Nama : Muhammad Athallah Yakarazi
NIM : 24/532752/PA/22532
Divisi : Programmer

Apakah perbedaan open-loop system dan close-loop system?

Open-loop system adalah sistem dimana sinyal output tidak di feedback ke input sistem. Oleh karena itu, sistem kendali open-loop juga disebut sebagai sistem kendali non-feedback.

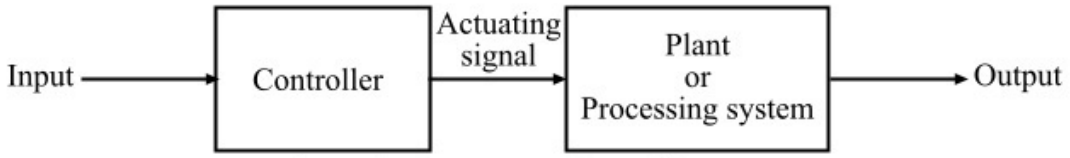


Figure 1 - Open Loop Control System

Close-loop system adalah sistem dimana sinyal output di feedback ke input sistem. Oleh karena itu, dalam closed-loop system, aksi kendali merupakan fungsi dari output yang diinginkan.

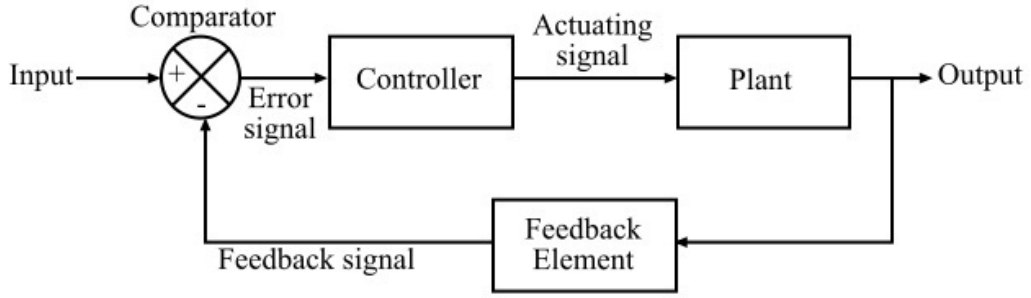


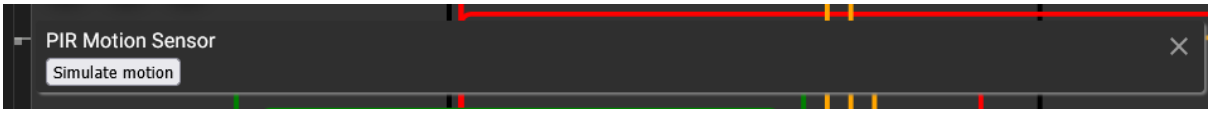
Figure 2 - Closed Loop Control System

Apakah sistem yang anda kerjakan dan simulasikan di atas termasuk open-loop system atau close-loop system? Justifikasi jawaban anda!

Closed-loop karena sinyal-sinyal output dari sensor di feedback ke input system untuk mengkontrolnya. Jadi, dari output sensor MPU6050 yang mengukur kondisi sistem terkini yaitu roll, pitch, dan yaw, ini jadi input ke ESP32 yang membandingkannya dengan kondisi yang diinginkan (INITIAL_POS 90 degree). Nah terus dari hasil perbandingan ini, dia menghitung dan mengirim command korektif ke servo-servo untuk mengurangi perbedaan (minimise the error) dari kondisi kini dengan yang diinginkan. Ini mirip dengan cara kerja autopilot.

Misalnya nih, kalau MPU6050 bilang ada roll 10 degree, hasil ini akan secara aktif digunakan untuk meng-command servo 1 dan servo 2 untuk bergerak reaktif dari roll itu. Nah, karena feedback loop ini, jadi ini system close-loop.

Jelaskan fungsi masing-masing sensor yang digunakan pada sistem di atas!

