



COBOTTHON

Robot Kontrol Kategorisi *Yarışma Kılavuzu*



Ekim 2023

İçindekiler

Giriş.....	3
Gereksinimler.....	4
Yarışma Simülasyon Ortamı.....	5
Yarışma Materyalleri.....	6
Yarışma Konfigürasyonu.....	8
Yarışma Senaryosu.....	9
Yarışma Süreci.....	15
Başarı Kriterleri.....	16

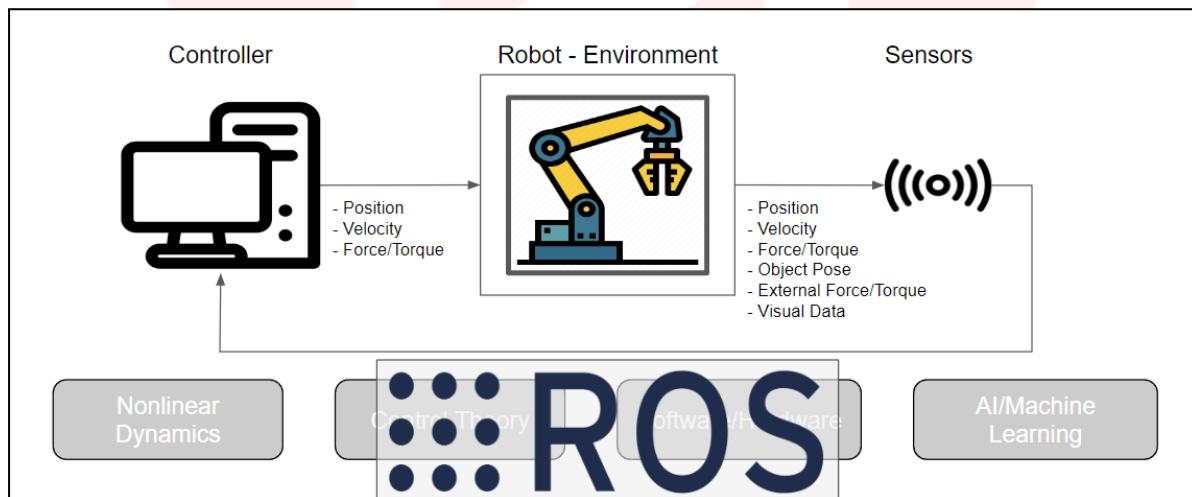


Giriş

Günümüzde teknolojinin gelişmesi ve endüstri 4.0'ın entegrasyonu ile üretim hatlarından hizmet sektörüne kadar bir çok sektörde insanların yaptıkları işi yapabilecek akıllı ve programlanabilir makineler, yani robotlar, kullanılmaya başlanmıştır. Her ne kadar endüstri 3.0 ile birlikte robotlar, özellikle otomotiv endüstrisinde uzun bir süredir kullanılmakta olsa da, bu kullanım belirli operasyonlarla sınırlı kalmış ve insan ile robotun çalışma alanlarının birbirinden fiziksel bariyerlerle ayrılması üretimin verimliliğini ve esnekliğini düşürmüştür. Özellikle KOBİ statüsündeki fason üretim yapan atölyeler ve fabrikalarda, insanlara yardımcı olması amacıyla tasarılan işbirlikçi robotlar, akıllı üretim alanında son zamanların en hızlı artan trendi haline gelmiştir ve dünyanın farklı yerlerinden bir çok mühendis, bu trendi yakalamak için çalışmalara başlamıştır.

ROS (Robotik İşletim Sistemi), robotik program geliştirme ve üst seviye robot kontrolü için büyük bir önem taşımaktadır. Özellikle işbirlikçi robot kolların üzerinde yeni nesil kontrol algoritmalarının geliştirilmesi konusunda ROS ekosistemi, AR-GE yapan bir çok firma ve akademisyen tarafından kullanılmaktadır.. ROS ekosisteminin sunduğu kolaylıklar özellikle robotların simülasyon ortamında gerçek zamanlı fizik simülatörleri üzerinde test edilmesine olanak sağlar ve böylece algoritma ve kod geliştirme aşamasında maliyet ve zaman tasarrufu yaratır.. MCFLY Robot Teknolojileri olarak; verimli ve temiz robotik programların geliştirilmesi önemsenmektedir.

Cobothon yarışmamızda, yarışmacılara hazır bir simülasyon ortamı verilecek ve yarışmacılardan ortamdaki robot kolu belirli bir otomasyon görevini yerine getirmesi istenecektir. Buna ek olarak son bölümde yarışmacılara bir araştırma konusu verilecek ve ilgili sunumlarını gerçekleştirecektir.



Gereksinimler

Yarışmacıların önceden kişisel bilgisayarlarına ilgili kurulumları yapması gerekmektedir. Bu kurulumlar aşağıda verilmiştir.

- Ubuntu 20.04 Focal
- ROS 1 Noetic Desktop Full

Ayrıca simulasyon için gereken ROS Noetic paketleri aşağıda verilmiştir.

```
sudo apt install ros-noetic-gazebo-ros*  
sudo apt install ros-noetic-gazebo-dev  
sudo apt install ros-noetic-gazebo-plugin  
sudo apt install ros-noetic-gazebo-msgs
```

```
sudo apt install ros-noetic-robot-controllers*  
sudo apt install ros-noetic-robot-state*  
sudo apt install ros-noetic-joint*  
sudo apt install ros-noetic-control*  
sudo apt install ros-noetic-trajectory*  
sudo apt install ros-noetic-transmission*
```

Yarışma Simülasyon Ortamı

Cobothon yarışması için Ubuntu 20.04 Focal Linux işletim sistemi üzerinde ROS 1 Noetic Desktop Full sürümü kullanılarak Gazebo ortamında fabrika simülasyonu hazırlanmıştır. Simülasyon çevresi endüstriyel bir fabrikaya benzetilmiştir. Fabrika içerisinde ürün kutuları, boşaltma palet alanları, konveyörler, robot standı ve Mcfly Orion 5 işbirlikçi robot kol mevcuttur.



