Avrupa Bilim ve Teknoloji Dergisi
Sayı 31 (Ek Sayı 1), S. 127-133, Aralık 2021

Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
No. 31 (Supp. 1), pp. 127-133, December 2021

Research Article

Derin Öğrenme ile Nesne Tanıyan Robot

Uğur Talaş[1*], Uğur Yüzgeç[2], Burakhan Çubukçu[3]

[1*] Bilecik Şeyh Edebali Üniversitesi, Mühendislik Fakültesi, Bilgisayar Bölümü, Bilecik, Türkiye, (ORCID: 0000-0002-9287-413X), ugur.talas@bilecik.edu.tr
[2] Bilecik Şeyh Edebali Üniversitesi, Mühendislik Fakültesi, Bilgisayar Bölümü, Bilecik, Türkiye, (ORCID: 0000-0002-5364-6265), ugur.yuzgec@bilecik.edu.tr
[3] Bilecik Şeyh Edebali Üniversitesi, Mühendislik Fakültesi, Bilgisayar Bölümü, Bilecik, Türkiye, (ORCID: 0000-0003-0480-1254), burakhan.cubukcu@bilecik.edu.tr

(İlk Geliş Tarihi 5 Temmuz 2021 ve Kabul Tarihi 16 Aralık 2021)

(**DOI:** 10.31590/ejosat.962558)

ATIF/REFERENCE: Talaş, U., Yüzgeç, U. & Çubukçu, B. (2021). Derin Öğrenme ile Nesne Tanıyan Robot. *Avrupa Bilim ve Teknoloji Dergisi*, (31), 127-133.

Öz

Günümüzde birçok farklı alanda insanlara fayda sağlamak için teknolojik cihazlar ve robotlar kullanılmaktadır. Özellikle askeri alanda insan hayatının riske girebileceği ortamlarda robotlar yardımıyla, hayati riskler minimize edilmek istenilmektedir. Askeri operasyonlarda bir binaya keşif amaçlı bir insanın girmesi oldukça riskli bir durumdur. Bu çalışmada bu tür riskli durumlarda insanın keşif yapması yerine uzaktan kontrol edilebilen, gördüğü nesneleri tanıyabilen ve tanıdığı nesneleri kontrol ekranında gösteren bir robot tasarlanmıştır. Bu çalışmada geliştirilen robot nesne tanımak için Google tarafından geliştirilen TensorFlow derin öğrenme kütüphanesini kullanmaktadır. Python diliyle geliştirilen yazılım robot üzerinde bulunan Raspberry Pi3/B mini bilgisayarı üzerinde çalıştırılmıştır. Robot hareketi için DC motorlardan faydalanılmıştır. Raspberry Pi3/B mini bilgisayarı üzerindeki GPIO pinleri ile motor sürücü devresine sinyal gönderilerek robotun hareketlerinin kontrol edilebilmesi sağlanılmıştır. Yapılan prototipin testlerinde nesneleri çoğunlukla başarılı şekilde tanınabildiği ve uygun ışık ortamında başarı oranının arttığı gözlemlenmiştir.

Anahtar Kelimeler: TensorFlow, Derin Öğrenme, Nesne Tanıma, Bilgisayar Görmesi, Robotik.

Object Recognizing Robot Application with Deep Learning

Abstract

Today, technological devices and robots are used to benefit in many different areas. In environments where people of other military field may be at risk, vital risks are desired to be minimized for robots. It is very risky for a person to enter a building for reconnaissance purposes during military operations. It is the place where this kind of risky person learns a robot that can be remotely controlled, recognize the text he sees, and display the text control text he knows, instead of making exploration. In this structure, the robot uses the TensorFlow deep learning library offered by Google to recognize objects. It was run on the Raspberry Pi3/B minicomputer on the software with the language of Python. DC motors are used for robot movement. In Raspberry Pi3/B minicomputer, the robot's movements can be controlled by sending a signal to the motor driver circuit with GPIO pins. In the tests of the prototype, it has been observed that the guarantee of success in the distribution area has increased.

Keywords: TensorFlow, Deep Learning, Object Recognition, Computer Vision, Robotics.

