6.Hafta Notu
ROS Kavramsal Temelleri:
ROS (Robot Operating System): Robot yazılımı geliştirmek için kullanılan açık kaynaklı bir altyapıdır.

ROS ÇALIŞMA ORTAMI HAZIRLAMA

ROS1 her zaman şu yapıda çalışır:

```
catkin_ws/
└── src/
└── build/
└── devel/
```

-Workspace Oluştur:

```
mkdir -p ~/catkin_ws/src
cd ~/catkin_ws
```

-Derleme:

```
catkin_make
```

(Bu sana build/ ve devel/ klasörlerini oluşturur.)

-ROS Ortamına Tanıt:

```
source devel/setup.bash
```

Bunu kalıcı yap:

```
nano ~/.bashrc
```

En alta ekle:

```
source ~/catkin_ws/devel/setup.bash
```

İlk Paketini Oluştur:

```
cd ~/catkin_ws/src
catkin_create_pkg my_robot roscpp rospy std_msgs
```

Derle:

```
cd ~/catkin_ws
catkin_make
source devel/setup.bash
```

Kontrol:

```
rospack list | grep my_robot
```

PAKETLER(PACKAGES):

Paketler, ROS'ta yazılımın düzenli olmasını sağlayan ana birimlerdir. Bir paket, işlemleri (düğümleri), ROS'a bağlı kütüphaneleri, yapılandırma dosyaları gibi paket için faydalı olabilecek dosyaları içerir. Paketlerin oluşturulma amacı, kodların parçalara bölünerek yeniden kullanılabilir olmasını sağlamaktır.
Paketlerde bulunan klasörler:
CMakeLists.txt: Derleyiciye verilen emirleri ve birçok yapı bilgisini içeren derleme dosyasıdır.
package.xml: Paketlerle alakalı bilgileri içeren bildiri dosyasıdır.
scripts/: Doğrudan çalıştırılabilir kodların bulunduğu klasördür.
src/: Programların kaynak dosyalarının bulunduğu klasördür.
msg/:Mesaj tiplerinin içerildiği klasördür.
srv/: Servis dosyalarının bulunduğu klasördür.
action/: Aksiyon dosyalarının bulunduğu klasördür.
bag/: Kayıt için gereken çanta dosyalarını barındıran klasördür.
launch/: Başlatma dosyalarını barındıran dosyadır.

PAKET ARAÇLARI:
rospack: Paketleri bulma ve paketler hakkında bilgi edinmeye yarar.
roscreate-pkg: Yeni paket oluşturmaya yarar.
catkin_create_pkg: Yeni paket oluşturmaya yarar.
rosmake: Paketleri derlemeye (compile) yarar.
rosdep: Paketlerin bağımlılıklarını (dependencies) yüklemeye yarar.
catkin_make: Paketlerin çalışma alanını derlemeye yarar.
rqt: Paket bağımlılıklarını grafik olarak görselleştiren bir eklenti bulundurur
rosed:
ROS dizinleri arasında dolaşma imkanı sağlar.
roscp: ROS dizininde bulunan bir dosyayı başka bir yere kopyalamayı sağlar.
rosls: Paketteki dosyaları listelemeye yarar.
rosed: Dosyaları düzenlemeyi sağlar.
rosd: Dizin yığındaki klasörleri listeler.
rosrun: Çalıştırılabilir (executable) dosyaların çalıştırılmasını sağlar.

DÜĞÜMLER VE ROSNODE ARAÇLARI:

Düğümler, hesaplama yapan işlemlerdir. ROS'taki düğüm yapısı kullanılarak, kodlar parçalara ayrılır ve sistem basitleştirilir.
目 Sistemde her şeyi yapan büyük bir düğüme sahip olmak yerine, yalnızca tek bir işlev sağlayan birçok düğüme sahip olmak daha verimlidir.
➤Bir düğüm, C++ için roscpp, Python için rospy gibi farklı kütüphaneler kullanılarak yazılabilir.
➤ROS düğümlerinin güçlü bir özelliği, düğümü başlatırken parametreleri (düğüm adı, konu adı, vb.) değiştirebilme özelliğidir. Değiştirme işlemi ile, kod yeniden derlenmeden düğüm

yapılandırılabilir, böylece farklı senaryolara kolayca adapte edilebilir.
➤Bir düğümün sistemde benzersiz bir adının olması gerekir.
➤ROS, nodelets adında başka bir düğüm türüne sahiptir. Bu özel düğümler, tek bir işlemde birden fazla düğümü çalıştırmak üzere tasarlanmıştır. Bu şekilde düğümler ağa aşırı yüklenmeden, daha verimli şekilde iletişim kurabilirler.

