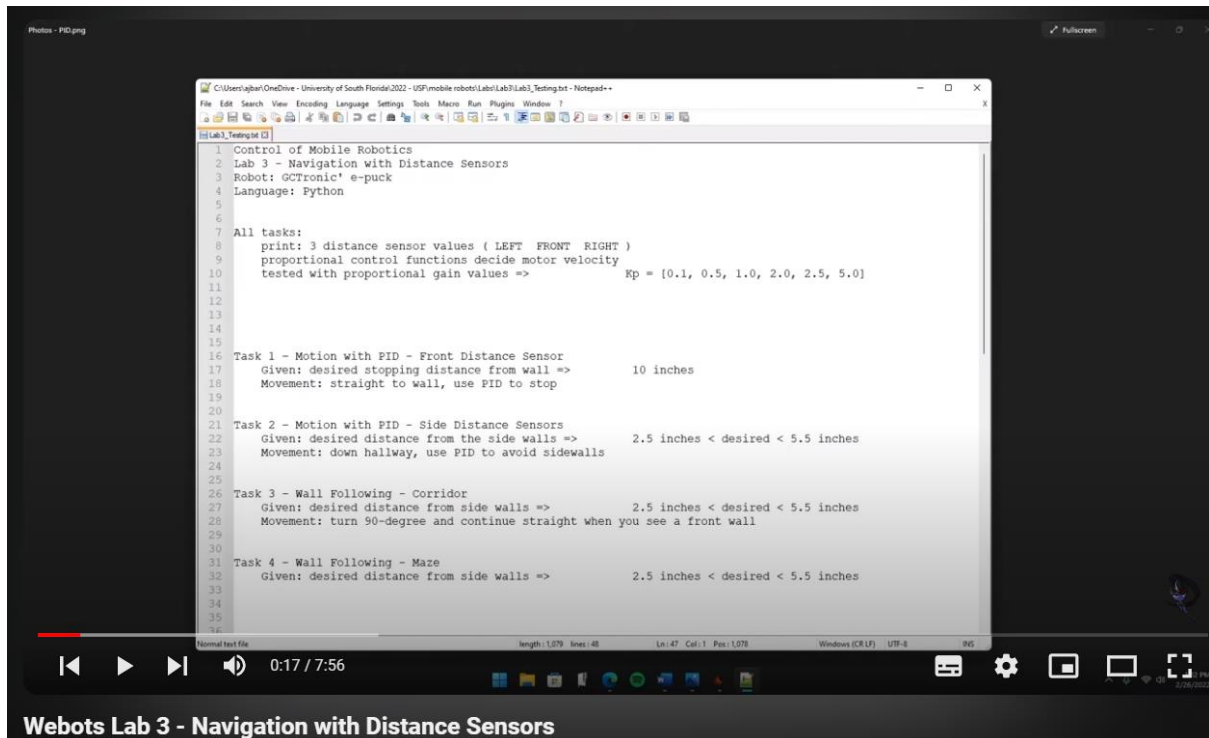


Nama : Andi Aswan

Nim : 1103204095

TUGAS ROBOTIC VIDEO 3 LECTURE 6

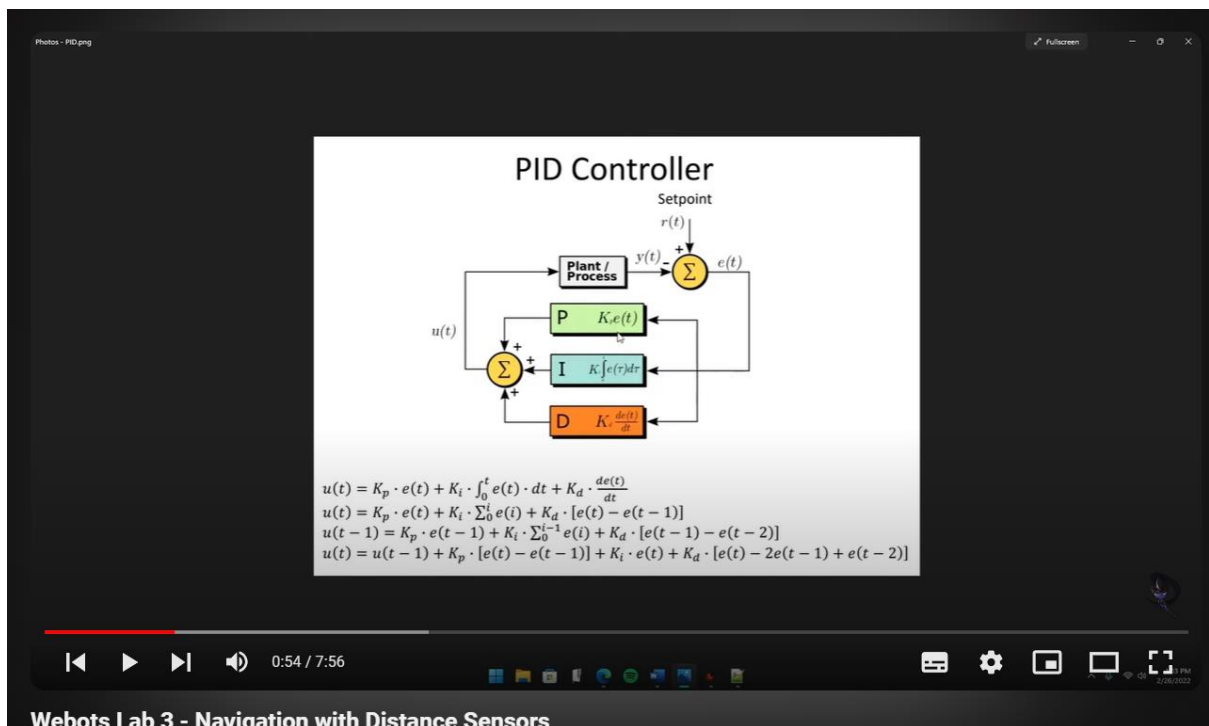
Langkah Pertama, sesuai dengan Task 1 yaitu Front Distance Sensor atau Kontrol PID sistem kontrol umpan balik yang banyak digunakan dalam rekayasa dan robotika untuk mengendalikan gerakan suatu sistem.



The screenshot shows a video player interface with a Notepad++ window open. The Notepad++ window contains a list of tasks for a mobile robot. The tasks are as follows:

- 1 Control of Mobile Robotics
- 2 Lab 3 - Navigation with Distance Sensors
- 3 Robot: GCTronic' e-puck
- 4 Language: Python
- 5
- 6
- 7 All tasks:
- 8 print: 3 distance sensor values (LEFT FRONT RIGHT)
- 9 proportional control functions decide motor velocity
- 10 tested with proportional gain values => $K_p = [0.1, 0.5, 1.0, 2.0, 2.5, 5.0]$
- 11
- 12
- 13
- 14
- 15
- 16 Task 1 - Motion with PID - Front Distance Sensor
- 17 Given: desired stopping distance from wall => 10 inches
- 18 Movement: straight to wall, use PID to stop
- 19
- 20
- 21 Task 2 - Motion with PID - Side Distance Sensors
- 22 Given: desired distance from the side walls => 2.5 inches < desired < 5.5 inches
- 23 Movement: down hallway, use PID to avoid sidewalls
- 24
- 25
- 26 Task 3 - Wall Following - Corridor
- 27 Given: desired distance from side walls => 2.5 inches < desired < 5.5 inches
- 28 Movement: turn 90-degree and continue straight when you see a front wall
- 29
- 30
- 31 Task 4 - Wall Following - Maze
- 32 Given: desired distance from side walls => 2.5 inches < desired < 5.5 inches
- 33
- 34
- 35
- 36

