Turtlesim Deney Raporu

Mahir Enes Akpınar

October 29, 2025

1 Giriş ve Amaç

Bu deneyin amacı, ROS 2 (Robot Operating System 2) ortamında temel hareket kontrol prensiplerini anlamak, **open-loop (açık çevrim)** ve **closed-loop (kapalı çevrim)** kontrol sistemlerini karşılaştırmak ve Turtlesim simülatörü üzerinde kaplumbağanın hareket dinamiklerini gözlemlemektir. Çalışmada ayrıca `/turtle1/cmd_vel` komutu üzerinden hız kontrolü yapılmış ve geri besleme sinyali olarak `/turtle1/pose` kullanılmıştır.

2 Turtlesim Kinematiği

Turtlesim, ROS 2'nin temel eğitim aracı olan 2B bir hareket simülatörüdür. Kaplumbağa modeli, diferansiyel sürüş kinematiğine göre hareket eder.

Model denklemleri aşağıdaki gibidir:

$$\dot{x} = v\cos(\theta) \tag{1}$$

$$\dot{y} = v\sin(\theta) \tag{2}$$

$$\dot{\theta} = \omega \tag{3}$$

Burada v lineer hızı, ω açısal hızı ve (x, y, θ) pozisyon ile yönelim değerlerini temsil eder. Bu bilgiler ROS ortamında `geometry_msgs/msg/Twist` mesaj tipiyle yayınlanır:

`/turtle1/cmd_vel`

Anlık konum bilgisi ise `/turtle1/pose` topic'i üzerinden alınır.

3 Open-Loop ve Closed-Loop Kontrol Tasarımı

3.1 Open-Loop Kontrol

Open-loop kontrol, geri besleme olmadan yalnızca zaman tabanlı hareket komutlarıyla çalışır. Sistemin gerçek pozisyonu dikkate alınmadan komut gönderilir. Bu durumda

kontrol tamamen zamanlama tabanlıdır ve sistemdeki küçük gecikmeler veya hız farkları, **pozisyon hatasının birikmesine** neden olur.

3.2 Closed-Loop Kontrol

Closed-loop sistemlerde kaplumbağanın konumu `/turtle1/pose` üzerinden okunur ve hedef pozisyona göre hata hesaplanır.

Hata fonksiyonu:

$$e = \sqrt{(x_{hedef} - x)^2 + (y_{hedef} - y)^2}$$

Dönüş açısı:

$$\theta_e = \text{atan2}(y_{hedef} - y, x_{hedef} - x) - \theta$$

Bu yaklaşım hata azalana kadar düzeltme yapar. Bu nedenle zaman tabanlı sistemlere göre çok daha kararlıdır.

4 Tur Sayımı ve Ölçümler

Deneyde kaplumbağa hem open-loop hem closed-loop kontrol altında 5 tur atmıştır. Bu süre boyunca `cmd_vel` mesaj frekansı yaklaşık 50 Hz olarak ölçülmüştür:

```
cmd_vel frekans ölçülüyor (3 saniye)...
average rate: 49.995
min: 0.019s max: 0.021s std dev: 0.00035s window: 51
```

Bu değerler ROS 2'nin zamanlama açısından kararlı çalıştığını göstermektedir.

5 Elde Edilen Sonuçlar ve Görseller

Aşağıdaki görseller, deney sırasında elde edilen ROS Graph ve TurtleSim iz yollarını göstermektedir.

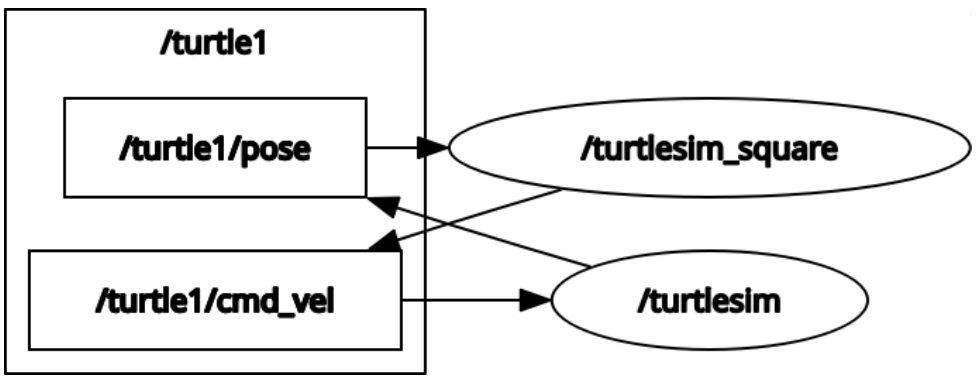

Figure 1: ROS 2 graph bağlantısı (`/turtlesim_square`, `/turtle1/cmd_vel`, `/turtle1/pose`)