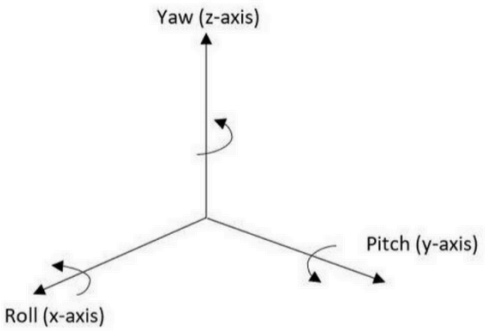


Accelerometer mengukur percepatan di ketiga sumbu x, y, z, normalnya di sumbu z (atas-bawah) nilainya 1g karena gravitasi Bumi. Nah, kalau x dan y tidak 0 maka sistem tidak tegak, hitungan fisisnya bisa dihitung pake freebody diagram dan hasil rumusnya ini dipakai untuk mendeteksi seberapa miring dan terus untuk menggerakan servo 1, 2, 3, dan 4.

Rotation mengukur kecepatan rotasi sistem di ketiga sumbu. Kalau di sistem hanya dipakai satu sumbu yaitu z untuk mengukur yawing.

rotasi.

- Bidang rotasi *roll, pitch, yaw*, dapat diperhatikan pada diagram berikut:



Sama halnya dengan accelerometer, hasil dari sensor ini dipakai untuk menggerakan servo 5.

Ada sensor temperatur juga di MPU6050, tapi hasil sensor ini tidak digunakan untuk menggerakan servo-servo. Tapi, kalau di dunia nyata ini berpengaruh sih, karena temperatur mempengaruhi densitas fluida, kalau misal ini udara, sistem kontrol pesawat perlu meng-utilise ini.

Terus akhirnya ada PIR motion sensor yang outputnya ini diskrit 0 atau 1, kalau 1 berarti dia mendeteksi suatu gerakan eksternal. Hasil outputnya ini akan membuat semua servo bergerak, dan arah geraknya terserah aku.

Jelaskan alasan, fungsi, dan arah tuju koneksi setiap pin ESP32 yang dimanfaatkan dari skematik (poin 5 Informasi Pengerjaan) yang telah anda buat!

Pin ESP32	Arah Tuju
5V	- VCC PIR - VCC MPU6050 - V+ SERVO 1 - V+ SERVO 2 - V+ SERVO 3 - V+ SERVO 4 - V+ SERVO 5
GND	- GND PIR - GND MPU6050 - GND SERVO 1 - GND SERVO 2 - GND SERVO 3 - GND SERVO 4 - GND SERVO 5
4	PWM SERVO 5
5	PWM SERVO 4
16	PWM SERVO 2
17	PWM SERVO 1
18	PWM SERVO 3
19	D_OUT PIR
21	SDA MPU6050
22	SCL MPU6050

Fungsi dan alasan:

1. **5V**

   Untuk memberi daya device-device yang terhubung, alasanku menggunakan pin 5V untuk memberi daya device-device yang terhubung dan lebih convenient waktu membuatnya di wokwi untuk menghubungkan pin daya semua device ke pin 5V, and it is also easier to organise the wiring.
2. **GND**

   Untuk ground device-device yang terhubung, alasanku menggunakan satu pin GND ini untuk semua mirip seperti pin 5V.
3. **GPIO** 4

GPIO 4 aku gunakan untuk PWM servo 5, ini adalah pilihan bebas yang aku pilih karena peletakkan servo-servo ku di awal membuat pin ini easier to organise the wiring.

4. **GPIO 5**

   Alasan yang sama dengan sebelumnya.

5. **GPIO 16**

   Alasan yang sama dengan sebelumnya.

6. **GPIO 17**

   Alasan yang sama dengan sebelumnya.

7. **GPIO 18**

   Alasan yang sama dengan sebelumnya.

8. **GPIO 19**

   GPIO 19 aku set untuk menerima input dari D_OUT PIR. Juga karena ini pin Master In Slave Out.

9. **GPIO 21**

   Kareana untuk ESP32, GPIO 21 adalah standard untuk I2C SDA. SDA (Serial Data Line) adalah data line bidirectional untuk mengirim dan menerima data.

