What are whycon markers and how do they work?

WhyCon is a version of a vision-based localization system that can be used with low-cost web cameras, and achieves millimeter precision with very high performance. The system is capable of efficient real-time detection and precise position estimation of several circular markers in a video stream. WhyCon is meant as an alternative to widely used and expensive localization systems. It is fully open-source.

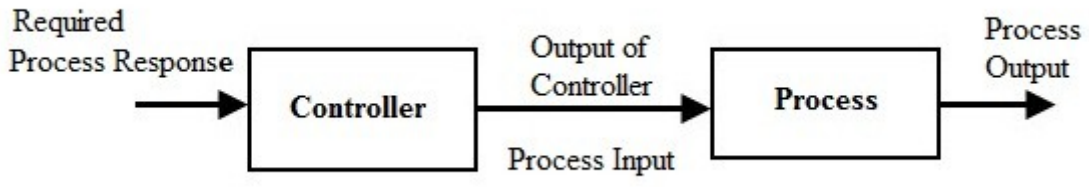
At stage 1 the algorithm perform flood fill technique and search for continuous black segments and then it The ratio of each segment’s pixels to its bounding box area is then compared to a value calculated from dimensions of the whycon marker. After that it searches for the white pixels to get the centroid and calculates the 3D position .then the other whycon markers are detected and the distance between each markers are calculated and using those data the localisation of the robot is done.

What are AruCo markers and how do they work?

Aruco markers are markers which are placed on objects and scenes for uniquely identifying based on the pattern on the marker. Each marker represents different objects which makes robot easier to identify the particular object associated with the pattern

Open Loop Control System

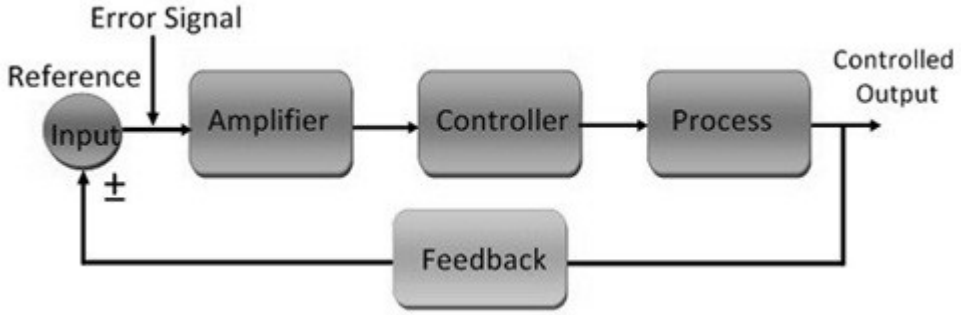
It is system in which the output of the system does not affect the action of the control system .Basically , the system that totally depends on the input and provides output based on the input .So it does not require a feedback .Since it does not check the output it is prone to have errors at the output. And the advantage of the open loop system is that it is quick to react as the control actions are based on the input



Closed-Loop Control System

In this system the action of the control systems is based on the error, which is the difference in input and output signal. As the control system requires output signal to calculate the error, this system requires a feedback node. This control system is not as fast as open loop control system as the output is based

on previous cycle(refers that the true output is produced at second cycle) .The advantage is that the errors caused due to some external factors can be reduced.



PID controllers

PID (proportional integral derivative) controllers acts based on the errors between set point and measured value. Each part in PID controller contributes for minimising the error. Each part of the PID controller has constants or multiplier and it has to be tuned properly to get desired output.

The proportional part just multiplies the error with the multiplier.

The integral part integrates over the error.

The derivative part tries to reduce the slope in order to reduce overshoot.

