**Nama : Ardhien Fadhillah Suhartono**

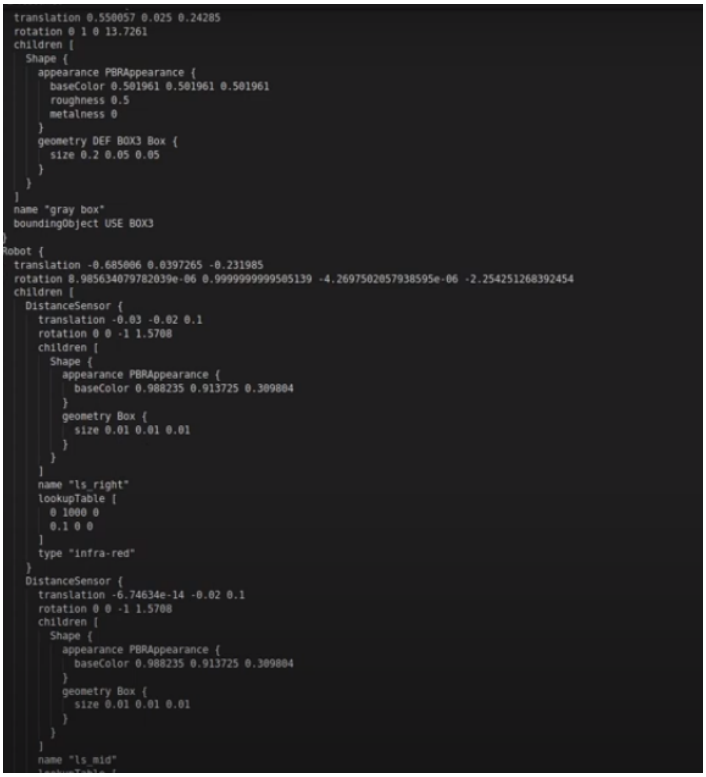
**NIM : 1103204137**

**Kelas : TK4402**

**Lecture 5**

**Integrasi ROS2 dengan Webots Tutorial 6**

Kita disini akan menccoba melakukan simulasi robot line follower pada webots dengan melakukan penginstallan dengan nama line_following_launch.py. Disini kita diberitahu kalu kitab isa mengedit dan melihat letak sensor kita. Lalu pada file robot.wtb kita dapat melihat berbagai sensor yang digunakan didalam robot kalian.



```
translation 0.550057 0.025 0.24285
rotation 0 1 0 13.7261
children [
  Shape {
    appearance PBRAppearance {
      baseColor 0.501961 0.501961 0.501961
      roughness 0.5
      metalness 0
    }
    geometry DEF BOX3 Box {
      size 0.2 0.05 0.05
    }
  }
]
name "gray box"
boundingObject USE BOX3
}
Robot {
translation -0.685006 0.0397265 -0.231985
rotation 8.985634079782039e-06 0.9999999999505139 -4.269750205793859 5e-06 -2.254251268392454
children [
  DistanceSensor {
    translation -0.03 -0.02 0.1
    rotation 0 0 -1 1.5708
    children [
      Shape {
        appearance PBRAppearance {
          baseColor 0.988235 0.913725 0.309804
        }
        geometry Box {
          size 0.01 0.01 0.01
        }
      }
    ]
    name "ls_right"
    lookupTable [
      0 1000 0
      0.1 0 0
    ]
    type "infra-red"
  }
  DistanceSensor {
    translation -6.74634e-14 -0.02 0.1
    rotation 0 0 -1 1.5708
    children [
      Shape {
        appearance PBRAppearance {
          baseColor 0.988235 0.913725 0.309804
        }
        geometry Box {
          size 0.01 0.01 0.01
        }
      }
    ]
    name "ls_mid"
```