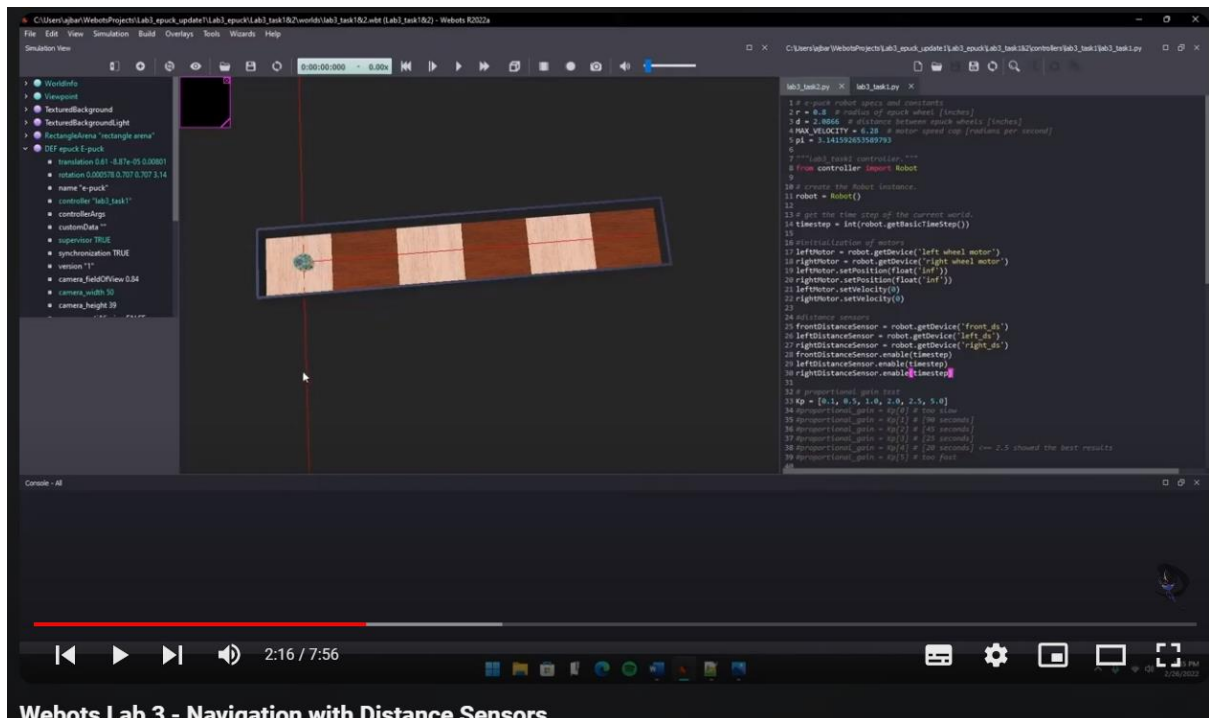
The video player interface shows a progress bar at 0:17 / 7:56 and a title bar at the bottom that reads "Webots Lab 3 - Navigation with Distance Sensors".



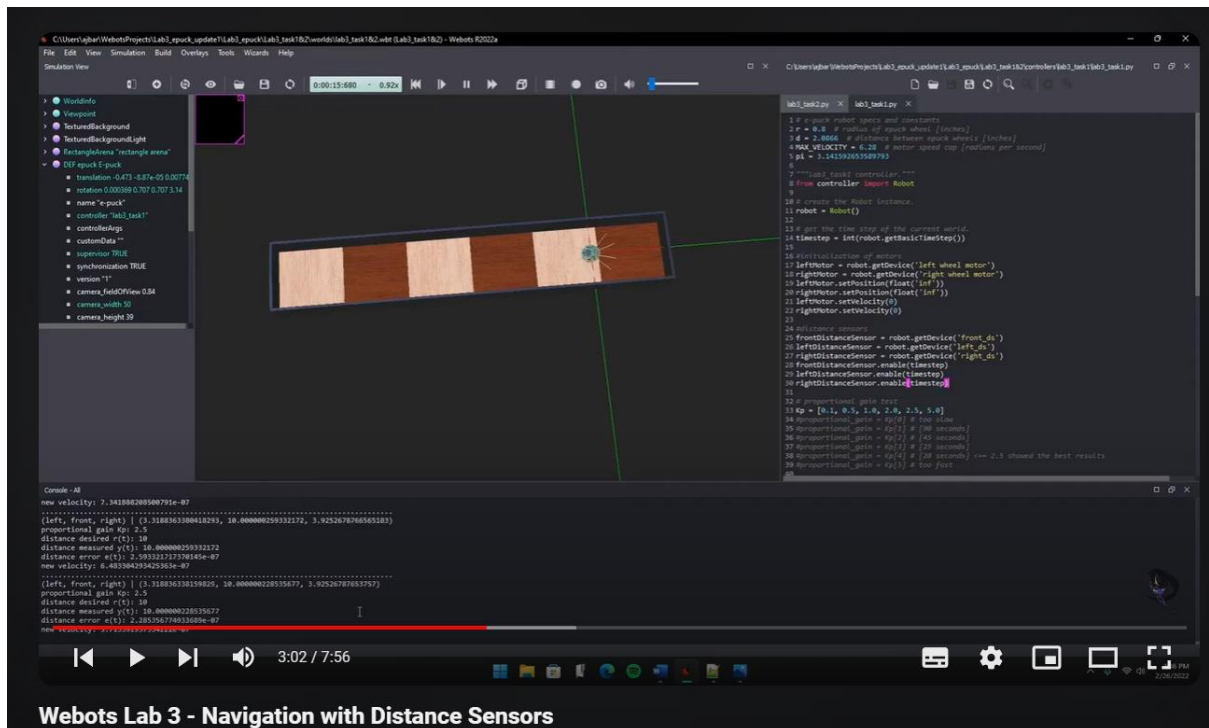
The screenshot shows a video player interface with a diagram of a PID Controller. The diagram is a block diagram of a PID control system. It includes a "Setpoint" input $r(t)$ and a "Plant / Process" block. The output of the plant is $y(t)$, which is compared with the setpoint to produce the error signal $e(t)$. The error signal is then processed by three parallel blocks: a Proportional (P) block with gain K_p , an Integral (I) block with gain K_i and an integrator \int , and a Derivative (D) block with gain K_d and a differentiator $\frac{d}{dt}$. The outputs of these three blocks are summed to produce the control signal $u(t)$, which is fed back into the plant. The diagram is labeled "PID Controller".