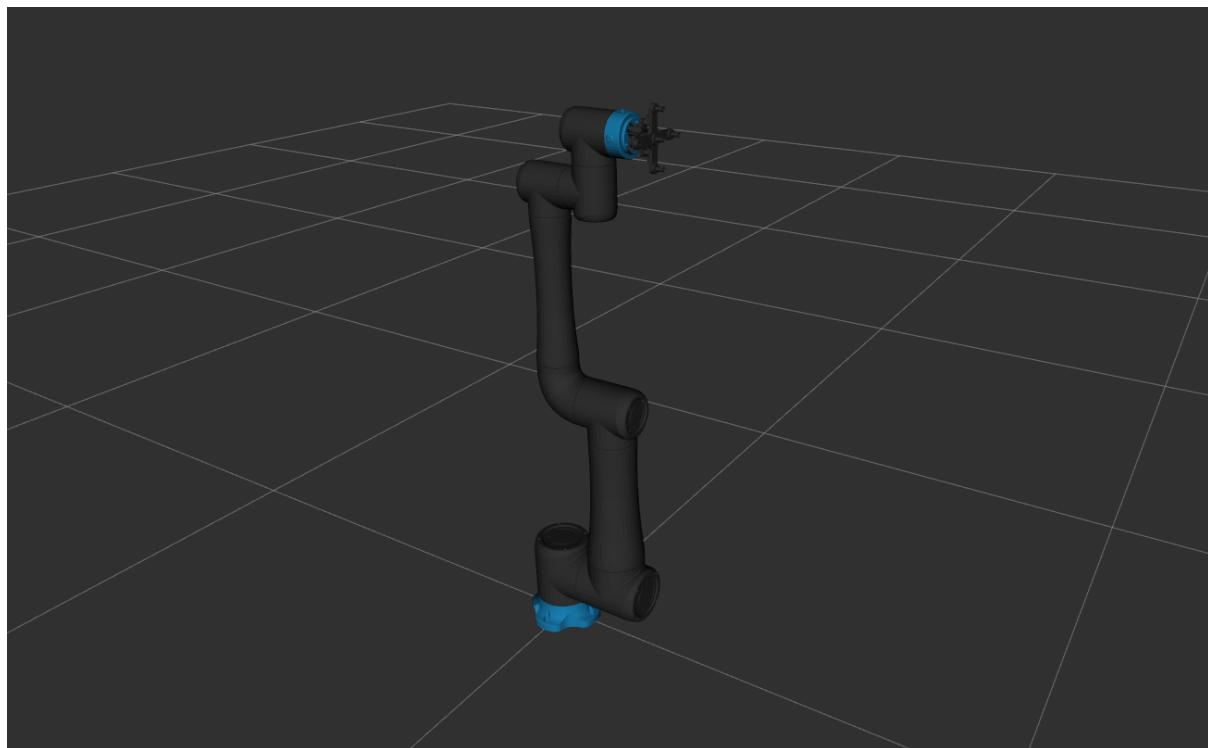
Şekil 1. Gazebo fabrika ortamı

İlgili Gazebo ortamı ve senaryolar endüstride kullanılan gerçek bir uygulamadan esinlenerek hazırlanmıştır. Uygulamalarda robot kol kullanılarak hedef kutuların taşınması, paletlenmesi ve dizilmesi gerçekleştirilecektir.

Yarışma Materyalleri

Yarışmanın kritik nesnesi Mcfy Orion 5 işbirlikçi robot koldur. Bu robotun eksen takımlarının (TF) RViz aracı kullanılarak incelenmesi için aşağıdaki komut kullanılabilir.

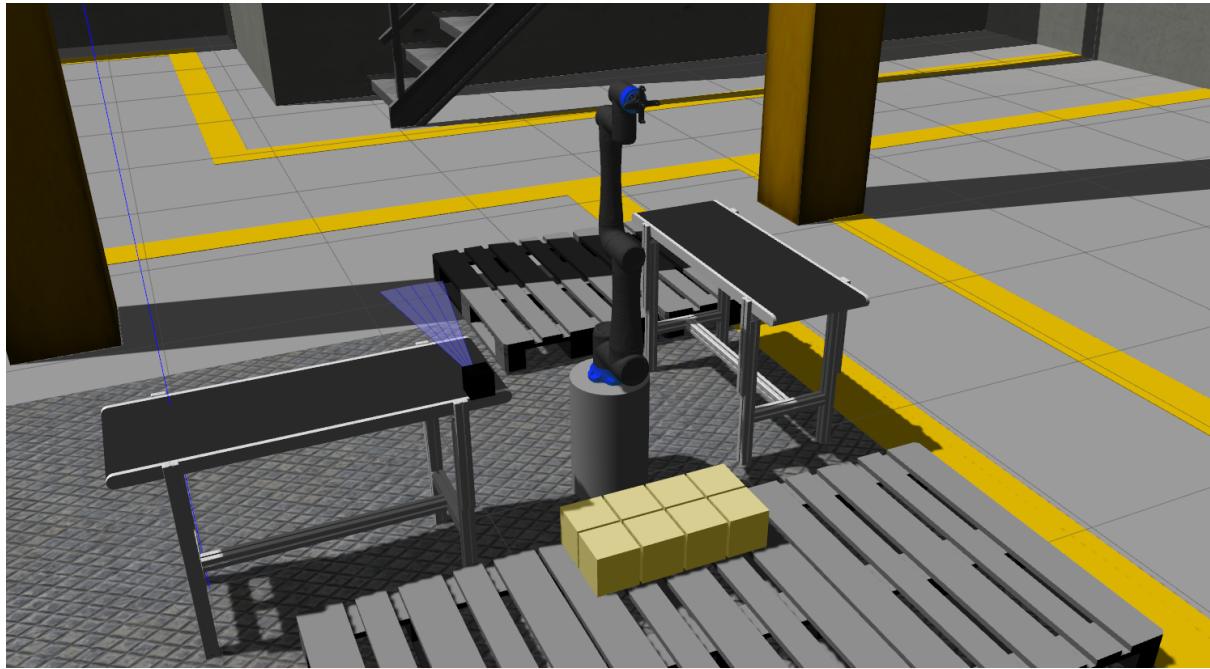
`roslaunch robot_description display_robot.launch`