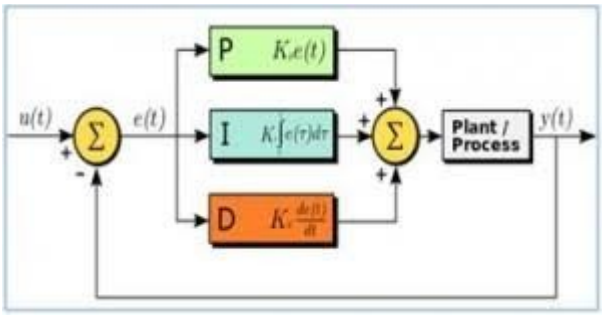
Optimal way of tuning PID controller

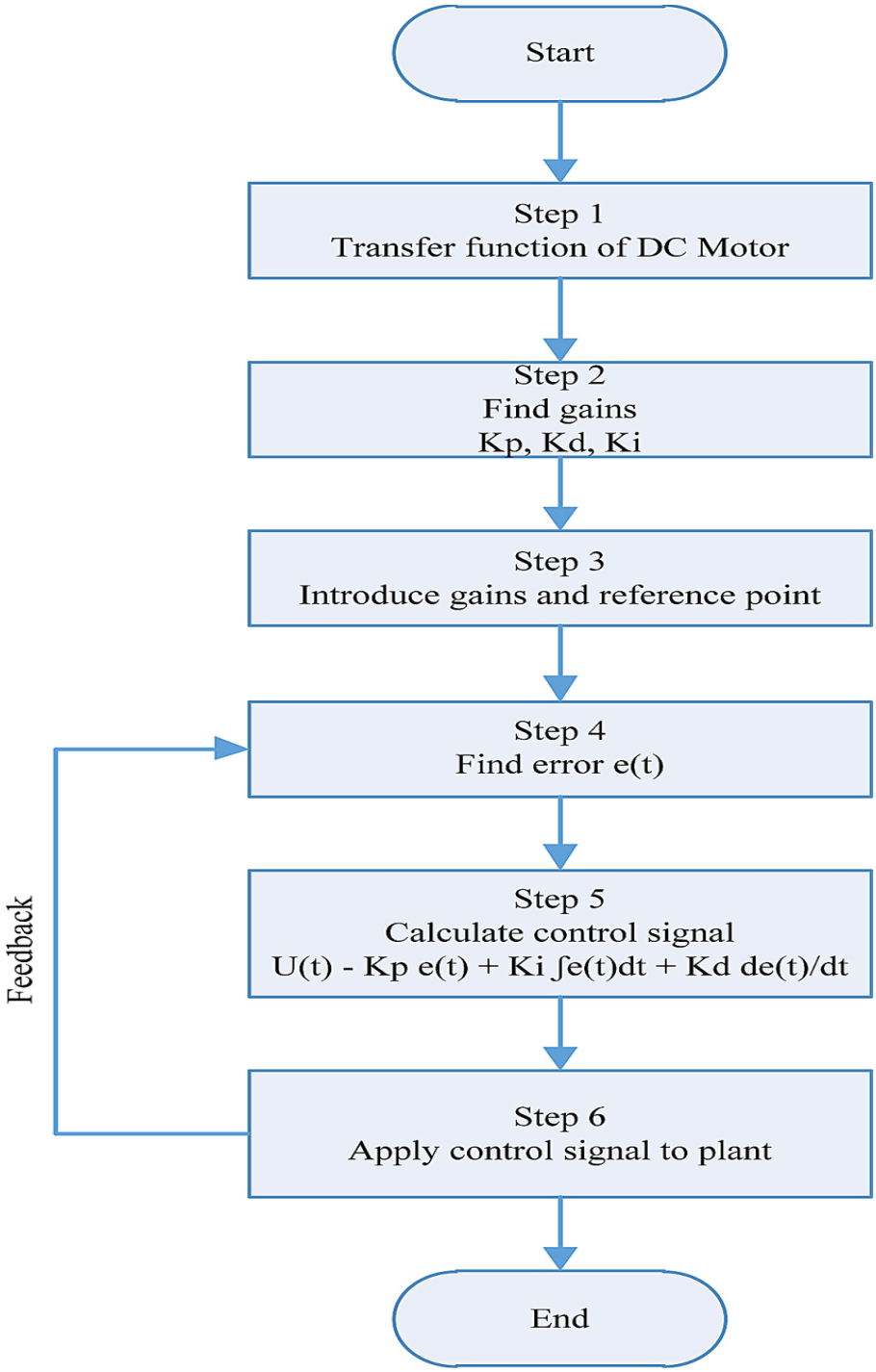
Rule Name	Tuning Parameters
Classic Ziegler-Nichols	Kp = 0.6 Ku Ti = 0.5 Tu Td = 0.125 Tu
Pessen Integral Rule	Kp = 0.7 Ku Ti = 0.4 Tu Td = 0.15 Tu
Some Overshoot	Kp = 0.33 Ku Ti = 0.5 Tu Td = 0.33 Tu
No Overshoot	Kp = 0.2 Ku Ti = 0.5 Tu Td = 0.33 Tu