Düğüm Araçları:
rosnode info düğüm adı: Düğüm hakkındaki bilgileri yazdırır.
rosnode kill düğüm adı: Çalışan bir düğümü sonlandırır.
rosnode list: Aktif düğümleri listeler.
rosnode machine hostname: Belirli bir makinede çalışan düğümleri listeler.
rosnode ping düğüm_adı: Düğümle olan bağlantıyı test eder.
rosnode cleanup: Ulaşılamayan düğümlerin kayıt bilgilerini temizler.

not:
roscd ogretici_paket/scripts/
chmod +x servis.py
ls
chmod +x istemci.py
cd ~/catkin_ws/
catkin_make

PARAMETRE SERVİSLERİ VE ROSPARAM ARACI:

ROS'ta merkezi bir "ayar deposu" vardır.
Buna Parameter Server denir.
Robot çalışırken:

PID kazançları

Maksimum hızlar

Kamera thresholdları

Algoritma eşikleri

Dosya yolları

👉 Hepsi buraya kaydedilir ve node'lar buradan okur.

Yani robotun "ayar paneli" burasıdır.

rosparam list:Tüm parametreleri göster.
rosparam get /param_adi:Parametre oku.
rosparam set /param_adi değer:Parametre yaz.

rosparam delete /param_adi:Parametre sil.
rosparam dump file.yaml:Tüm parametreleri dosyaya kaydet.
rosparam load file.yaml:Dosyadan parametre yükle.

örnek:
rosparam get /max_speed -> Maksimum hızı oku
rosparam set /max_speed 2.0 -> Hızı değiştir

MESAJLAR VE ROSMSGS ARACI:
ROS'ta topic'lerden akan her veri bir "mesaj"tır.

Topic Mesaj
/cmd_vel -> geometry_msgs/Twist
/scan -> sensor_msgs/LaserScan
/camera/image_raw -> sensor_msgs/Image
/odom -> nav_msgs/Odometry

ROSTOPİC ARACI:

Topic	**Taşıdığı veri**
/cmd_vel	Twist (hareket)
/scan	LaserScan
/camera/image_raw	Image
/vehicle_cmd	VehicleControl

Topicleri listele:
-rostopic list

Topic'in mesaj tipini öğren:
-rostopic info /scan

Canlı veri izle:
-rostopic echo /cmd_vel

Test için veri gönder:
-rostopic pub /vehicle_cmd vehicle_msgs/VehicleControl '{throttle: 0.4, steer: -0.2, shift_gears: 2}'

servisler:

Servis = Soru–Cevap Modeli

Topic → sürekli akar
Service → “Sor, cevabı al, biter”

Örnek	Anlam
`/reset_odometry`	Konumu sıfırla
`/start_engine`	Motoru çalıştır
`/change_lane`	Şerit değiştir

Servis: anlık
Action: “git – yap – bitince haber ver”

ACTION = Uzun Süren Görevler

Görev	Action
Park et	ParkAction
Şerit değiştir	ChangeLaneAction
Eve dön	ReturnHomeAction

ACTION CLIENT NEDİR?

Action Client = Görev başlatan, izleyen ve yöneten karar yöneticisidir.

Robot kendi kendine park etmeye başlayamaz.
Bir karar katmanı onu göreve başlatır.

XML Etiketleri
Herhangi bir başlatma dosyasının dosyasının kök öğesidir.
Başlatmak istenilen bir ROS düğümünü belirtir. İçerisinde bulunan özelliklerden bazıları:
pkg="paket_adı" Düğümün bağlı olduğu paketin adını belirtir.
type="düğüm_türü" Düğümün türünü belirtir.
name="düğüm_adı" Düğüme verilecek ismi belirtir.
args="argüman 1 argüman2..." Düğümün alacağı argümanları belirtir.
required="True" Düğüm sonlandığında, tüm başlatma dosyasını sonlandırmaya yarar.
ns="isim" Düğümü belirtilen alan adı altında başlatmaya yarar.
launch-prefix="prefix argümanlar" Düğümü başlatırken kullanılabilecek bağımsız değişkenleri belirtir.
<
remap> Düğümde kullanılan varsayılan konu ismini değiştirmek için kullanılır. İşlendiği satırdan itibaren geçerlidir. İçerisinde iki özellik bulunur.