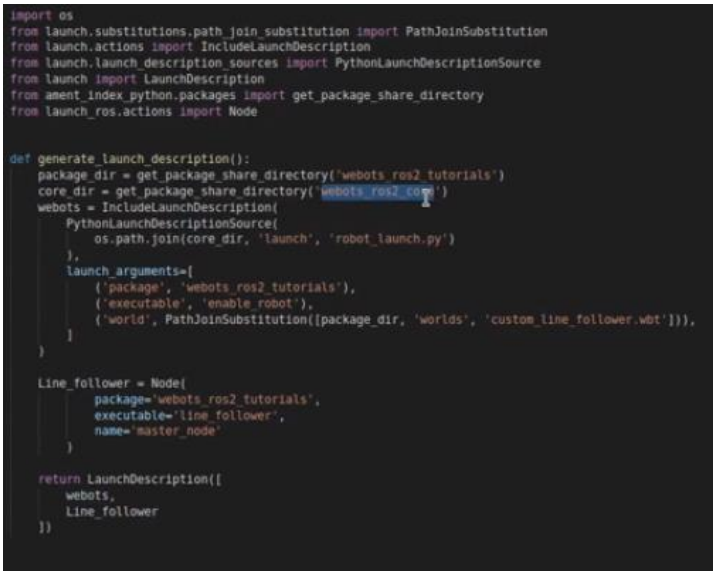
Selajutnya kita mempunyai node yang dinamakan slave.py yang dimana berfungsi untuk berinteraksi dengan robot kita. Dimana disini kitab isa mengatur bagaian bagian dari robot kita seperti badan roda ataupun sensor, dimana kita juga dapat mengatur jarak sensor sesuka kita

```
        self.get_logger().info('Sensor enabled')

        # Wheels section
        # |1 2|
        # |3 4|
        # Front wheels
        self.leftMotor_front = self.robot.getMotor('wheel1')
        self.leftMotor_front.setPosition(float('inf'))
        self.leftMotor_front.setVelocity(0)

        self.rightMotor_front = self.robot.getMotor('wheel2')
        self.rightMotor_front.setPosition(float('inf'))
        self.rightMotor_front.setVelocity(0)

        # Rear wheels
        self.leftMotor_rear = self.robot.getMotor('wheel3')
        self.leftMotor_rear.setPosition(float('inf'))
        # self.leftMotor_rear.setVelocity(0)

        self.rightMotor_rear = self.robot.getMotor('wheel4')
        self.rightMotor_rear.setPosition(float('inf'))
        # self.rightMotor_rear.setVelocity(0)

        self.motorMaxSpeed = self.leftMotor_rear.getMaxVelocity()

        # Create Subscriber
        self.cmdVelSubscriber = self.create_subscription(Twist, 'cmd_vel', self.cmdVel_callback, 1)

    def cmdVel_callback(self, msg):
        wheelGap = 0.1  # in meter
        wheelRadius = 0.04  # in meter

        leftSpeed = ((2.0 * msg.linear.x - msg.angular.z * wheelGap) / (2.0 * wheelRadius))
        rightSpeed = ((2.0 * msg.linear.x + msg.angular.z * wheelGap) / (2.0 * wheelRadius))
        leftSpeed = min(self.motorMaxSpeed, max(-self.motorMaxSpeed, leftSpeed))
        rightSpeed = min(self.motorMaxSpeed, max(-self.motorMaxSpeed, rightSpeed))

        self.leftMotor_front.setVelocity(leftSpeed)
        self.rightMotor_front.setVelocity(rightSpeed)
        self.leftMotor_rear.setVelocity(leftSpeed)
        self.rightMotor_rear.setVelocity(rightSpeed)

    def sensor_callback(self):
        # Publish distance sensor value
        msg_right = Float64()
        msg_right.data = self.right_sensor.getValue()
        self.sensorPublisher_right.publish(msg_right)
```

Selanjutnya terdapat node lain bernama master.py dimana kita akan mencoba menganalisis data apa saja yang diberikan dari slave.py. Disini kita dapat mengaturrobot kita untuk mencoba mengambil direction sendiri.

```
import rclpy
from rclpy.node import Node
from std_msgs.msg import Float64
from geometry_msgs.msg import Twist


class LineFollower(Node):
    def __init__(self):
        super().__init__('linefollower_cmdvel')
        # Subscribe IR sensors
        self.subs_right_ir = self.create_subscription(Float64, 'right_IR', self.rightIR_cb, 1)
        self.subs_left_ir = self.create_subscription(Float64, 'left_IR', self.leftIR_cb, 1)
        self.subs_mid_ir = self.create_subscription(Float64, 'mid_IR', self.midIR_cb, 1)
        # Publish cmd vel
        self.pubs_cmdvel = self.create_publisher(Twist, 'cmd_vel', 1)

        # vehicle parameters
        self.speed = 0.2
        self.angle_correction = 0.01

        # Initialize parameters
        self.GS_RIGHT, self.GS_MID, self.GS_LEFT = 0, 0, 0
        self.DeltaS = 0
        self.cmd = Twist()
        self.stop = False
        self.count = 0
        self.count_threshold = 10

    def LineFollowingModule(self):...

    # Call backs to update sensor reading variables
    def rightIR_cb(self, msg):...

    def leftIR_cb(self, msg):...

    def midIR_cb(self, msg):...


def main(args=None):...
```

Selanjutnya terdapat line_following_launch.py dimana berfungsi untuk menyimpan packages directory yang sudah kita buat sebelumnya. Lalu setelah semuanya sudah siap jangan lupa untuk memasukkannya pada setup.py



```python
import os
from launch.substitutions.path_join_substitution import PathJoinSubstitution
from launch.actions import IncludeLaunchDescription
from launch.launch_description_sources import PythonLaunchDescriptionSource
from launch import LaunchDescription
from ament_index_python.packages import get_package_share_directory
from launch_ros.actions import Node


def generate_launch_description():
    package_dir = get_package_share_directory('webots_ros2_tutorials')
    core_dir = get_package_share_directory('webots_ros2_core')
    webots = IncludeLaunchDescription(
        PythonLaunchDescriptionSource(
            os.path.join(core_dir, 'launch', 'robot_launch.py')
        ),
        launch_arguments=[
            ('package', 'webots_ros2_tutorials'),
            ('executable', 'enable_robot'),
            ('world', PathJoinSubstitution([package_dir, 'worlds', 'custom_line_follower.wbt'])),
        ]
    )

    Line_follower = Node(
            package='webots_ros2_tutorials',
            executable='line_follower',
            name='master_node'
        )

    return LaunchDescription([
        webots,
        Line_follower
    ])
```