the period Tu of the oscillation frequency at the stability limit

the gain margin Ku for loop stability

Flowchart for working of PID controller

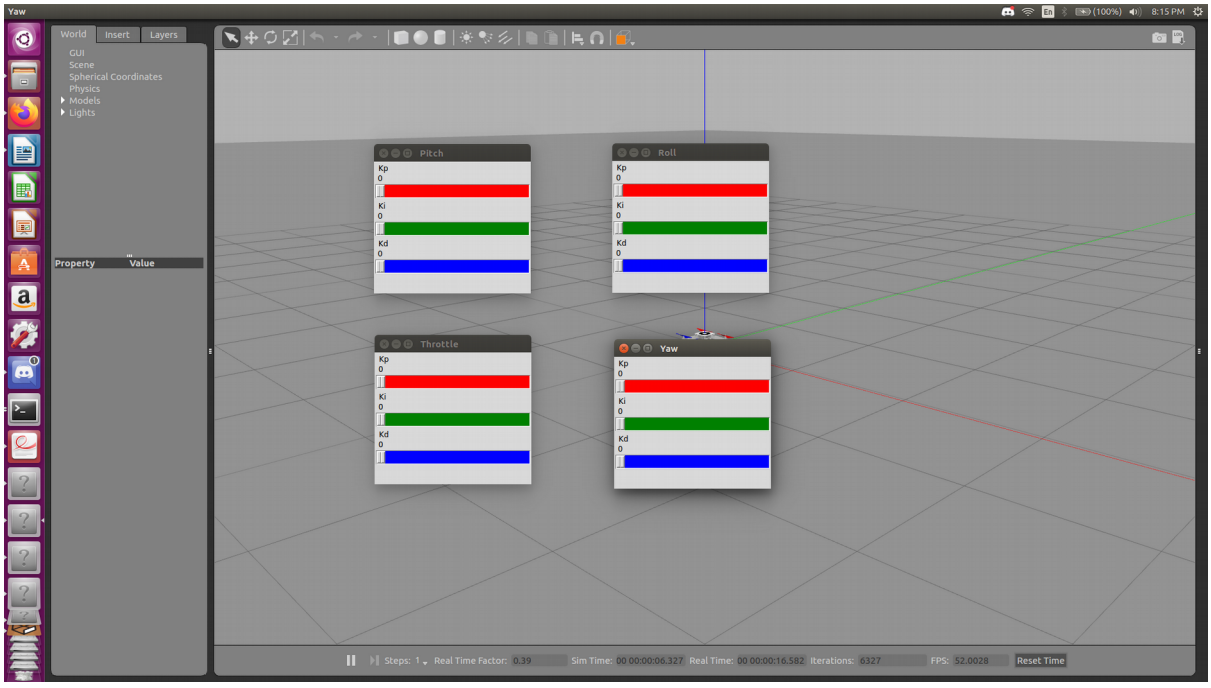

P
I
D
Plant / Process
u(t)
e(t)
y(t)