The video player interface shows a progress bar at 0:54 / 7:56 and a title bar at the bottom that reads "Webots Lab 3 - Navigation with Distance Sensors".

Langkah Kedua yaitu menjalankan task 2 yaitu Side Distance Sensor menggunakan simulasi robotic pada aplikasi Webots Project.



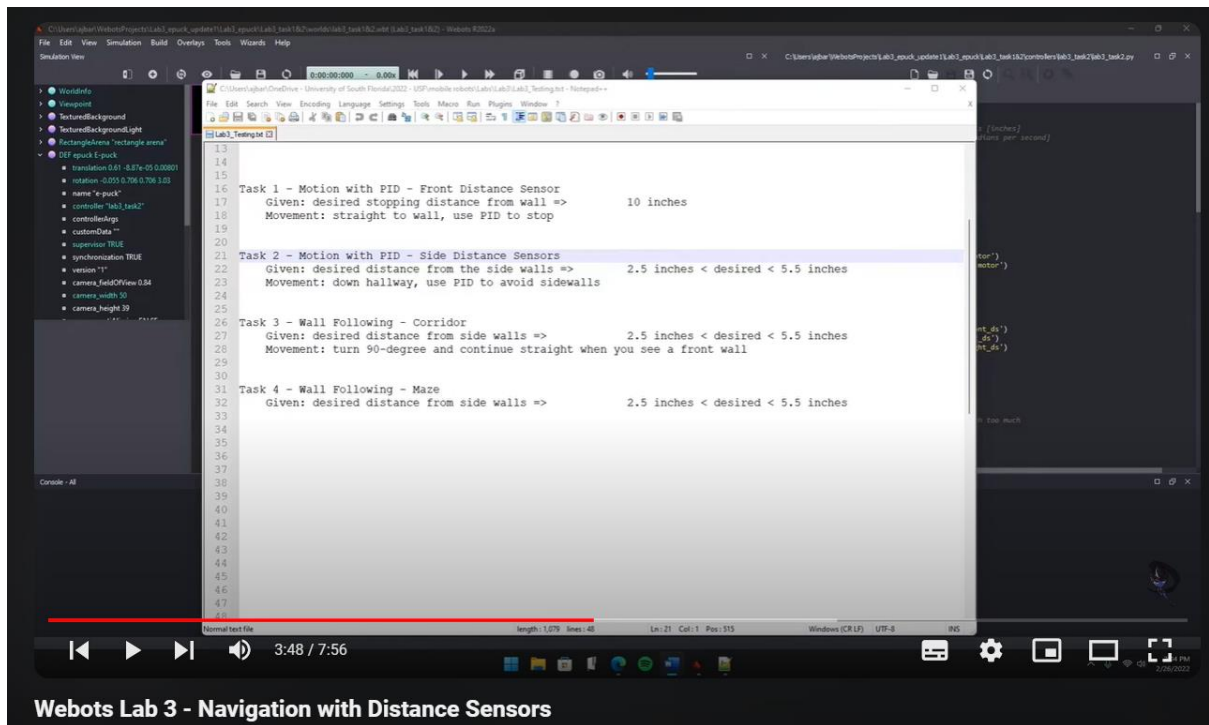
Webots Lab 3 - Navigation with Distance Sensors