GÖRÜNTÜ İŞLEME:

OpenCV Nedir?
OpenCV, bilgisayarla görü ve makine öğrenmesi gibi alanlarda çok sayıda algoritmayı destekleyen bir araçtır.
➤Açık kaynak kodludur.
➤OpenCV, C++, Python, Java gibi çok çeşitli programlama dillerini destekler
➤Windows, Linux gibi farklı platformlarda çalışır.

Temel İşlemler
cv2.imread(arg1, arg2): İstenilen dosya yolundaki görüntü istenilen formatta yüklenir.
cv2.IMREAD_COLOR: Varsayılan seçenektir. Görüntüyü BGR formatında yükler. Bayrak değeri 1'dir.
cv2.IMREAD_UNCHANGED: Görüntüyü olduğu gibi yükler. Bayrak değeri -1'dir.
cv2.IMREAD_GRAYSCALE: Görüntüyü gri seviye görüntü olarak yükler. Bayrak değeri O'dır.
imge.shape: Görüntünün satır, sütun, kanal sayısını (renkli görüntü için) belirtir.
imge.size: Görüntüdeki toplam piksel sayısını belirtir.
imge.dtype: Görüntünün veri türünü belirtir.
cv2.imshow(arg1, arg2): Görüntü gösterilir. İlk argüman gösterilecek pencerenin başlığını, ikinci argüman gösterilecek görüntüyü belirtir.
cv2.waitKey(arg): Bu fonksiyon içerisine milisaniye cinsinden pencerenin açık kalması için gereken süreyi belirten bir parametre alır. Eğer bu parametre 0 olarak ayarlanırsa pencere kapatılana kadar açık kalır.
cv2.imwrite(arg1, arg2): İstenilen dosya yoluna görüntü kaydedilir. İlk argüman kaydedilecek dosya yolunu ve ismi, ikinci argüman kaydedilecek görüntüyü belirtir.

ROS Düğümü Oluşturma

import rospy
import cv2
from cv_bridge import CvBridge
from sensor_msgs.msg import Image

class Kamera():
def_init__(self):
rospy.init_node("kamera_dugumu")
self.bridge CvBridge()
rospy. Subscriber("camera/rgb/image_raw",age
rospy.spin()

```
def kameraCallback(self,mesaj):
    img = self.bridge.imgmsg_to_cv2(mesaj,"bgr8")
```

```
    cv2.imshow("Robot Kamerasi,ing)
    cv2.waitKey(1)
```

Kamera()

Renk Uzayları ve Uzay Dönüşümleri:
Renk uzayı renklerin farklı şekillerde temsil edildiği sistemlerdir.
BGR renk uzayında görüntüdeki her piksel değeri, mavi (B), yeşil (G) ve kırmızı (R değerleri ile temsil edilir.
HSV renk uzayında görüntüdeki her pikse renk özü (H), doygunluk (S) ve parlaklık değeri (V) ile temsil edilir. Renk özü, rengin baskın dalga uzunluğunu, doygunluk rengin canlılığını, parlaklık rengin aydınlığını (içindeki beyaz oranını) belirtir.

Aritmetik İşlemler
cv2.add(imge1, imge2): İki görüntüyü toplamak için kullanılır. Bu komut için her iki görüntünün aynı boyutlarda ve aynı türde olması gerekmektedir. Bu komut, toplama işleminde doygunluğa ulaşan bir işlem sağlar.
cv2.addWeighted(imge 1, ağırlık 1, imge 2, ağırlık2): Ağırlıklı toplama işlemi için kullanılır. ağırlık 1 değeri imge 1 görüntüsüne uygulanacak ağırlığı belirtirken, ağırlık2 değeri imge2 görüntüsüne uygulanacak ağırlığı belirtir. Ağırlık değerlerinin toplamı (ağırlık1 + ağırlık2) 1 olmalıdır.
cv2.subtract(imge 1, imge2): İki görüntüyü çıkarmak için kullanılır.

