Extra Jawaban Penugasan Week 1 Day 3

Ursula Maurentti Amarely

24/533008/TK/59050

1. Apakah perbedaan *open-loop system* dan *close-loop system*?
   - *Open-loop system* bekerja tanpa menggunakan *feedback* dari *output* untuk melanjutkan aksi kontrol selanjutnya. *Output* tidak memengaruhi *lines of code* yang akan dijalankan kemudian. Hasil akhir dari sebuah program sangat bergantung pada kalibrasi awal, dan sulit untuk dimaintain. Sistem ini mudah untuk diimplementasikan dan biayanya relatif murah. Cocok digunakan pada sistem yang tidak membutuhkan control yang spesifik. Misalnya, televisi dan kipas yang akan terus menyala dengan kecepatan atau dengan volume yang sudah diset sebelumnya tanpa peduli keadaan lingkungannya (apakah televisi masih dilihat, apakah kipas masih dipakai, dll.)
   - Selanjutnya adalah *closed-loop system*, dimana *action* yang atau *command* yang akan dieksekusi sangat dependan terhadap *output*nya. Walaupun implementasinya kompleks, biaya *maintenance* yang mahal, dan juga lebih lambat, closed-loop cocok untuk system yang membutuhkan memberikan presisi dan akurasi yang tinggi, serta membutuhkan shield dari faktor eksternal. Contoh impementasinya dapat dilihat pada AC atau mesin setrika, setelah mencapai suhu spesifik mereka akan mematikan elemen pemanas atau pendingin untuk mempertahankan suhu yang sudah di set oleh penggunanya.

2. Apakah sistem yang anda kerjakan dan simulasikan di atas termasuk *open-loop system* atau *close-loop system*? Justifikasi jawaban anda!
   - Sistem yang saya kerjakan pada project ini termasuk *closed-loop system*. Bisa dengan mudah kita lihat melalui komponen yang digunakan. *Sensor MPU6050* dan *PIR sensor* sendiri berfungsi untuk mengambil data secara real time yang kemudian diproses. Ketika kondisi tertentu dideteksi, output spesifik akan dijalankan. Ini merupakan konsep dasar dari *closed-loop system* dimana kondisi eksternal akan memengaruhi keluaran.

3. Jelaskan fungsi masing-masing sensor yang digunakan pada sistem di atas!
   - Ada 2 sensor yang digunakan pada sistem ini. Pertama ada *sensor MPU6050* untuk mengukur dan mendeteksi orientasi, percepatan, dan gerakan rotasi secara *real-time*. Komponen dalam sensor ini terdiri dari *gyroscope* dan *accelerometer*. *Gyroscope* bertugas untuk mengukur kecepatan sudut serta rotasi pada bidang *roll, pitch,* dan *yaw.* Sedangkan *accelerometer* bekerja untuk mendeteksi kecepatan linear pada ketiga sumbu (X, Y, Z).
   - Sensor kedua adalah *PIR* atau *Passive Infrared*. Ini berguna untuk mendeteksi gerakan eksternal di lingkungan sekitar. Sensor ini bekerja dengan mendeteksi perubahan radiasi inframerah yang biasanya terpancar oleh objek yang bergerak. Ketika perubahan terdeteksi,

maka sinyal digital *HIGH* akan terkirim ke *ESP32*, yang kemudian mengeksekusi command untuk menggerakan semua *servo* secara serentak ke posisi yang sudah di deteksi sebelumnya.