Şekil 2. Rviz robot görünümü

Yarışma senaryosu bölümünde belirtilen launch komutları çalıştırıldıkten sonra Gazebo ortamında paletler, konveyörler, robot kol ve kutular görülmektedir. **Robotun base frame eksen takımının merkezi Gazebo World eksen takımına göre X'de -1.5, Y'de 0 ve Z'de 0.5 metre konumunda olmakla beraber roll, pitch ve yaw değerleri 0 ile başlatılmaktadır.**

Robot otomatik olarak ortama home pozisyonunda getirilmektedir. **Home pozisyonu eklem pozisyonu şeklinde [0, 0, 0, 0, 0, 0] olarak tanımlanmıştır.**



Şekil 3. Bölüm 1 metaryelleri

Sarı kutuların her biri özdeştir. **Toplamda 8 adet olan bu kutular $0.15m \times 0.15m \times 0.15m$ boyutunda küp geometriye sahip** ve her birinin kütlesi 3 kg'dır.

Benzer şekilde 2. bölümde de tüm metaryeller aynı olmakla beraber sadece sarı kutular bulunmamaktadır.

Yarışmaya eklenen konveyörler sadece model olarak eklenmiştir. Yarışmacıların konveyörleri ilgili pluginler yardımıyla gerçek bir şekilde çalıştırması gerekmektedir. Takımlar arzu ederse bu konveyörleri değiştirerek **konumları ve yönelimleri aynı kalacak şekilde** farklı bir konveyör kullanabilirler.

Yarışmada kullanılacak konveyörlerin kullanımı mutlaka ROS servisleri ile gerçekleştirilmelidir. Aşağıda konveyörlerin çalışması için iki farklı ROS servisi CLI komut örneği verilmiştir.

```
rosservice call /conveyor1_service "enable: true"
rosservice call /conveyor2_service "power: 25.0"
```

Yarışma Konfigürasyonu

Yarışma parametrelerinin tamamı factory_simulation paketinde config adlı klasörün içinde **box_models_links.yaml** adlı dosyanın içerisinde kaydedilmiştir. Yarışmacıların yarışma senaryolarında belirtilen görevleri bu parametreleri kullanarak yerine getirmesi gerekmektedir. Bu parametreler aşağıda verilmiştir.

box_links:

```
# - [model_name, link_name, x_center_pos, y_center_pos, z_center_pos]
- ["unit_box1", "link", -1.2, 0.5, 0.218]
- ["unit_box2", "link", -1.36, 0.5, 0.218]
- ["unit_box3", "link", -1.52, 0.5, 0.218]
- ["unit_box4", "link", -1.68, 0.5, 0.218]
- ["unit_box5", "link", -1.2, 0.66, 0.218]
- ["unit_box6", "link", -1.36, 0.66, 0.218]
- ["unit_box7", "link", -1.52, 0.66, 0.218]
- ["unit_box8", "link", -1.68, 0.66, 0.218]
```

conveyor_2:

```
spawn_count: 99
spawn_interval: 22
spawn_box_location: [0, 0.13, 0.9] # x, y, z coordinates
```

conveyor_3:

```
spawn_count: 99
spawn_interval: 18
spawn_box_location: [0, -0.36, 0.9]
```

chapter_2:

```
first_box_location: [-1.36, 0.5, 0.218]
box_unit_x_columns: 3
box_unit_y_columns: 3
box_unit_height: 3
```

box_links başlığı altındaki tüm parametreler bir liste içinde verilmiştir. Bu listede 1. bölümde kullanılacak her bir kutunun model ismi, link ismi, Gazebo ortamındaki merkez noktalarının x, y ve z lokasyonları verilmiştir.

Diğer parametreler 2. bölümde kullanılacak konfigürasyonları içermektedir. Bu parametreler yarışma senaryosu bölümünde anlatılacaktır.

Yarışma Senaryosu

Robot kontrol kategorisi 3 bölümden oluşmaktadır.