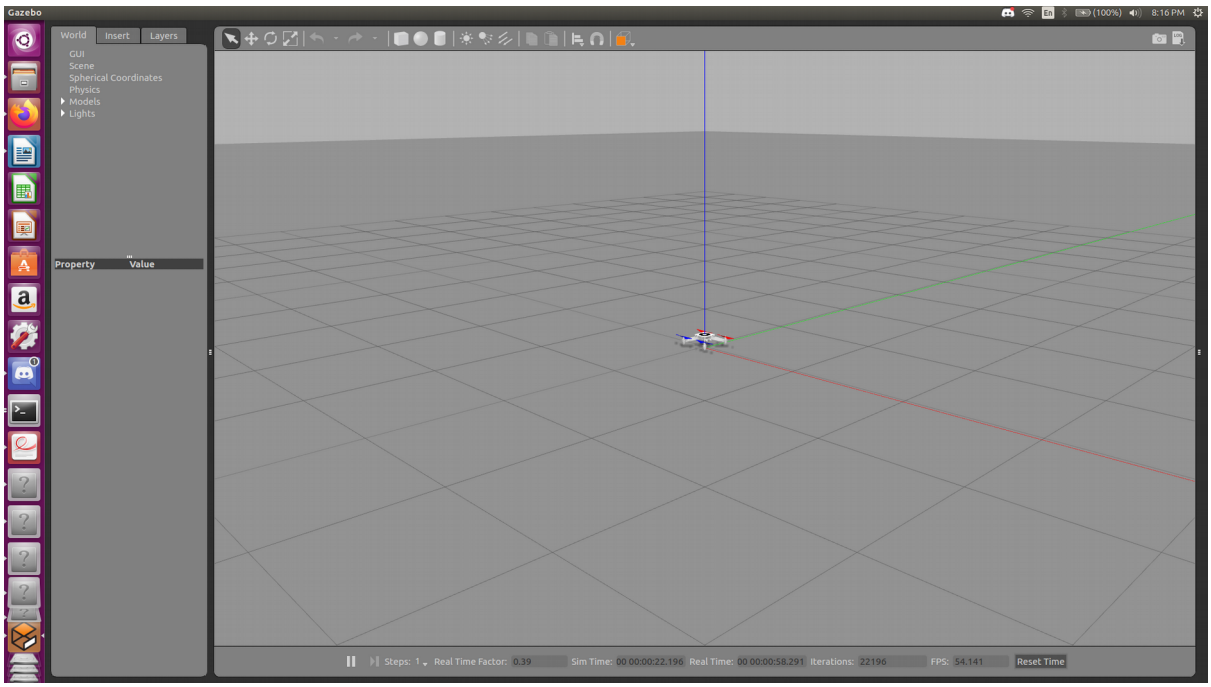

Start
Step 1
Transfer function of DC Motor
Step 2
Find gains
Kp, Kd, Ki
Step 3
Introduce gains and reference point
Step 4
Find error e(t)
Step 5
Calculate control signal
U(t) - Kp e(t) + Ki ∫e(t)dt + Kd de(t)/dt
Step 6
Apply control signal to plant
End
Feedback

