**Virtual Pi2Go Programming: Exercises with Machine Learning**

**Exercise 1:** Write a Machine Learning algorithm to get the virtual Pi2Go to avoid obstacles. It should choose between the actions forward, right and left, and use the input from the three infra-red obstacle sensors as its state.

**Exercise 2:** Write a Machine Learning algorithm to get the virtual Pi2Go to follow a wall by keeping the wall always to one side of it.

**Exercise 3**: Write a Machine Learning algorithm to get the virtual Pi2Go to turn towards a bright light.

**Hints:** You may want to use the following function to calculate the state. This returns a tuple where the first number determines whether it is brighter on the front left or the front right, the second whether it is brighter on the back left or back right and the third number whether it is brighter at the front or the back.

def get\_state():

fl = pi2go.getLightFL()

fr = pi2go.getLightFR()

bl = pi2go.getLightBL()

br = pi2go.getLightBR()

comp\_f = 0

if (fl > fr):

comp\_f = 1

elif (fr > fl):

comp\_f = -1

comp\_b = 0

if (bl > br):

comp\_b = 1

elif (br > bl):

comp\_b = -1

comp = 0

if (fl + fr > bl + br):

comp = 1

elif (bl + br > fl + fr):

comp = -1

return (comp\_f, comp\_b, comp)

You may also want to use the following function to calculate the reward rather than using a dictionary. This function increases the reward as the values of the two front light sensors get closer to each other:

def calculate\_reward():

fl = pi2go.getLightFL()

fr = pi2go.getLightFR()

bl = pi2go.getLightBL()

br = pi2go.getLightBR()

if (bl + br > fl + fr):

return 0

f\_diff = fl - fr

f\_diff\_mod = f\_diff/1023

if (f\_diff\_mod > 0):

return (1 - f\_diff\_mod)

else:

return (1 + f\_diff\_mod)



University of Liverpool, 2020

This work is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-nc-sa/4.0/).