10. **GPIO 22**

   Karena untuk ESP32, GPIO 22 adalah standard untuk I2C SCL. SCL (Serial Clock Line) adalah clock line untuk mengsingkron transfer data.

Dalam suatu rapat monitoring, anda diminta untuk menjelaskan kode yang anda buat ke rekan kerja tim anda yang berbeda divisi dengan anda. Buatlah penjelasan yang mudah dipahami untuk menjelaskan alur bagaimana sistem yang anda program bekerja berdasarkan eksekusi kode yang telah dibuat hingga ke eksekusi yang dilakukan oleh mikrokontroler, sensor, dan aktuator yang ada!

Awalnya #define SERVO1_PIN 17, #define SERVO2_PIN 16, ... dkk. Agar lebih mudah untuk mengganti definisi pin yang terhubung, jadi tidak perlu mengganti semua satu per satu.

Lalu MPU dan servo-servo di deklarasikan, lalu kondisi INITIAL_POS di deklarasikan dengan alasan yang mirip dengan #define SERVO1_PIN 17 dkk. Lalu yawStopTime, isYawing, dan YAW_MOTION_THRESHOLD di deklarasikan untuk memfasilitasi pergerakan servo 5. Aku yakin anotasi kode nya sudah cukup jelas untuk variabel-variabel ini.

Lalu program memasuki setup(). Pin servo-servo di attach dan MPU di tunggu untuk inisialisasi, setelah itu, batas-batas pengukuran sensor-sensor MPU di set. Kemudian, servo-servo di set ke posisi netral mereka yaitu pada 90 degree, sehingga ada batas pergerakan yang mungkin -90 degree hingga 90 degree (0 degree hingga 180 degree).

Lalu program memasuki loop(). Awalnya nilai dari PIR dan sensor didaptakan. Jika ada nilai dari PIR maka servo-servo akan dikendalikan berdasarkan hasil dari PIR tersebut, yaitu arahnya terserah aku, lalu servo-servo akan kembali ke posisi 90 degree. Namun, jika tidak ada nilai dari PIR maka servo-servo akan dikendalikan berdasarkan hasil dari sensor-sensor MPU6050.

Jika servo-servo dikendalikan berdasarkan hasil dari sensor-sensor MPU6050, maka hasil raw dari sensor tersebut akan diolah secara matematis terlebih dahulu agar dihasilkan nilai sudut dalam degree untuk menggerakan servo-servo. Untuk mendapat nilai roll dan pitch dari accelerometer, program melakukan kalkulasi invers tangen dari vektor-vektor percepatan gravitasi. Lalu nilai sudut dari invers tangen ini diubah dari radian ke degree, yang kemudian digunakan untuk mengubah posisi servo dimana

1. Ketika sistem mengalami rotasi pada bidang *roll* ke arah positif, maka servo 1 dan 2 akan berputar melawan arah (arah negatif) dari bidang. Begitu pula sebaliknya, jika sistem mengalami rotasi pada bidang *roll* ke arah negatif, maka servo 1 dan 2 akan berputar melawan arah (arah kanan) dari bidang tersebut.
2. Ketika sistem mengalami rotasi pada bidang *pitch* ke arah positif, maka servo 3 dan 4 akan berputar searah dengan bidang rotasi yang terjadi. Begitu pula dengan sebaliknya, jika sistem mengalami rotasi pada bidang *pitch* ke arah negatif, maka servo 3 dan 4 akan berputar searah dengan bidang rotasi yang terjadi tersebut.

Untuk yawing, nilai yaw rate didapat langsung dari sensor (raw). Nilai raw ini kemudian di check berdasarkan ambang batas apakah ini considered yawing atau tidak. Jika iya yawing, maka servo 5 akan bergerak sesuai dengan nilai yaw rate yang, gerakan servo 5 bisa di scale dengan multiplier. Lalu, jika sistem berhenti yawing, maka timer akan dimulai, dan setelah 1 detik berlalu maka servo 5 akan dikembalikan ke posisi 90 degree, dan timer di reset.