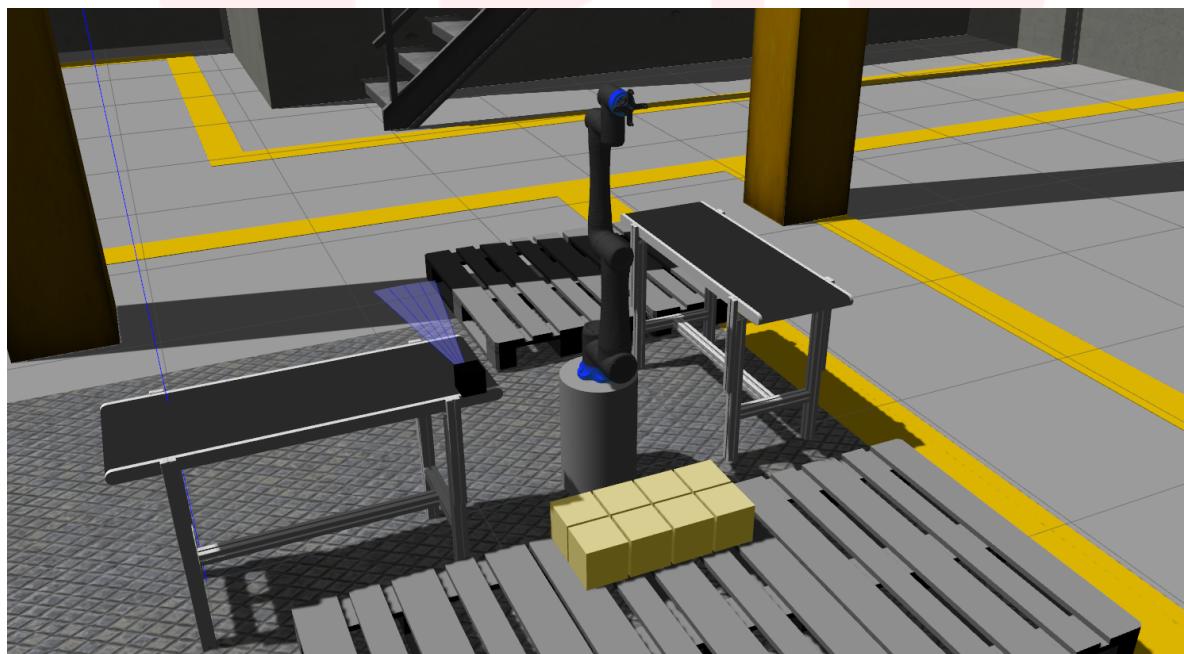
a) Birinci Bölüm

İlk bölümün başlatılması için aşağıdaki komut çalıştırılmalıdır.

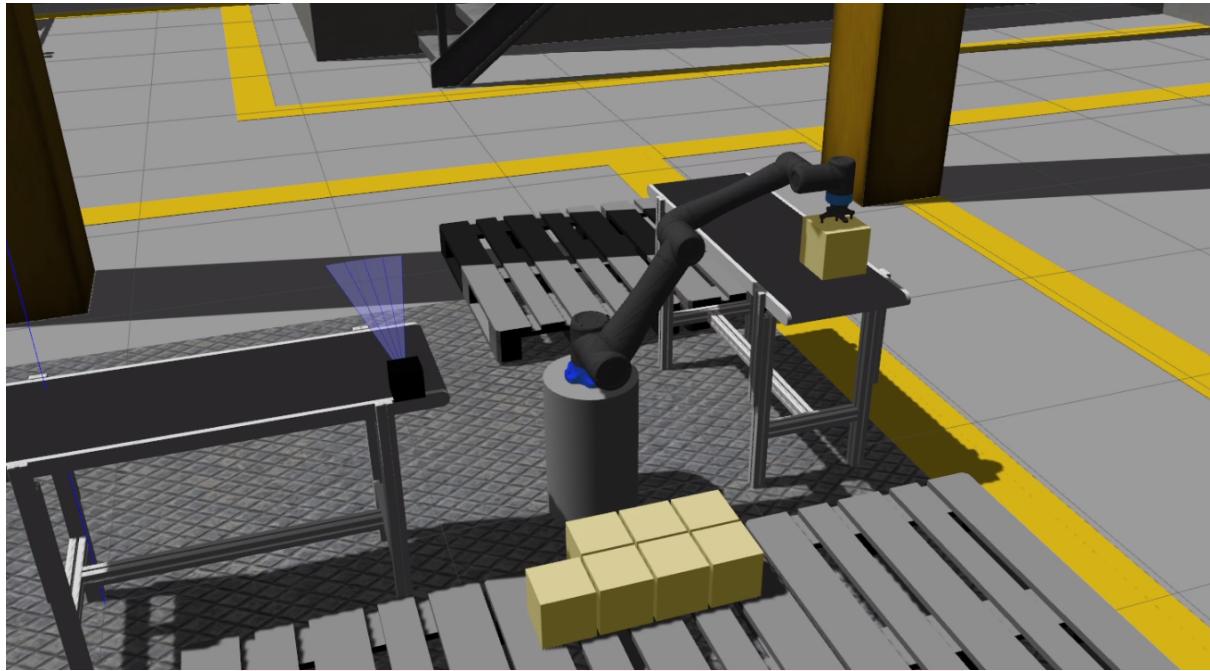
```
roslaunch factory_simulation sim_chapter1.launch  
roslaunch factory_simulation spawn_robot.launch
```

Oluşturulan ilk senaryoya göre yerdeki kutuların sırasıyla 1. konveyöre taşınması ve arka paletlere iletilmesi gerekmektedir. Robot kol şeklindeki gibi home konumundan başlayarak paletin üstündeki sarı kutuları ilgili Gazebo pluginları yardımıyla üzerindeki vakum gripper'a gelecek şekilde alması gerekmektedir. Kutular teker teker konveyörün üzerine yerleştirildikten sonra konveyör çalıştırılmalı ve arka palet tarafına gönderilmelidir. İlk kutu tamamlandıktan sonra robotun ikinci kutuyu alması ve konveyöre taşıyıp konveyörün çalıştırılması gerekmektedir. Tüm kutular bu şekilde sırasıyla gönderilmelidir. Bu sebeple robotun hareketi için gerekli ROS kontrol algoritmalarının yazılması gerekmektedir. Tüm kutular başarılı bir şekilde bitince robotun tekrar home pozisyonuna geri dönmesi beklenmektedir.

Kutular konfigürasyon dosyasında bulunan değerlere göre **sırasıyla taşınmalıdır**. Buna göre sırasıyla unit_box1, unit_box2 .. unit_box8 adlı modellerin taşınması beklenmektedir. Kutuların merkez noktaları yarışma konfigürasyonu başlığı altında verilecektir.



