**How I was going to do Part3 of Task3**

1->in a new terminal run roscore

2->in a new terminal run turtlesim turtlesim\_node

This will open the turtlesim window with a turtle name turtle1

3->in a new terminal run rosservice call /spawn 8.3166675 5.544445 0.0 “Turtle2”

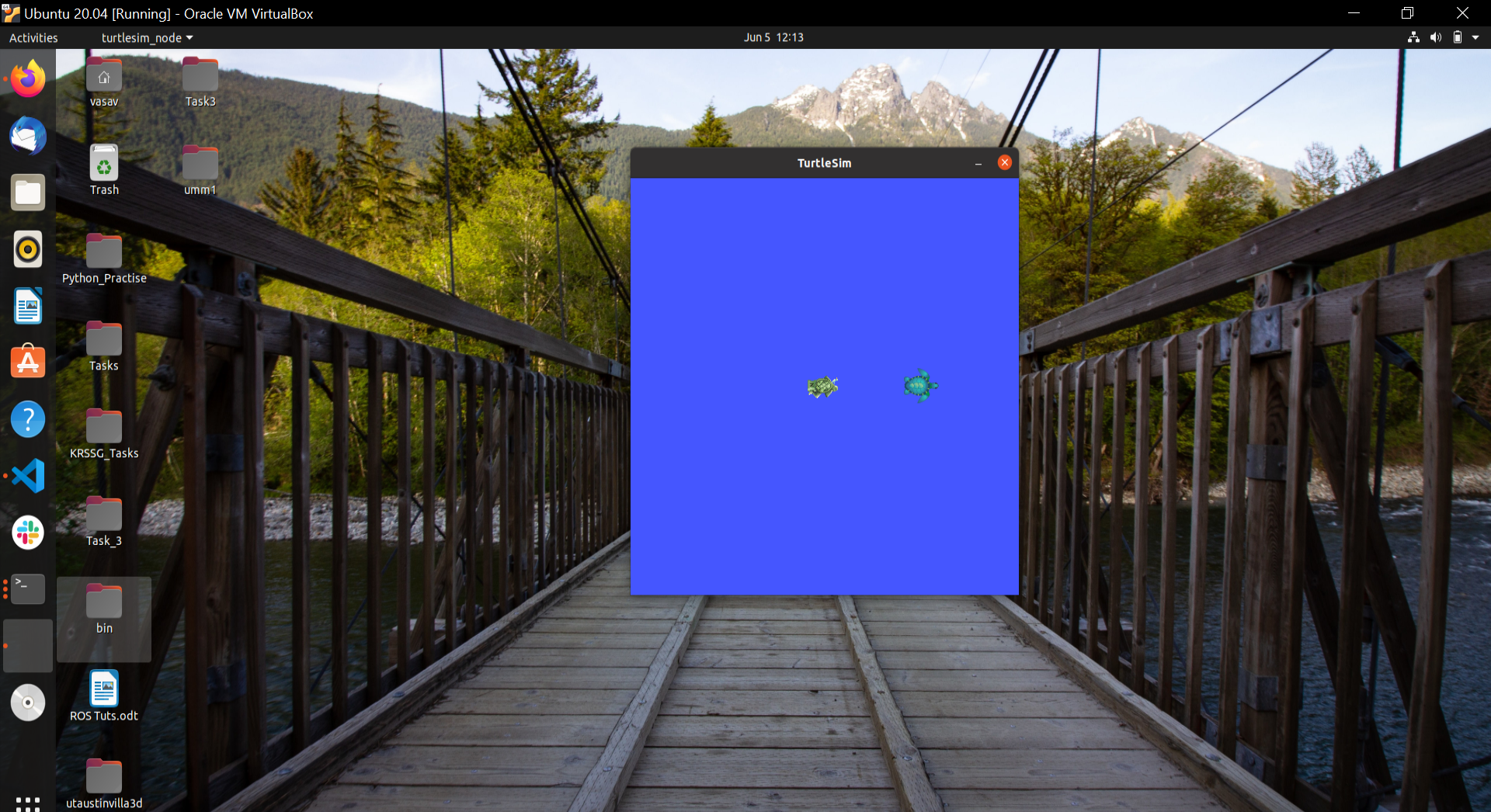
This will spawn another turtle in the earlier turtlesim window at (x = 8.3166675 )