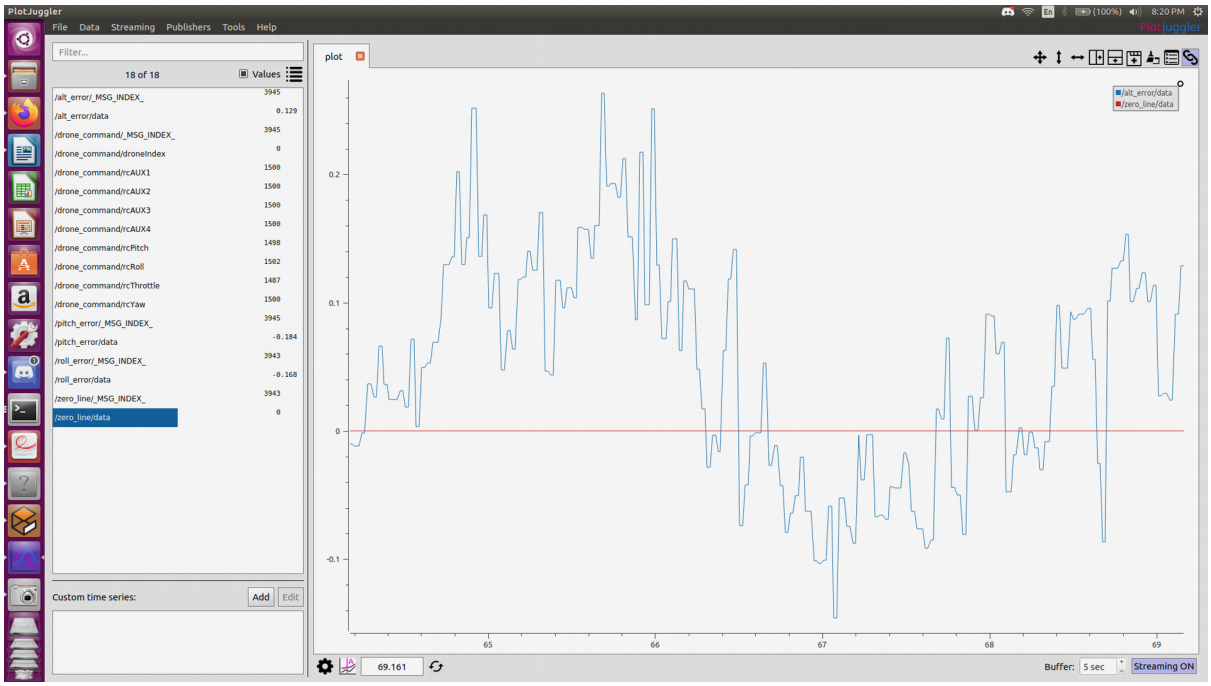
task_2: implementation and tuning of PID controller

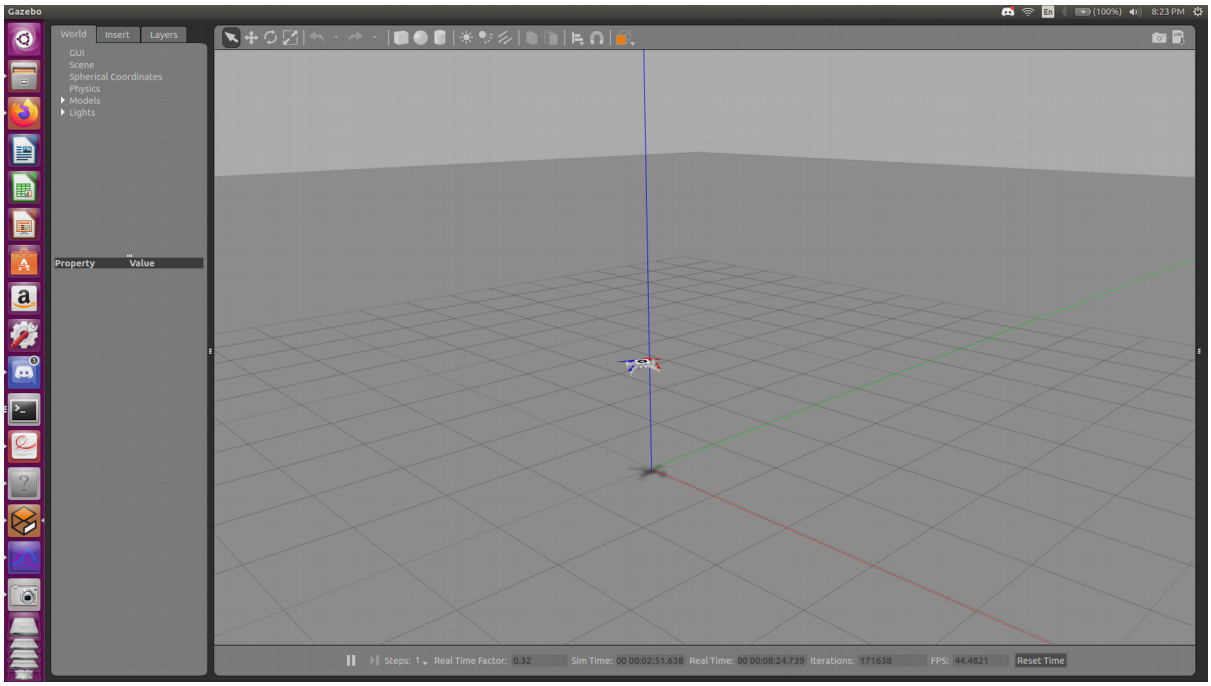
In this week we had to build a PID controller for the drone given to us and the objective of this week is to make the drone fly and stay in the given position.The controller used for the task is a PID controller.The most important part in this task is to tune the PID controller. The implementation of the PID controller code was refered from http://brettbeauregard.com/blog/2011/04/improving-the-beginners-pid-introduction/ a blog which explains how to develop aPID controller for beginers . Then the tuning is done with the help of plotjuggler, which is a graph visualisation tool and PID tune, which is a python script .The python script displays 4 dialog boxes with control bars for setting the value of KP,KI,KD for throttle pitch and roll