Şekil 4. Bölüm 1 başlangıç ortamı



Şekil 5. Bölüm 1 çalışma örneği

Robotun kontrolü uygun ROS Control yapıları kullanılarak yapılacaktır. Aynı zamanda yarışmacıların kullanabileceği ROS Control mimarisinin üzerine bina edilmiş Moveit, Tesseract vb. gibi 3. parti hazır kontrol yazılımı paketleri bulunmaktadır. Takımların bu hazır paketleri kullanmasında herhangi bir engel olmamakla birlikte **takımların kendi robot hareket kontrolörünü geliştirmesi ve görevlerde bu kontrolcüyü kullanması ek puan olarak haneye yazılacaktır**. Buna ek olarak yarışmacılar ileri ve ters kinematik hesaplamalar için arzu edilen kinematik çözücü kütüphaneleri kullanabilirler.

b) İkinci Bölüm

İkinci bölümde ise farklı bir senaryo mevcuttur. Bu bölümün başlatılması için aşağıdaki komut çalıştırılmalıdır.

```
roslaunch factory_simulation sim_chapter2.launch  
roslaunch factory_simulation spawn_robot.launch
```

Oluşturulan ikinci senaryoya göre robotun yan yana bulunan ikinci ve üçüncü konveyörlerden gelen kutularla paletleme uygulaması yapması beklenmektedir. İkinci ve üçüncü konveyörlerden farklı zaman aralıklarında kutular spawn edilecektir. Bu değişkenler yarışma konfigürasyon dosyasında mevcuttur.

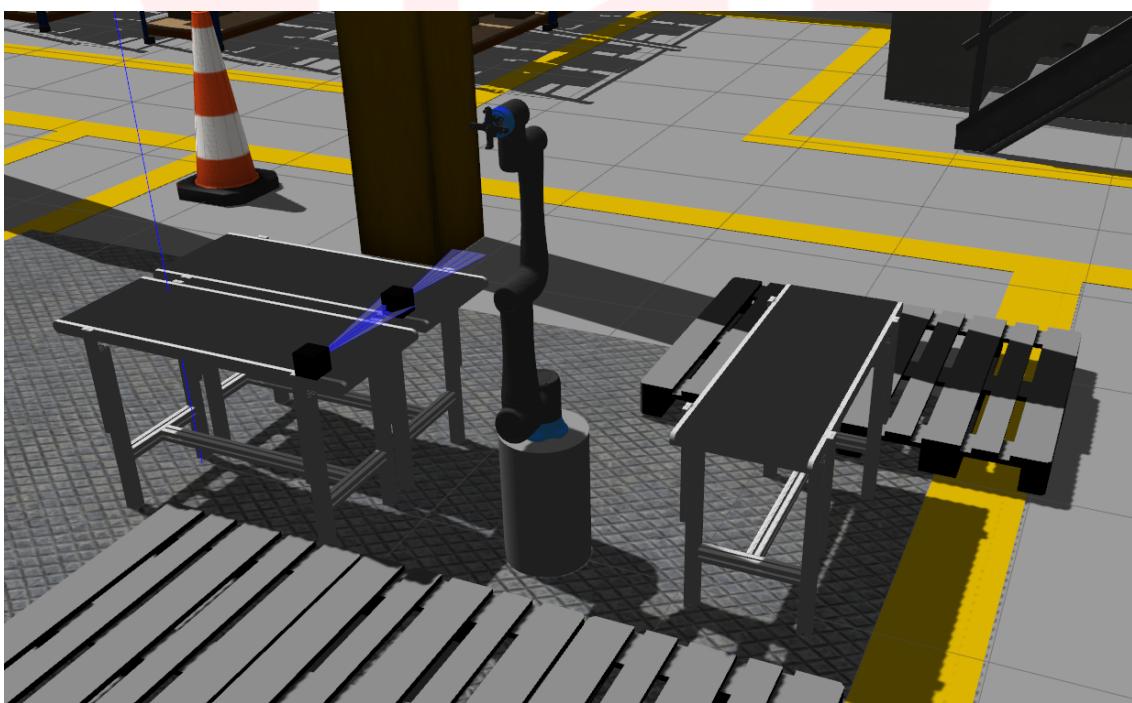
Senaryoya göre konveyörler üzerinden `spawn_box_location` konumundan, her `spawn_interval` süresi aralığında `spawn_count` adedi kadar kutu simülasyona aktarılacaktır. Bu kutular ilk bölümdeki kutularla aynı geometriye sahip ancak beyaz renktedir. İlgili kutuların bu parametrelere bağlı bir şekilde ortama aktarılması için Python kullanılarak ROS düğümü (node) yazılmıştır. Bu düğüm (`box_spawner.py`) aşağıdaki komutla çalıştırılabilir.

`rosrun factory_simulation box_spawner.py`

Yarışmacılar bu senaryoya göre robot ve konveyörlerin kontrolünü parametrik bir şekilde gerçekleştirmelidir. Öncelikle yarışmacılar ortam hazırlıklarını yaptıktan sonra kendi kontrol algoritmalarını çalıştırmalı, konveyörleri aktif hale getirmeli ve en son `box_spawner.py` adlı düğümü çalıştırmalıdır.