[*] Sorumlu Yazar: ugur.talas@bilecik.edu.tr

1. Giriş

Literatürde "Artificial Intelligence" olarak adlandırılan yapay zeka terimi ilk kez 1956 yılı Dartmouth Konferansında Stanford Üniversitesi öğretim üyelerinden Prof. John McCarthy tarafından ortaya atılmıştır (McCarthy, 2006) (Abadi, Agarwal ve diğerleri, 2016). 1956'dan ünümüze yapay zeka alanında bir çok akademik çalışma yapılarak geliştirilmiş ve özellikle teknolojinin ilerlemesiyle bu alanda yapılan çalışmaların, uygulamalardaki kullanımı da artmıştır(Abadi, Barham ve diğerleri, 2016). Günümüzde yapay zeka kullanan akıllı cihazlar ve robotlar geliştirilmeye başlanmıştır. Yapay zeka yardımıyla geliştirilen bu akıllı robotlar uygulanacak tasarıma göre kamera, sensor, internet v.b. yöntemlerle elde ettikleri verileri işleyebilme, verileri sınıflayabilme, bu verilerden çıkarım yapabilme ve karar verebilme gibi zeka gerektiren yeteneklere sahiptirler (Kocamaz, 2012; Köse, 2017).

Bilgisayar görmesi yapay zekanın bir alt dalı olarak adlandırılmaktadır. Bilgisayar görmesi alanında yapılan çalışmalar temel olarak kameradan elde edilen görselden bir anlam çıkarılmasına dayanmaktadır. Kameradan anlık olarak alınan görsel üzerinde görüntü işlemek, görüntü üzerinde farklı türde nesnelerin tespitini yapmak oldukça zor bir problemdir. Bu işlemleri yapabilmek için bilgisayar görmesi alanında yapılan çalışmalar sonucunda elde edilen teknik ve yöntemler kullanılarak görüntüyü işlemek ve değerlendirmek daha kolay bir şekilde yapılabilir (Dey, 2016). Literatürde bilgisayar görmesi yöntem ve tekniklerinin hazır olarak uygulanabilmesini sağlayan TensorFlow, Keras, Theano, Torch gibi derin öğrenme kütüphaneleri bulunmaktadır (E. R. Davies, 2005). Yapılan çalışmalara göre uygulama ortamındaki CPU gücü, GPU gücü, işlenmek istenen görüntünün boyutları gibi faktörler hazır kullanılan derin öğrenme kütüphanelerinin başarım sonuçları etkilemektedir(Parvat ve diğerleri, 2017; Shatnawi ve diğerleri, 2018; Vassili Kovalev ve diğerleri, 2016; Yapıcı ve diğerleri, 2021).

Bu çalışmada nesne tanıma işlemleri için TensorFlow derin öğrenme kütüphanesi kullanılmıştır. TensorFlow Google çalışanlarının 9 Kasım 2015'de yayınladığı bir makale ile birlikte dünyaya duyurulmuştur. TensorFlow, üreticisi Google'ın tanımıyla veri akış grafikleri kullanarak nümerik hesaplar yapabilen, açık kaynak kodlu bir derin öğrenme kütüphanesidir (Abadi, Barham ve diğerleri, 2016). TensorFlow'un nesne tanıma işlemleri için SSD (Single Shot Detection), CNN (Convolutional Neural Network), R-CNN (Recurrent Convolutional Neural Network) ve hızlı R-CNN (Fast R-CNN) gibi birbirinden farklı yapay sinir ağları ile eğittiği açık kaynak kodlu modelleri bulunmaktadır (Girshick ve diğerleri, 2014; Shaoqing Ren, Kaiming He, Ross Girshick, 2017; Tensorflow, 2019). Buradaki modellerin birbiri arasında avantajlı veya dezavantajlı oldukları durumları vardır (Aydın, 2020). Hızlı R-CNN ile geliştirilen bir model daha yüksek oranda doğru nesne tahmini yapabilirken, SSD ile geliştirilen modelin doğru tahmin oranı daha düşüktür. Ancak SDD model, Hızlı R-CNN'e göre daha az sistem kaynağı kullandığından düşük donanımlı cihazlarda SSD model kullanılması avantajlı 2017olmaktadır (Pena ve diğerleri, 2017.; Phon-Amnuaisuk ve diğerleri, 2018). Bu çalışma kapsamında yapılan robotta, yazılım Raspberry Pi3/B mini bilgisayarı üzerinde çalıştırıldığından düşük donanımlara uyumlu SDD model tercih edilmiştir ("TensorFlow", 2021).