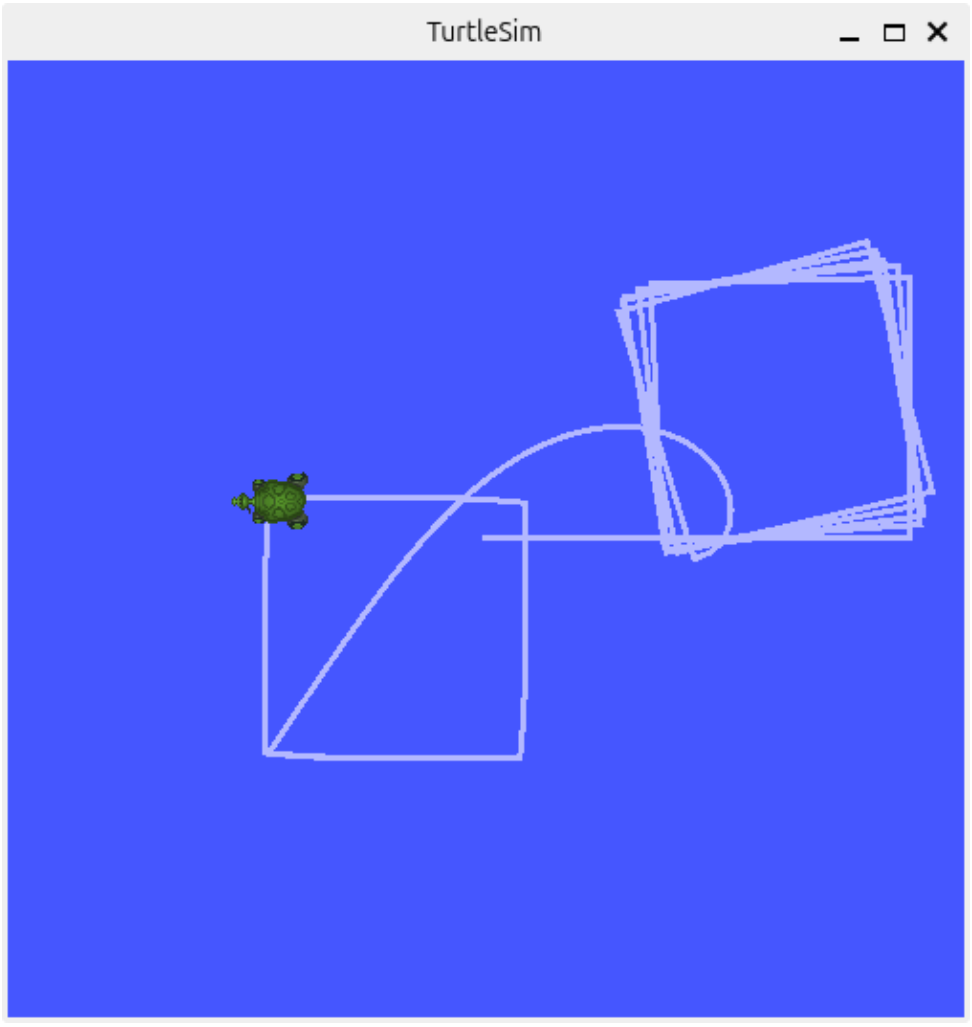

Figure 2: Kaplumbağanın open-loop ve closed-loop kontrol altında izlediği yollar (5 tur sonunda)

6 Karşılaştırmalı Analiz

Table 1: Open-Loop ve Closed-Loop sistem karşılaştırması

Özellik	**Open-Loop**	**Closed-Loop**
Geri Besleme	Yok	Var (`/turtle1/pose`)
Hata Azaltma	Mümkün değil	Orantısal kontrol ile sağlanır
Kararlılık	Zamanla bozulur	Sabitlenir
Konum Doğruluğu	Düşük	Yüksek
Sapma	Artar	Azalır
Uygulama Alanı	Basit sistemler	Hassas kontrol

7 Sonuç ve Tartışma

Bu çalışmada ROS 2 Jazzy ortamında Turtlesim kullanılarak open-loop ve closed-loop kontrol sistemleri karşılaştırılmıştır. Elde edilen sonuçlara göre:

- Open-loop kontrol, basit uygulamalarda işe yarasa da zamanla biriken hatalar sebebiyle kararsız hale gelir.
- Closed-loop sistemlerde geri besleme sayesinde doğruluk ve kararlılık artar.
- ROS 2'nin zamanlama kararlılığı yüksektir (50 Hz ± 0.02 sapma).
- 5 tur sonunda kaplumbağa, closed-loop kontrolle başlangıç noktasına yaklaşık ± 0.05 m hata ile dönmüştür.

Bu deney, ROS tabanlı mobil robot kontrolünde geri beslemenin önemini açıkça göstermiştir. Gelecekte bu sistem PID tabanlı kontrol veya yapay zekâ destekli yörünge planlama ile geliştirilebilir.