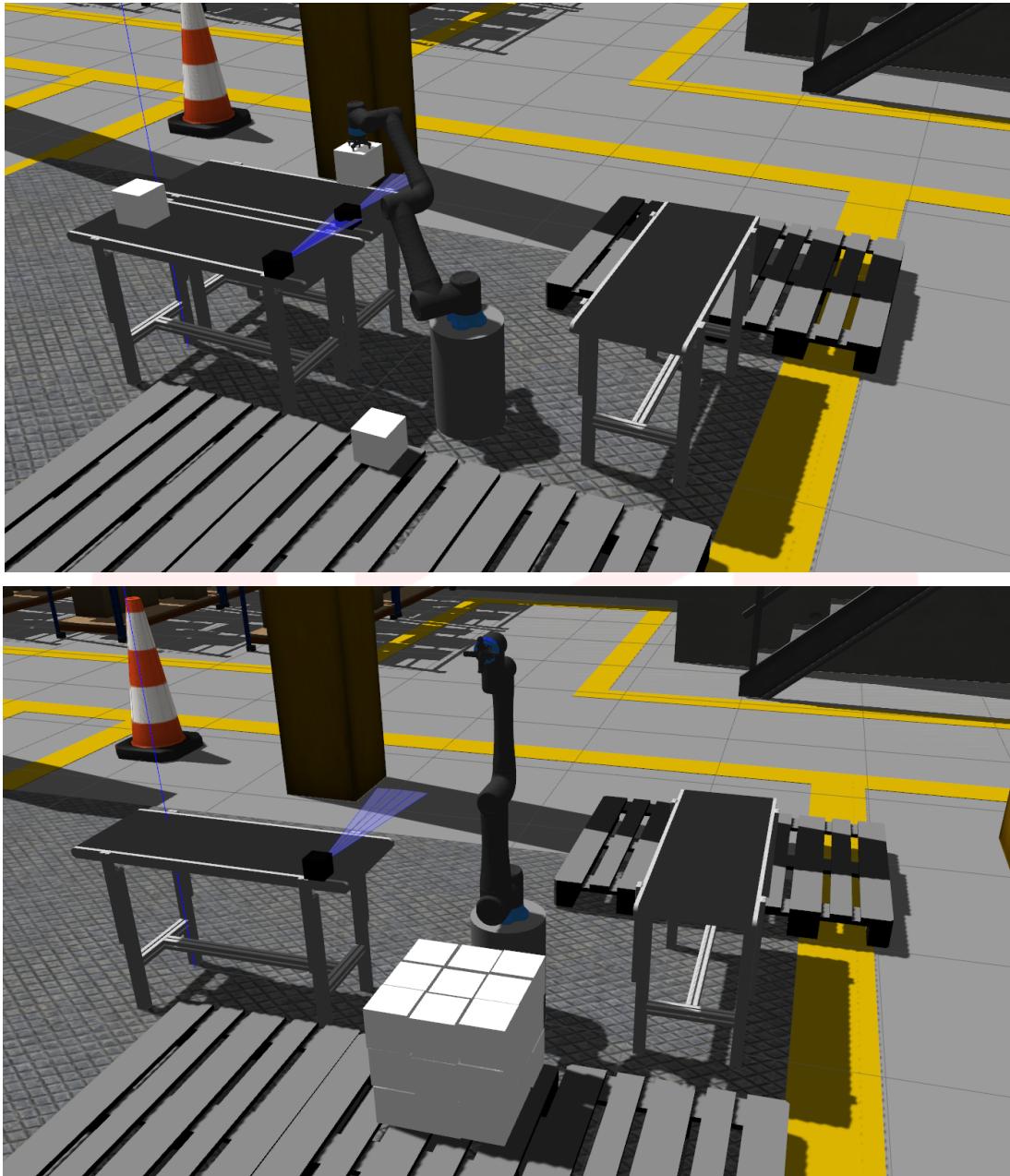
İkinci bölümde konveyörlerin kontrol edilmesi ve kutuların algılanması için sensörlere ihtiyaç vardır. **Yarışmacılardan ilgili konumlara sensörlerin eklenmesi beklenmektedir.** Böylece konveyör kontrolünde, sensörler kutu algılayana kadar ilgili konveyörün çalıştırılması ve kutu algılandığında ilgili konveyörün durdurulması gerekmektedir. Kutunun algılandığı esnada robot ilgili kutuyu almalı ve kutuyu yerdeki paletin üzerine `chapter_2/first_box_location` konumuna bırakmalıdır. Kutu alındığında konveyörler **robotla eş zamanlı olarak** çalışmalıdır ve aynı şekilde diğer kutu algılanana kadar aktif olmalıdır.

Yarışmacılar robotun, konveyörlerden gelen kutuları bir öncelik politikasına bağlı olarak almasını sağlamalıdır. Robotun en verimli bir şekilde kutuları konveyörlerden alması beklenmektedir. Konveyörlerden `spawn_count` adedi bitene kadar kutu yaratma işleminin durdurulmayacağı unutulmamalıdır (Bkz. `box_spawner.py`). Bu noktada yarışmacıların tasarlayacağı algoritmanın **overflow (taşma)** olmasına izin vermemesi beklenmektedir.



Şekil 6. Bölüm 2 ortam materyalleri

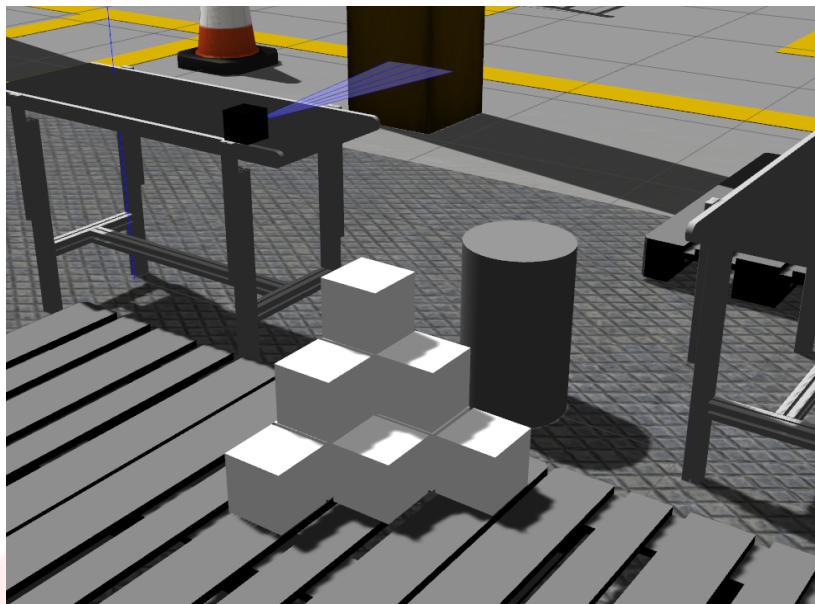
Paletleme uygulamasının parametreleri konfigürasyon dosyasındaki **box_unit_x_columns**, **box_unit_y_columns** ve **box_unit_height** adlı değerlerden alınacaktır. Bu değerler sırasıyla oluşturulacak kutu organizasyonun geometrisinin X eksenindeki birim kutu sayısı, Y eksenindeki birim kutu sayısı ve Z eksenindeki birim kutu sayısına tekabül etmektedir. Örnek olarak Şekil 6'da 3x3x3 geometriye sahip kutu organizasyonu verilmiştir. **Mutlaka kontrol algoritmasının bu değerlere göre parametrik yapılması gerekmektedir.**