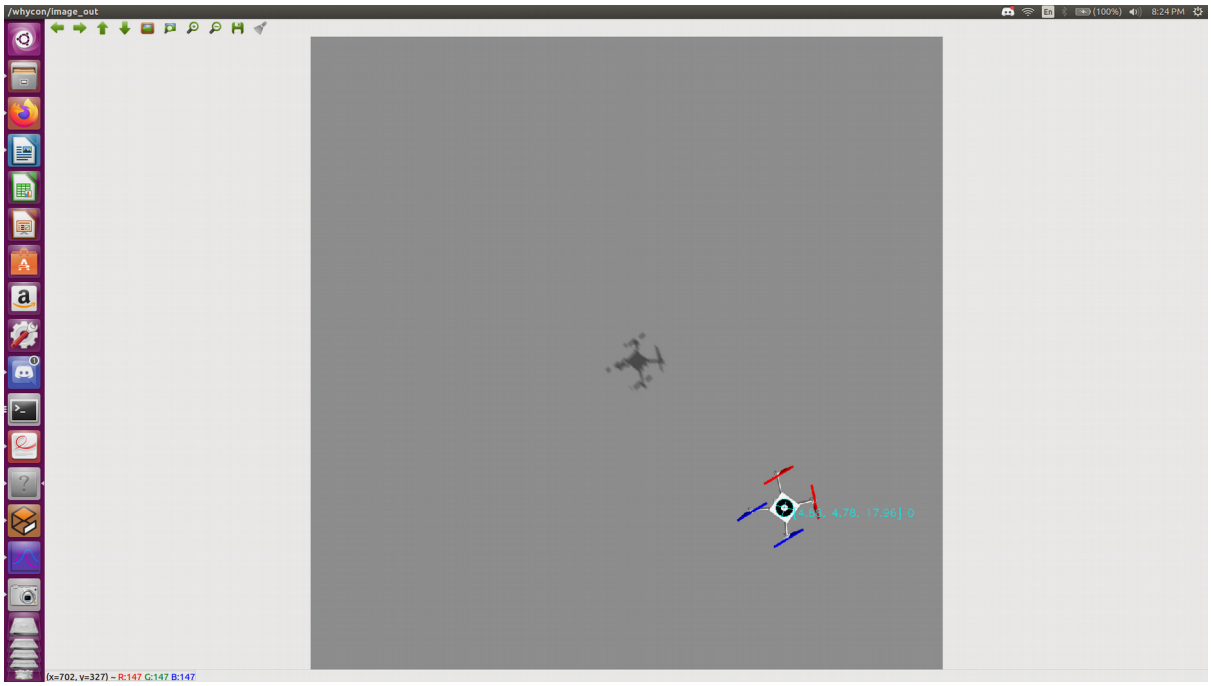
Then KP,KI,KD for each of the parameters(throttle pitch and roll) have to be tuned and the output can be viewed on the graph (plotjuggler) .You have to tune for one axis at a tine (eg. First tune the throttle then the pitch and then the roll).All shouldn’t be done together. After tuning substitute the values of KP,KI,KD in the PID controller script .