Which is ¾ of the turtlesim window width and (y = 5.544445) which is ½

turtlesim window height and making angle zero degree with the horizontal (facing

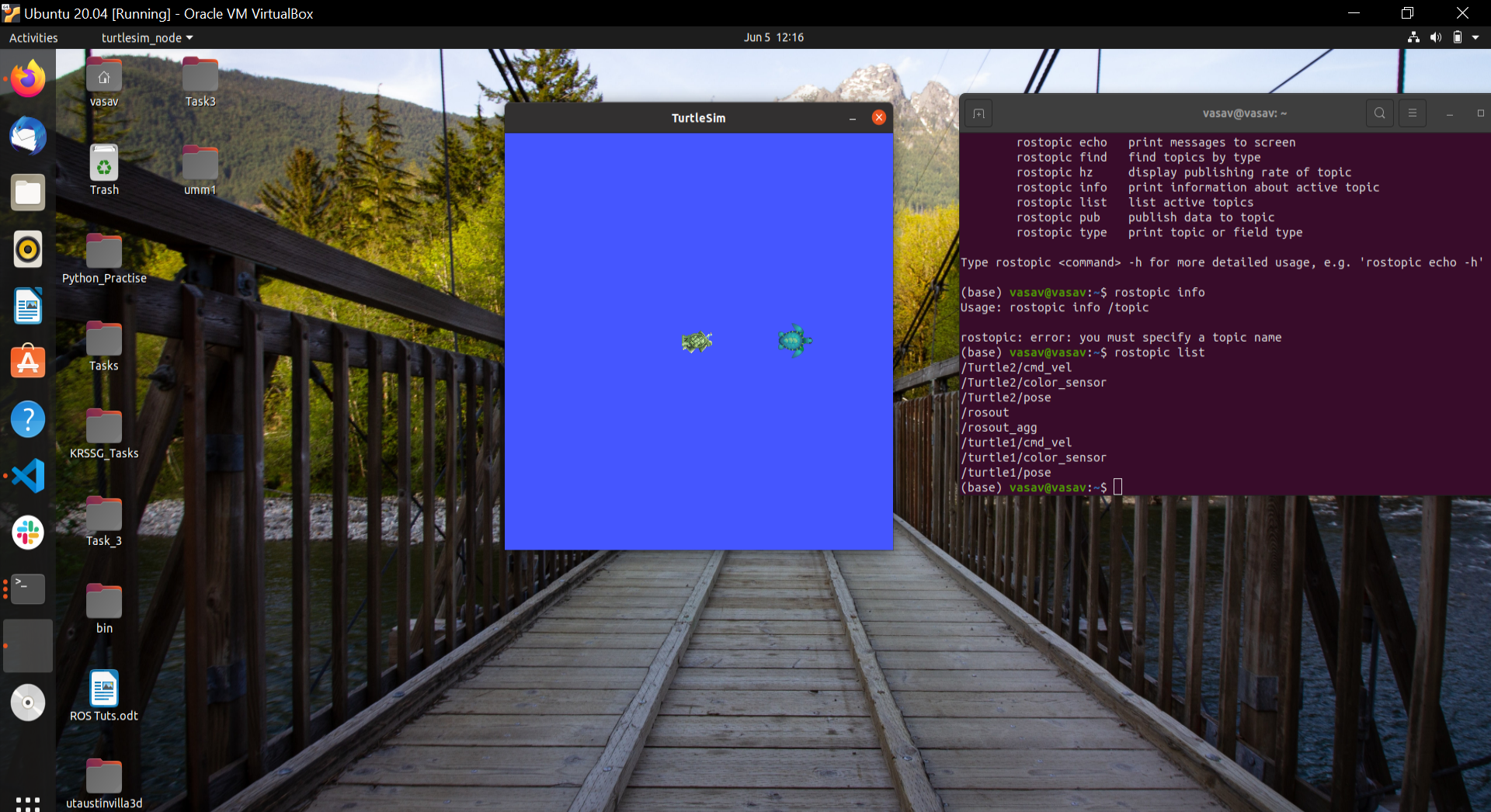
right) with the name Turtle2

Here is the image for how the configuration will look



4->now we can run rostopic list to find about the list of active topics

Which will show the output like this



So

/Turtle2/cmd\_vel

/Turtle2/color\_sensor

/Turtle2/pose

/rosout

/rosout\_agg

/turtle1/cmd\_vel

/turtle1/color\_sensor

/turtle1/pose

This is the output from here we can know that Turtle2 will publish its velocity to the topic /Turtle2/cmd\_vel and position to /Turtle2/pose

Now we can write a python file for it where we publish data to /Turtle2/pose to move it in ac circle of radius ¼ of the turtlesim window

And then we can subscribe to this channel to know the information of position of Turtle2

Then using the same idea

* I can make changes to my file gotogoal.py (which receives the coordinates of its path from rrt\_star\_connect.py and then moves on it using a PID controller) to subscribe to the /Turtle2/pose to know about the position of Turtle2
* then calculate the distance between my turtle and the Turtle2 and if that distance is less than 1.5(in the units of turtlesim window) then I will replan the path of my turtle
* I was planning to implement rrt star to in my gotogoal.py which will give me new path fast and a good path in which I was make check obstacle function which will consider the present position of Turtle2 and some points around it as obstacle
* also I was thinking of importing coordinates of all the obstacle points in image and then scaling them to turtlesim window dimensions and treating them also as an obstacle
* then finding the path and moving my turtle on it
* This meant that my obstacle function will make a list in which it will store coordinates of all the obstacle
* So continuously new points will be added and removed from it
* And path on which my turtle moves will also change every few seconds

**x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x**