4. Jelaskan alasan, fungsi, dan arah tuju koneksi setiap pin *ESP32* yang dimanfaatkan dari skematik (poin 5 Informasi Pengerjaan) yang telah anda buat!
   - Seluruh komponen ground dihubungkan ke pin *GND ESP32*, untuk *initialize* tegangan awal.
   - *Sensor MPU6050*, menggunakan protocol komunikasi *I2C (Inter-Integrated Circuit).* Jadi hanya menggunakan 2 jalur untuk dapat menghubungkan banyak perangkat. Arah tuju *SDA* ke *GPIO 22* berfungsi sebagai jalur transfer data bolak-balik antaran *MPU6050* dan *ESP32* dalam mengirimkan data *roll, pitch, yaw.* Pin *GPIO 21* berfungsi sebagai *SCL* atau *Serial Clock Line* yang bertugas memberikan sinyal clock untuk menyinkronkan proses komunikasi data. Lalu, karena hanya membutuhkan tegangan operasi 3,3V, maka sensor dihubungkan dengan pin *3V3*.
   - *Sensor PIR* terhubung dengan *GPIO 33* sebagai input digital, untuk mendeteksi sinyal *HIGH LOW* dari deteksi inframerah. Jika sinyal *HIGH* terdeteksi, *ESP32* akan mengeksekusi command untuk menggerakan seluruh *servo*.
   - *Servo* motor yang jumlahnya 5, pin *PWM* masing – masing terhubung ke: *Servo* 1 ke *GPIO 13, Servo* 2 ke *GPIO 12*, *Servo* 3 ke *GPIO 14*, *Servo* 5 ke *GPIO 26*. Sedangkan untuk *V+* dari semua *servo*, dihubungkan ke esp: *VIN* untuk memberinya tegangan 5V.
   - Sebagai tambahan, ada *LED* yang dihubungkan pada *GPIO 2* sebagai indikator status sistem yang sedang berjalan.

5. 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!
   - Jadi sistem ini bekerja dengan membaca keadaan lingkungannya. Sebagai unit pemroses utama, kita gunakan *ESP32*, ini gunanya untuk mengkoordinasi seluruh sistem dengan menghubungkan seluruh komponen sensor dan *servo*. Nah, pertama-tama kita perlu beritahu dulu ke *ESP* pin dan komponen apa saja yang kita gunakan dengan fungsi define. Setelah itu kita buat object untuk kelima *servo* dan *MPU*nya.
   - Tahap kedua, kita siapin dulu sistemnya. Langkahnya adalah, kamu buka dulu komunikasi dengan komputernya agar bisa nampilin laporan. Lalu, dia bakal nyiapin koneksi sama *sensor MPU 5060* melalui kabel yang Namanya *SDA* dan *SCL*. Barulah pin untuk s*ensor PIR* dan lampu indikator *LED* diatur. Terakhir, atur semua *servo* agar mereka mulai di posisi 90 derajat.
   - Setelah disiapin, sekarang sistem ini bisa bekerja non-stop. Gimana caranya? Pakai fungsi loop. *Sensor MPU* bakal ngasih tau sistem posisinya sekarang kayak gimana. Apakah ada pergerakan ke kanan-kiri, depan-belakang, ataukah malah berputar. Misal ternyata ada gerakan yang terdeteksi, *ESP32* bakal nyuruh semua *servo* buat kembali lagi ke posisi awalnya yaitu 90 derajat.

— Jika ternyata sensor *PIR* tidak mengirimkan data apapun ke *ESP32*, kita bakalan gantian baca data dari sensor satunya yaitu *MPU6050*. Ada 3 kondisi yang berbeda dan kondisi tersebut akan menentukan aksi dari program kedepannya. Pertama, kalua sistemmya miring kanan-kiri, *servo* 1 dan 2 bakal berusaha nyeimbangin posisi awalnya hingga stabil. Kedua, kalau sistemnnya gerak kedepan atau belakang, *servo* 3 dan 4 bakalan gerak berlawanan arah sehingga mereka balik lagi ke kondisi awal yaitu 90 derajat. Terakir, kalo ternyata sistemnya lompat keatas atau malah turun *ESP32* bakal memerintah *servo* 5 untuk mengikuti arah gerakan sebelum Kembali ke posisi awal.
— Kita kasih batasan buat *servo*nya biar ga terlalu banyak gerakan, disini kita kasih limit 0 sampai 180 dengan 90 sebagai titik tengah. *ESP32* bakalan ngelaporin semua status dari sensor dan ditampilin.