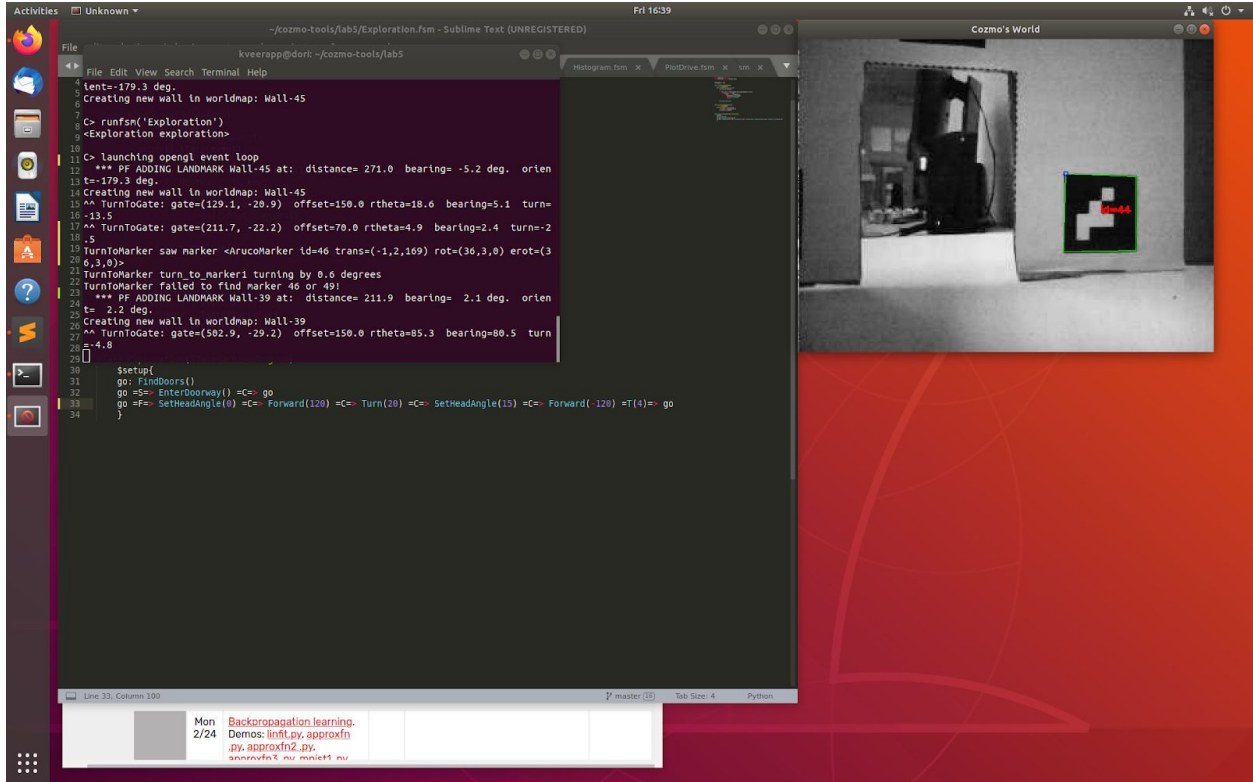
Please find relevant screenshots as well as pictures below.











Pictures of Cozmo passing through doors in the different possible orientations of the box (whether it starts with long edge, and then passes through short edge or whether it starts with the short, and then passes through the long):