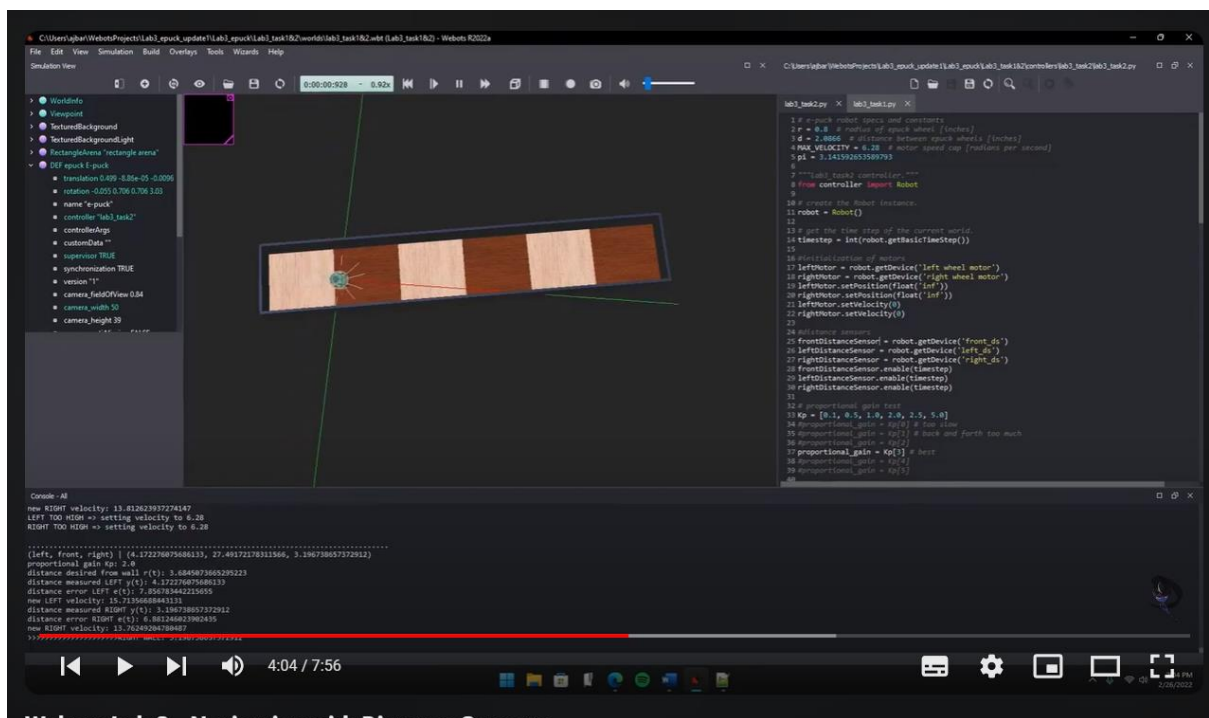
Webots Lab 3 - Navigation with Distance Sensors

Seperti pada beberapa pada gambar diatas, sensor yang terjadi pada robot tersebut yaitu berguna menjaga batas lintasan dengan robot yang bergerak dari kanan ke kiri.

Langkah Ketiga ialah pada task ketiga yang berjudul Wall Following Corridor yaitu melakukan simulasi robot sama seperti pada task 2 bedanya ialah dilakukan dengan lintasan yang lebih luas.