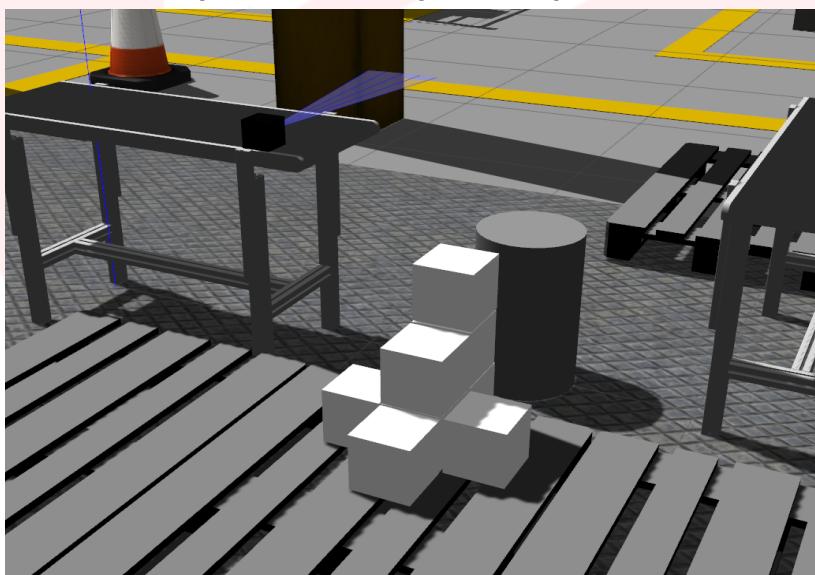
Şekil 7. Bölüm 2 palet geometri örneği

Yarışmacılar hesapladıkları kutu organizasyonuna göre toplamda **spawn_count** kadar kutuyu yarışmada istenecek olan değerlere göre sırasıyla konveyörden alıp paletin üzerine dizmesi gerekmektedir. Yerleştirilecek bitişik kutuların merkezleri arasındaki uzaklık 0.15 ile 0.2m arasında olmalıdır.

Buna ek olarak yarışmada geliştirilen algoritma yapılarının genel ve kapsamlı (generic) nitelikleri ölçülmek istenmektedir. Bu sebeple kutu organizasyonunun geometrisi ile ilgili farklı şekillerin (piramit vb.) oluşturulması istenebilir. Yarışmacıların bu duruma hazırlıklı olması gerekmektedir. İstenebilecek 2 farklı şekil örneği aşağıda verilmiştir.



Şekil 8. Bölüm 2 geometrik şekil 1

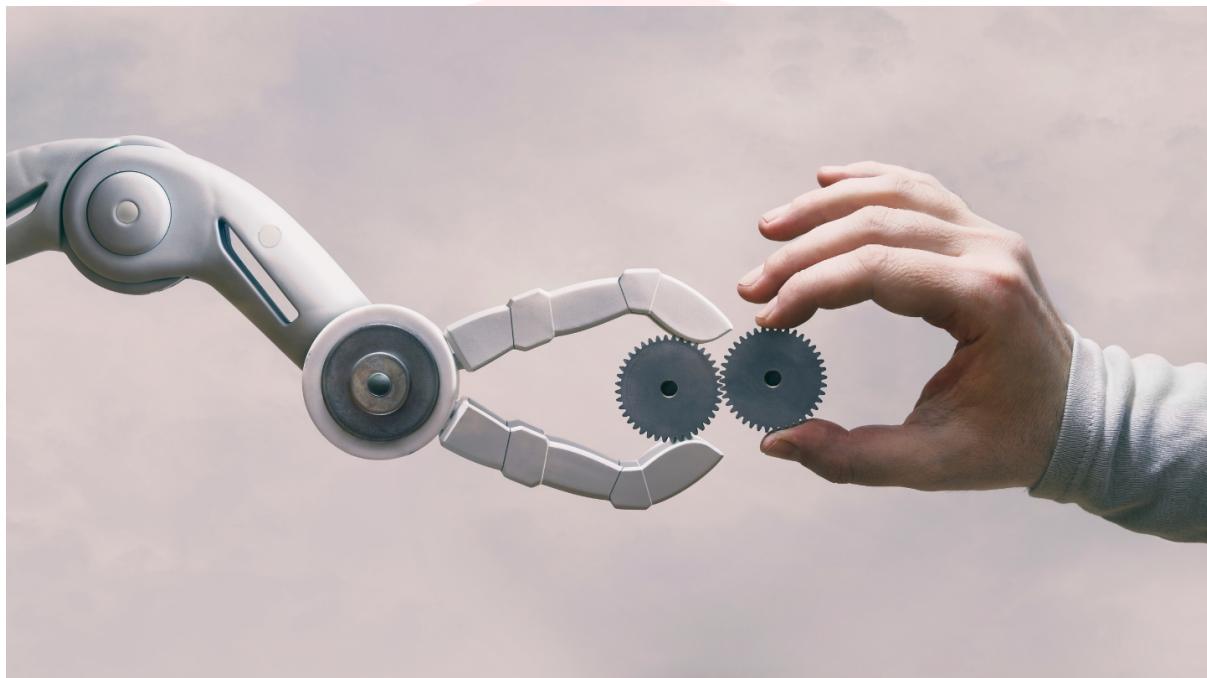


Şekil 9. Bölüm 2 geometrik şekil 2

Robotun kontrolü uygun ROS Control yapıları kullanılarak yapılacaktır. Aynı zamanda yarışmacıların kullanabileceği ROS Control mimarisinin üzerine bina edilmiş Moveit, Tesseract vb. gibi 3. parti hazır kontrol yazılımı paketleri bulunmaktadır. Takımların bu hazır paketleri kullanmasında herhangi bir engel olmamakla birlikte **takımların kendi robot hareket kontrolörünü geliştirmesi ve görevlerde bu kontrolcüyü kullanması ek puan olarak haneye yazılacaktır**. Buna ek olarak yarışmacılar ileri ve ters kinematik hesaplamalar için arzu edilen kinematik çözücü kütüphaneleri kullanabilirler.