Maskeleme ve Bit Bazında İşlemler
Maskeleme, görüntüde ilgilenilmek istenen veya ilgilenilmek istenmeyen bölgelerin bir maske ile görüntüden çıkarılması işlemidir.
cv2.bitwise_and(kaynak1, kaynak2, mask): İki görüntüyü bit bazında AND işlemine sokmak için kullanılır. Bu sözdiziminde, kaynak 1 ilk görüntü dizisini, kaynak2 ikinci görüntü dizisini ve mask işlem maskesini belirtir.
cv2.bitwise_or(kaynak 1, kaynak2, mask): İki görüntüyü bit bazında OR işlemine sokmak için kullanılır. Bu sözdiziminde, kaynak 1 ilk görüntü dizisini, kaynak2 ikinci görüntü dizisini ve mask işlem maskesini belirtir.
cv2.bitwise_xor(kaynak1, kaynak2, mask): İki görüntüyü bit bazında XOR işlemine sokmak için kullanılır. Bu sözdiziminde, kaynak1 ilk görüntü dizisini, kaynak2 ikinci görüntü dizisini ve mask işlem maskesini belirtir.
cv2.bitwise_not(kaynak, mask): Görüntüyü bit bazında NOT işlemine sokmak için kullanılır. Bu sözdiziminde, kaynak giriş görüntü dizisini ve mask işlem maskesini belirtir.

Kameradan Canlı Görüntü Alma:

```
import cv2
import numpy as np
```

```
cap = cv2.VideoCapture (0)
while True:
_,img = cap.read()
img = cv2.resize(ing, (640,480))
hsv = cv2.cvtColor(img,cv2.COLOR_BGR2HSV)
alt_mavi = np.array([110,100,100))
ust_navt = np.array([130,255,255])
maske = cv2.inRange(hsv,alt_mavi,ust_mavi)
sonuc = cv2.bitwise_and(img,img, mask = maske)
cv2.imshow("Orjinal",img)
cv2.imshow("Maske", maske)
cv2.imshow("Sonuc", sonuc)
k = cv2.waitKey(1)
if k == 27:
break

cv2.destroyAllWindows()
```

not: bu kodu çalıştırdığımızda 3 görüntü oluşur; orjinal,maske ve sonuc.
orjinal mavi olan görüntüyü olduğu gibi verir.
maske şeklini beyaz gösterir nokta nokta halinde.
sonuc sadece mavi rengini ve yine nokta nokta halinde gösterir.
ve bu kod sadece mavi olan nesneleri gösterir.

Serit Takip Etme:

```
import cv2
import numpy as np
import rospy
from sensor_msgs.msg inport Image
from geometry msgs.msg import Twist
from cv bridge import CvBridge

class SeritTakip():
def__init__(self):
rospy.init_node("serit takip")
self.bridge = CvBridge()
rospy.Subscriber("camera/rgb/Lmage_raw", Image,self.kameraCallback)
self.pub = rospy.Publisher("cnd_vel", Twist, queue size = 10)
self.hiz_mesaji = Twist()
rospy.spin()
```

```
def kameraCallback(self,mesaj):
img = self.bridge.imgmsg_to_cv2(mesaj, "bgr8")
hsv = cv2.cvtColor(img,cv2.COLOR_BGR2HSV)
alt_sari = np.array([20,100,100])
ust_sari = np.array([48,255,255])
maske = cv2.inRange(hsv,alt sart,ust_sari)
sonuc = cv2.bitwise_and(img, img, mask=maske)
M = cv2.moments(maske)
if M['m00'] > 8:
cx = int(M['m10']/M['m00'])
cy= int(M['m01']/M['m00']) 34
cv2.circle(img, (cx,cy), 5, (255,0,0),1)
self.hiz_mesaji.linear.x = 0.2
self.pub.publish(self.hiz_mesaji)

else:
self.hiz_mesaji.linear.x = 0.0
self.pub.publisher(self.hiz_mesaji)
cv2.imshow("Orijinal",img)
cv2.imshow("Maske",maske)
cv2.imshow("Sonuc",sonuc)
cv2.waitkey(1)

SeritTakip()
```

Find-Object ile Nesne Tanıma Uygulaması

```
import rospy
from find object_2d.msg import ObjectsStamped

class NesneTanima():
11 def init (self):
rospy.init_node("nesne tanima")
rospy.spin() 14
rospy. Subscriber("objectsStamped ObjectsStamped,self.nesneTani)
def nesneTani(self,mesaj): 16
try:
self.nesne_id = mesaj.objects.data[0]
print(self.nesne_Ld)
if self.nesne_id=1:
print("Koni Bulundu")
```

except IndexError:
print("Herhangi bir nesne bulunamadi !!!")

Nesne Tanima()