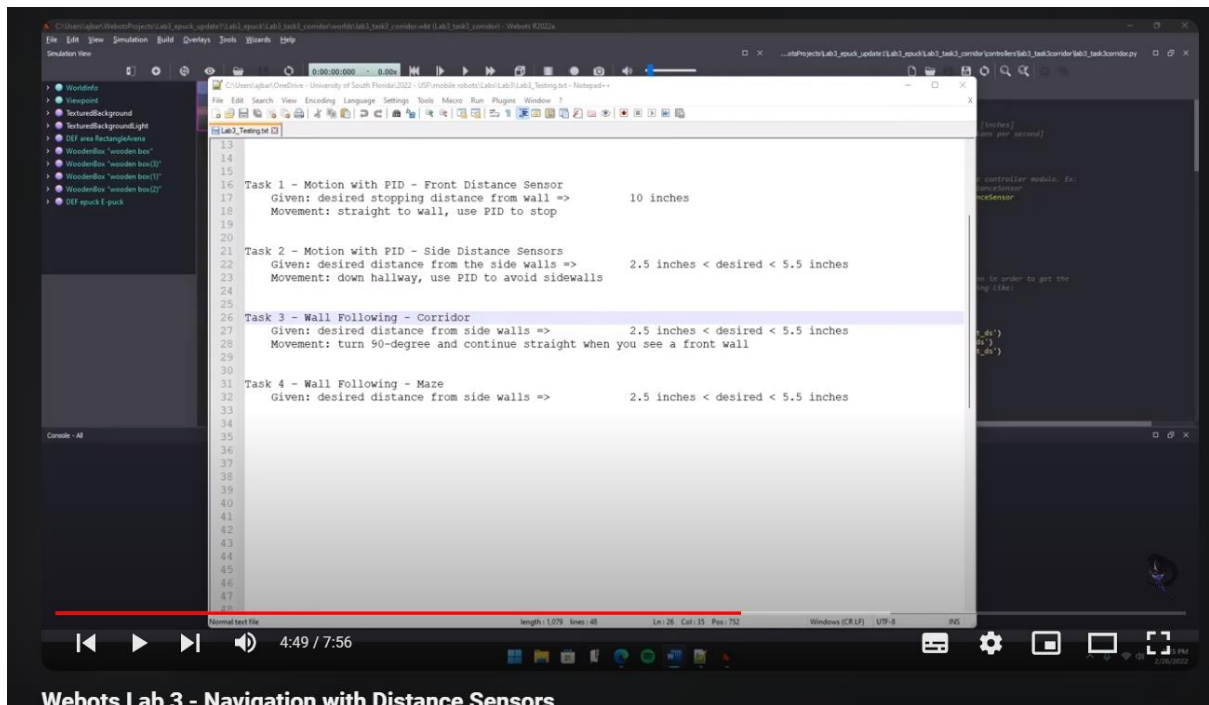


Webots Lab 3 - Navigation with Distance Sensors

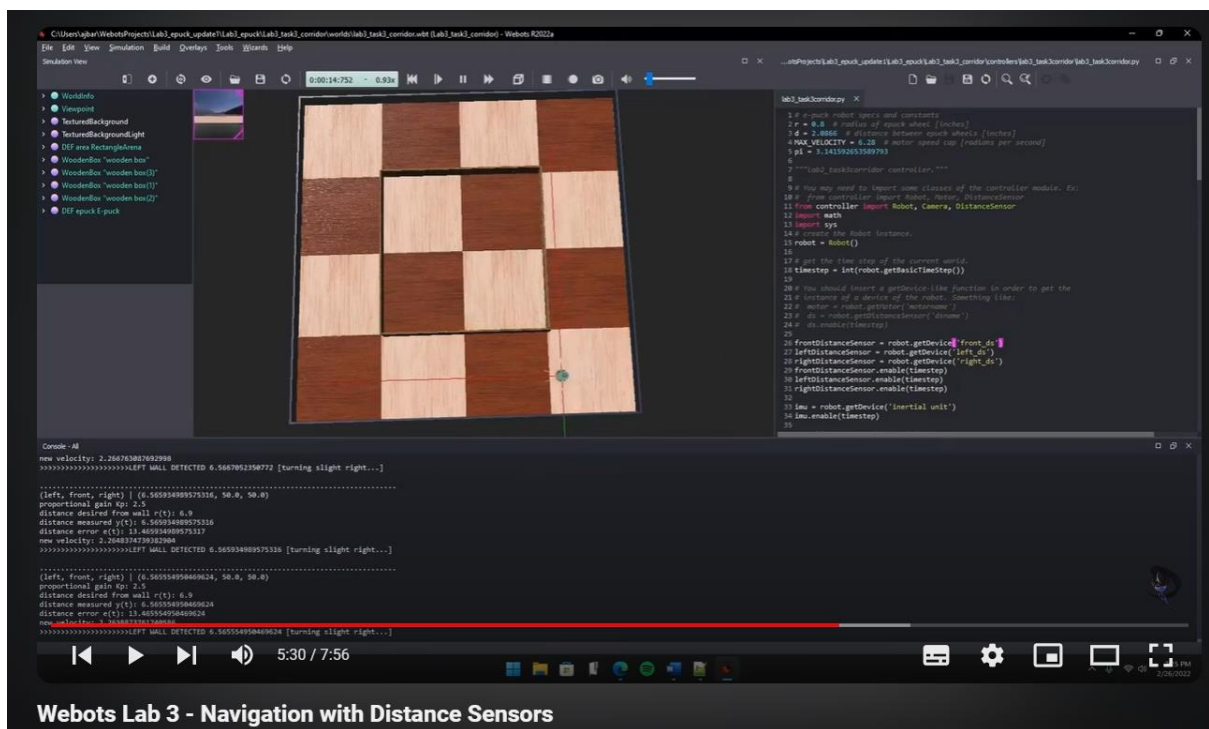


Webots Lab 3 - Navigation with Distance Sensors

Langkah Keempat yaitu maze robot diperintahkan seperti task ketiga bedanya pada task ini robot bergerak tidak beraturan.



Webots Lab 3 - Navigation with Distance Sensors



Webots Lab 3 - Navigation with Distance Sensors

C:\Users\ghar\WebotsProject\lab3_epuck\update\lab3_epuck\lab3_tank4_maze\world\lab3_tank4_maze.wld (lab3_tank4_maze) - Webots R2022a

Simulation View

5:01:29.896 - 6.11s

lab3_tank4_maze.py

```
1 # Create robot object and constants
2 r = 0.8 # radius of epuck wheel [meters]
3 d = 2.0000 # distance between epuck wheels [meters]
4 MAX_VELOCITY = 6.28 # motor speed cap [radians per second]
5 pi = 3.141592653589793
6
7 """lab3_tank4_maze controller"""
8 from controller import Robot, Camera, DistanceSensor
9
10 # Create the robot instance.
11 robot = Robot()
12
13 # Get the time step of the current world.
14 timestep = int(robot.getBasicTimeStep())
15
16 # Create distance sensors.
17 frontDistanceSensor = robot.getDevice('front_ds')
18 leftDistanceSensor = robot.getDevice('left_ds')
19 rightDistanceSensor = robot.getDevice('right_ds')
20 frontDistanceSensor.enable(timestep)