Gazebo
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Models
Lights
Property
Value
Steps: 1
Real Time Factor: 0.39
Sim Time: 00 00:00:22.196
Real Time: 00 00:00:58.291
Iterations: 22196
FPS: 54.141
Reset Time


PlotJuggler
File
Data
Streaming
Publishers
Tools
Help
Filter...
18 of 18
Values
/alt_error/_MSG_INDEX_ 3945
/alt_error/data 0.129
/drone_command/_MSG_INDEX_ 3945
/drone_command/droneIndex 0
/drone_command/rcAUX1 1500
/drone_command/rcAUX2 1500
/drone_command/rcAUX3 1500
/drone_command/rcAUX4 1500
/drone_command/rcPitch 1498
/drone_command/rcRoll 1502
/drone_command/rcThrottle 1487
/drone_command/rcYaw 1500
/pitch_error/_MSG_INDEX_ 3945
/pitch_error/data -0.184
/roll_error/_MSG_INDEX_ 3943
/roll_error/data -0.168
/zero_line/_MSG_INDEX_ 3943
/zero_line/data 0
Custom time series:
Add
Edit
plot
/alt_error/data
/zero_line/data
0.2
0.1
0
-0.1
65
66
67
68
69
69.161
Buffer: 5 sec
Streaming ON


Gazebo
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Models
Lights
Property
Value
Steps: 1
Real Time Factor: 0.32
Sim Time: 00 00:02:51.636
Real Time: 00 00:08:24.739
Iterations: 171638
FPS: 44.4821
Reset Time


/whycon/image_out
(x=702, y=327) – R:147 G:147 B:147

References

Open and Closed loop control system - https://www.elprocus.com/difference-between-open-loop-closed-loop-control-system/#:~:text=The%20main%20difference%20between%20open,depends%20on%20the%20controlled%20act.

PID controller - https://en.wikipedia.org/wiki/PID_controller

Optimal way of tuning PID controller - http://www.mstarlabs.com/control/znrule.html#Ref4

Whycon marker - https://core.ac.uk/download/pdf/42583963.pdf

http://brettbeauregard.com/blog/2011/04/improving-the-beginners-pid-introduction/

AruCo markers - https://www.learnopencv.com/augmented-reality-using-aruco-markers-in-opencv-c-python/#:~:text=An%20aruco%20marker%20is%20a,in%20a%20variety%20of%20sizes.

PID flowchart1 – https://upload.wikimedia.org/wikipedia/commons/thumb/4/43/PID_en.svg/400px-PID_en.svg.png

PID flowchart2 - https://www.researchgate.net/profile/Sohaib_Aslam2/publication/307092705/figure/fig3/AS:400324838871042@1472456287510/Flow-Chart-of-PID-Implementation.png

Task_2

Implementation of PID controller: http://brettbeauregard.com/blog/2011/04/improving-the-beginners-pid-introduction/

eyantra_drone documentation: http://wiki.ros.org/eyantra_drone

plotjuggler cheatsheet: https://www.plotjuggler.io/cheatsheet

Videos regarding PID controller :

1. https://www.youtube.com/watch?v=wkfEZmsQqiA&list=PLn8PRpmsu08pQBgjxYFXSsODEF3Jqmm-y

2. https://www.youtube.com/watch?v=UR0hOmjaHp0