ROS LAB - 6
Line_follow

```
import numpy
import rclpy
from rclpy.node import Node
from cv_bridge import CvBridge
from sensor_msgs.msg import Image
from geometry_msgs.msg import Twist
import cv2

class LineFollower(Node):
        def __init__(self):
        super().__init__('line_follower')
        self.bridge = CvBridge()
        self.subscriber =
self.create_subscription(Image,'/camera1/image_raw',self.process_data, 10)
        self.publisher = self.create_publisher(Twist, '/cmd_vel', 40)
        timer_period = 0.2
        self.timer = self.create_timer(timer_period, self.send_cmd_vel)
        self.velocity=Twist()
        self.empty = False
        self.error = 0
        self.action=""
        self.get_logger().info("Node Started!")

        def send_cmd_vel(self):
        if(self.empty):
        self.velocity.linear.x=0.2
        self.velocity.angular.z= 0.2
        self.action="Stop"
        else:
        if(self.error > 0):
                self.velocity.linear.x=0.1
                self.velocity.angular.z=0.1
                self.action="Go Left"
        elif(self.error < 0):
                self.velocity.linear.x=0.1
                self.velocity.angular.z=-0.1
                self.action="Go Right"
        elif(self.error==0):
                self.velocity.linear.x=0.1
                self.velocity.angular.z= 0.0
                self.action="Go Straight"
self.publisher.publish(self.velocity)

## Subscriber Call Back
def process_data(self, data):
self.get_logger().info("Image Received!")
frame = self.bridge.imgmsg_to_cv2(data)
light_line = numpy.array([120,120,0])
dark_line = numpy.array([150,150,10])
mask = cv2.inRange(frame, light_line,dark_line)
cv2.imshow('mask', mask)

canny= cv2.Canny(mask,30,5)
cv2.imshow('edge', canny)

r1=200;c1=0
img = canny[r1:r1+200,c1:c1+512]
cv2.imshow('crop', img)

edge=[]
row =150

for i in range(512):
if(img[row,i]==255):
        edge.append(i)
print(edge)

if(len(edge)==0):
left_edge=512//2
right_edge=512//2
self.empty = True
if(len(edge)==1):
if edge[0]>512//2:
        left_edge=0
        right_edge=edge[0]
        self.empty = False
else:
        left_edge=edge[0]
        right_edge=512
        self.empty = False
if(len(edge)==2):
left_edge=edge[0]
right_edge=edge[1]
self.empty = False
```

```
        if(len(edge)==3):
        if(edge[1]-edge[0]>5):
                left_edge=edge[0]
                right_edge=edge[1]
                self.empty = False
        else:
                left_edge=edge[0]
                right_edge=edge[2]
                self.empty = False
        if(len(edge)==4):
        left_edge=edge[0]
        right_edge=edge[2]
        self.empty = False

        if(len(edge)>=5):
        left_edge=edge[0]
        right_edge=edge[len(edge)-1]
        self.empty = False

        road_width=(right_edge-left_edge)
        frame_mid = left_edge + (road_width/2)
        mid_point = 512/2
        img[row,int(mid_point)]=255
        print(mid_point)
        self.error=mid_point-frame_mid
        img[row,int(frame_mid)]=255
        print(self.action)
        f_image = cv2.putText(img, self.action, (100,100), cv2.FONT_HERSHEY_SIMPLEX, 1,
(255,0,), 2, cv2.LINE_AA)

def main(args=None):
        rclpy.init(args=args)
        node = LineFollower()
        rclpy.spin(node)
        rclpy.shutdown()

if __name__ == '__main__':
        main()
```

Extract_Road

```
import cv2
import numpy

image = cv2.imread('/home/aryavarta/shot.png')

def mouse(event, x, y, flags, param):
        if event == cv2.EVENT_LBUTTONDOWN:
        h = image[y, x, 0]
        s = image[y, x, 1]
        v = image[y, x, 2]
        print("H:", h)
        print("S:", s)
        print("V:", v)

cv2.namedWindow('mouse')
cv2.setMouseCallback('mouse', mouse)
cv2.imshow("original image", image)
cv2.imshow("mouse", image)
cv2.waitKey(0)
cv2.destroyAllWindows()

light_line = numpy.array([120,0,0])
dark_line = numpy.array([150,10,10])
mask = cv2.inRange(image, light_line, dark_line)
cv2.imshow('mask', mask)
cv2.waitKey(0)
cv2.destroyAllWindows()

canny = cv2.Canny(mask, 30, 5)
cv2.imshow('edge', canny)
cv2.waitKey(0)
cv2.destroyAllWindows()

print(canny.shape)
r1 = 200
c1 = 0
img = canny[r1:r1+200, c1:c1+512]
cv2.imshow('crop', img)
cv2.waitKey(0)
cv2.destroyAllWindows()

edge = []
```

```
row = 150
for i in range(512):
        if img[row, i] == 255:
        edge.append(i)

print(edge)

if len(edge) == 4:
        left_edge = edge[0]
        right_edge = edge[2]
        print(edge)

if len(edge) == 3:
        if edge[1] - edge[0] > 5:
        left_edge = edge[0]
        right_edge = edge[1]
        else:
        left_edge = edge[0]
        right_edge = edge[2]

road_width = (right_edge - left_edge)
frame_mid = left_edge + (road_width / 2)
mid_point = 512 / 2
img[row, int(mid_point)] = 255
print(mid_point)

error = mid_point - frame_mid
if error < 0:
        action = "Go Right"
else:
        action = "Go Left"

print("error", error)
img[row, int(frame_mid)] = 255
print("mid point of the frame", frame_mid)

f_image = cv2.putText(img, action, (50, 50), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0), 1,
cv2.LINE_AA)
cv2.imshow('final image', f_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

RVIZ Launch file

```
import os
from ament_index_python.packages import get_package_share_directory

from launch import LaunchDescription
from launch_ros.actions import Node

def generate_launch_description():
        package_dir='/home/aryavarta/ros2_lab/src/lab6/urdf'
        urdf = os.path.join(package_dir,'car.urdf')
        # rviz_config_file=os.path.join(package_dir,'config.rviz')
        return LaunchDescription([
        Node(
        package='robot_state_publisher',
        executable='robot_state_publisher',
        name='robot_state_publisher',
        output='screen',
        arguments=[urdf]),
        Node(
        package='joint_state_publisher_gui',
        executable='joint_state_publisher_gui',
        name='joint_state_publisher_gui',
        arguments=[urdf]),
        Node(
        package='rviz2',
        executable='rviz2',
        name='rviz2',
        # arguments=['-d',rviz_config_file],
        output='screen'),
        ])
```