21 leftDistanceSensor.enable(timestep)
22 rightDistanceSensor.enable(timestep)
23
24
25 # Create camera and recognition
26 camera = robot.getDevice('camera1')
27 camera.enable(timestep)
28 camera.recognitionEnable(timestep)
29
30 # Set motor limits to infinity and set target position to infinity
31 leftMotor = robot.getDevice('left wheel motor')
32 rightMotor = robot.getDevice('right wheel motor')
33 leftMotor.setPosition(float('inf'))
34 rightMotor.setPosition(float('inf'))
35 leftMotor.setVelocity(0)
36 rightMotor.setVelocity(0)
```

Console #8

```
distance measured y(t): 16.245728161115213
distance error e(t): 23.145728161115215
new velocity: 17.86412846078088
.....
(left, front, right) | (16.186237461715374, 28.646896602791668, 16.393752746288328)
proportional gain Kp: 2.5
distance desired from wall r(t): 6.9
distance measured y(t): 16.186237461715374
distance error e(t): 23.686237461715373
new velocity: 57.715939584268435
.....
(left, front, right) | (16.126314153942683, 28.53249170461123, 16.402189742628373)
proportional gain Kp: 2.5
distance desired from wall r(t): 6.9
distance measured y(t): 16.126314153942683
distance error e(t): 23.426314153942685
new velocity: 57.36378532643574
```

7:41 / 7:56

Webots Lab 3 - Navigation with Distance Sensors