

1. FSM (Sonlu Durum Makinesi) Yapısı

Robotun otonom davranışı, 4 ana duruma sahip gelişmiş bir Sonlu Durum Makinesi (FSM) ile kontrol edilmiştir. Bu yapı, robotun önce duvarı bulmasını, ardından duvarı sağına alarak PID kontrolcüsü ile pürüzsüz bir şekilde takip etmesini sağlar.

Durum Diyagramı Şeması:

- BAŞLANGIÇ_TARAMA (Initial Scan):** Robot çalıştırıldığında önünü kontrol eder. Eğer önünde hemen bir engel varsa ($Mesafe < 0.5m$), olduğu yerde sola dönerek yönünü açar. Önü boşsa DUVAR_BUL durumuna geçer.
- DUVAR_BUL (Find Wall):** Robot, önündeki engel mesafesi 0.5 metreye düşene kadar düz ilerler. Duvara yaklaştığında durur ve SOLA_DON durumuna geçer.
- SOLA_DÖN (Turn Left):** Robot, duvarı sağına almak için sola döner. Önü açılana ($>0.6m$) VE sağ sensörler duvarı görene ($<1.0m$) kadar dönüş devam eder. Şartlar sağlandığında DUVARI_TAKIP_ET başlar.
- DUVARI_TAKIP_ET (PID Follow):** Robot, sağ tarafındaki duvara olan mesafeyi referans alarak ilerler.
 - PID Kontrol:** $Hata = Hedef\ Mesafe (0.6m) - Mevcut\ Mesafe$ formülüyle robotun açısal hızı dinamik olarak hesaplanır.
 - Güvenlik:** Eğer takip sırasında önüne aniden bir engel çıkarsa, PID devre dışı kalır ve robot çarpışmayı önlemek için sola manevra yapar.

2. ROS Node Kaynak Kodu

Aşağıda, /scan konusunu dinleyen, Lidar verilerini filtreleyen ve PID algoritması ile /mobile_base/commands/velocity konusuna hassas hız komutları gönderen Python kodu yer almaktadır.

Dosya Adı: scripts/task_1_obstacle.py

3. Görev Raporu

3.1. Kullanılan Topic'ler (Konular)

Proje kapsamında kullanılan topic'ler, Kobuki robot platformuna (Turtlebot 2) özel olarak seçilmiştir:

- /scan (sensor_msgs/LaserScan):**
 - RPLidar sensöründen gelen ham mesafe verileri okunmuştur.

- Veriler ham halde kullanılmamış, get_sector_min fonksiyonu ile dilimlere (Ön ve Sağ sektör) ayrılarak işlenmiştir.

2. /mobile_base/commands/velocity (geometry_msgs/Twist):

- Standart /cmd_vel yerine, Kobuki base sürücüsünün (nodelet) doğrudan dinlediği bu topic kullanılmıştır. Bu seçim, hız komutlarının multiplexer (mux) tarafından engellenmeden doğrudan motora iletilmesini sağlamıştır.

3. ~state (std_msgs/String):

- FSM'in o an hangi durumda olduğunu (örn: 'sola_don', 'duvari_takip_et') dışarıdan gözlemlemek ve debug yapmak için eklenmiştir.

3.2. Eşik Değerleri ve Seçim Nedenleri

| Parametre | Değer | Seçim Nedeni |
|--------------------------------|----------------|--|
| Hedef Duvar Mesafesi | 0.60 Metre | Robotun duvara çok yaklaşmadan (çarpışma riski) ve çok uzaklaşmadan (sensör menzili) stabil bir koridorda ilerlemesi için ideal mesafe olarak belirlendi. |
| Çarpışma Mesafesi | 0.50 Metre | PID kontrolü sırasında robotun önünde ani bir engel belirirse (duvar köşesi vb.), PID'yi iptal edip kaçış manevrası yapmak için güvenlik sınırı. |
| PID Katsayıları (Kp, Kd) | Kp=1.5, Kd=5.0 | Kp, hatalara hızlı tepki vermek için; Kd ise robotun duvara yaklaşırken yavaşlamasını ve salınım (oscillation) yapmasını engellemek için deneysel olarak ayarlandı. Yüksek Kd değeri ani hareketleri söndürdü. |
| Robot Kör Nokta (Blind Radius) | 0.20 Metre | Lidar sensörünün, robotun kendi şasesindeki metal çıtalara veya kabloları engel sanmasını önlemek için 20cm altındaki veriler filtrelemiştir. |

3.3. Karşılaşılan Problemler ve Çözümleri

Problemle karşılaşılmamıştır