c) Üçüncü Bölüm

Birinci ve ikinci bölümde, endüstriye yakın bir ortamda pick and place yapabilen robotik bir sistem geliştirilmiştir. Bu sistem simülasyon ortamında iyi çalışıyor olsa da insanların da çalıştığı gerçek bir fabrika veya atölyede sistemin endüstriyel olarak devreye alınması ve verimlilik, güvenlik ve insanla çalışabilmesi için işbirlikçi özelliklerinin de entegre edilmesi gerekmektedir. Üçüncü bölümde, yarışmacıların birinci ve ikinci bölümde hazırladıkları robotik otomasyon sisteminin daha verimli, güvenli ve insanla çalışmaya uygun olması için algoritmalar ve yöntemlerin araştırılması ve maksimum 10 dakikalık bir sunum hazırlanması beklenmektedir. Araştırmalar, işbirlikçi robotların dünya çapında endüstriyel uygulamaları ve yerli/yabancı akademik kaynakları kapsamlı ve sunum akademik bir dilde, gerekli referanslar gösterilerek yapılmalıdır.



Şekil 10. İnsan robot etkileşimi (temsili)

Araştırmalara yön verilmesi için aşağıdaki anahtar kelimeler/konu başlıkları kullanılabilir ancak bunlarla sınırlı olmak zorunda değildir:

- Hareket Planlama (motion planning)
- Çarpışmadan kaçınma (Collision Avoidance)
- Robot Kontrol Yöntemleri (PID, Computed Torque Control, Impedance Control)
- İnsan – Robot İşbirliği (Human – Robot Collaboration)

Son olarak bu bölümün ek puanı implementasyondur. Takımlar hazırladıkları sunumun kritik bir noktasını bir senaryo tasarlayarak Gazebo ortamında küçük bir demo ile gerçekleyebilirler. **Implementasyonun yapılması takımların hanesine ek puan olarak yazılacaktır.**

Yarışma Süreci

Yarışma esnasında ilk olarak yarışmacılara bir **Public Github adresi** verilecektir. Takımlar ROS paketlerini ve simülasyon dosyalarını buradan edinecektir. Ardından takımlar kontrol algoritmalarını geliştirmeye başlayacaklardır. Yarışmada internet ve kod geliştirmeye yardımcı olan tüm sitelerin kullanımı serbesttir.

Takımlardan geliştirdikleri kodların teslim alınması için takımların kişisel Github adreslerine ihtiyaç vardır. Takımlar kendi çalışmalarını şahsi hesaplarına **private repository** olarak yükleyeceklerdir. Ardından yarışmada belirtilecek olan hesaplara erişim verilmesi istenecektir. Takımlardan geliştirilen kodların program takviminde belirtilen her bir freework süreci sonunda repo'lara yüklenmesi (**commit push**) istenmektedir. Bu ilerlemenin izlenebilirliği açısından önem taşımaktadır.

Kod geliştirme esnasında sadece simülasyon ortamı ve puanlama ile ilgili sorular sorulabilmekte ancak kod geliştirme ile ilgili herhangi bir soruya cevap verilmeyeciktir.

Kodların geliştirme süreci tamamlandıktan sonra mentörlerin isteği doğrultusunda farklı parametre değerleriyle yarışmacılardan geliştirilen algoritmaların çalıştırılması ve gösterilmesi istenecektir.

Başarı Kriterleri

Puanlandırma noktasında birinci bölümün amacı ROS ve Kontrol becerisinin ölçülmesi iken ikinci bölümün amacı robotik ve algoritma yeteneğinin ölçülmesidir. Üçüncü bölümde ise araştırma ve geliştirme (AR-GE), yorumlama, akademik çalışma yetenekleri ön plana çıkmaktadır. **Yarışma isterlerinin geliştirilmesindeki en önemli kriter özgünlüktür.**

Her üç bölüm için de ek puan alınabilecek başarı kriterleri oluşturulmuştur. Bu kriterler bölümlerin aşılması kritik noktalarını oluşturmaktadır.

Bölüm 1:

- ROS Control konfigürasyonu ve robotun hareket ettirilmesi,
- Görevin güvenli bir şekilde maksimum 2 dk 30 sn içinde tamamlanması,
- ROS çalışma dizininde paket organizasyonunun Readme dokümantasyonu dahil olmak üzere konvansiyonel yapıda, temiz ve okunabilir bir şekilde düzenlenmesi.
- **Ek Puan:** Özgün robot hareket kontrolcüsünün yazılması ve kullanılması

Bölüm 2:

- Kontrol düğümlerinin senkron bir şekilde haberleşmesi,
- Robotun kutuları istenen herhangi bir düzende, en yüksek verimlilikle yerlestirebilmesi.
- **Ek Puan:** Özgün robot kontrolcüsünün kullanılması

Bölüm 3:

- Verilen araştırma konusu ile ilgili hem endüstriyel, hem yerli ve hem de yabancı akademik araştırmaların yapılması ve akademik formatta sunum yapılması.
- **Ek Puan:** Araştırma implementasyonunun yapılması