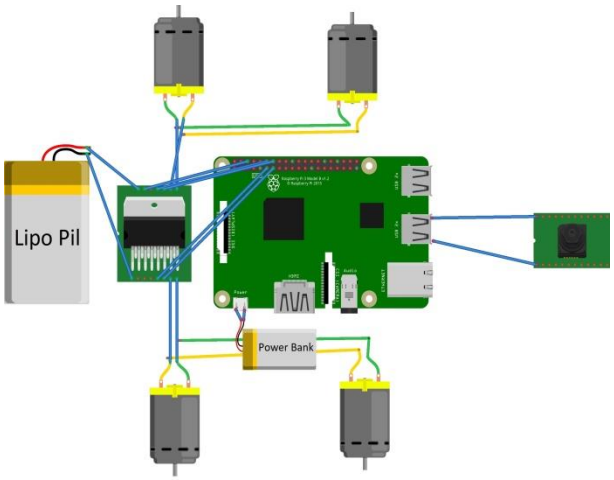
Literatürde TensorFlow ve derin öğrenme teknikleri kullanılarak birçok farklı çalışmalar yapılmıştır. İnsan becerilerini taklit ederek, cerrahi operasyonlarda kullanılmak üzere tasarlanan robotlar (Krishnan ve diğerleri, 2017), navigasyon cihazlarının yol kenarlarındaki trafik işaretlerini tanıması(Puthussery ve diğerleri, 2017), engelli ve yaşlı bakımı yapılan evlerde uygunsuz durumların tespiti yapılması (Yang ve diğerleri, 2018), geri dönüşüm atıklarının sınıflandırılması (Sağlam ve diğerleri, 2020), insansız su araçları (Ataner ve diğerleri, 2020) gibi derin öğrenme yardımıyla geliştirilen birçok güncel çalışmalar literatürde yer almaktadır.

Bu çalışmada savunma sanayinde kullanılması amacıyla, uzaktan kontrol edilebilen, gördüğü nesneleri ve canlıları tanıyabilen bir mobil robot tasarlanmıştır. Robot özellikle hayati risk taşıyan ortamlarda, insan yerine robot kullanılarak bölgede nesneleri veya insan olup olmadığını tespit edebilmektedir. Python diliyle geliştirilen robotun yazılımı TensorFlow derin öğrenme kütüphanesini kullanmaktadır. Robotun donanımında Raspberry Pi3/B mini bilgisayarı ve DC motorlar kullanılmıştır.

Çalışmanın ikinci bölümünde robotun donanımsal yapısından, üçüncü bölümünde kullanılan TensorFlow kütüphanesinden, dördüncü bölümünde oluşturulan yazılımdan, son bölümde ise elde edilen sonuçlardan bahsedilmiş ve daha sonra yapılabilecek çalışmalar için önerilere yer verilmiştir.

2. Robot Donanımı

Robot devresinin gövdesi piyasada hazır olarak satılan pleksi şeffaf bir gövdedir. Bu gövde üzerinde 4 DC motor, L298N motor sürücü devresi, Raspberry Pi3/B mini bilgisayarı, lipo pil, batarya ve kamera bağlanarak bulunmaktadır (Şekil 1).

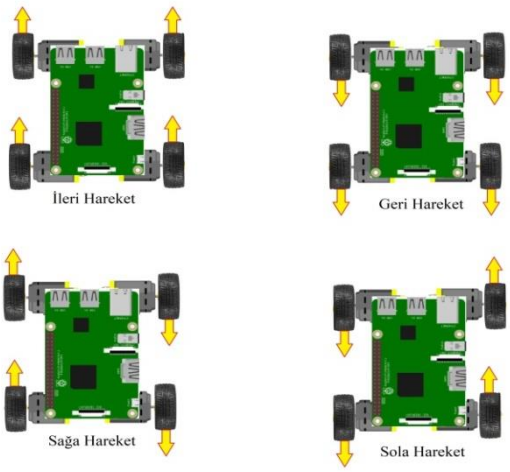


Şekil 1. Robot devre şeması

Şekil 1'de görüldüğü üzere devrede iki adet enerji kaynağı bulunmaktadır. L298N motor sürücü devresi -20 ile +135 arası sıcaklıkta, 5 ile 35 volt aralığında ve maksimum 2 amper ile çalışmaktadır (Celik ve diğerleri, 2018). Motorları beslemek için taşınabilir şarj aletine göre daha fazla güç sağlayabilecek

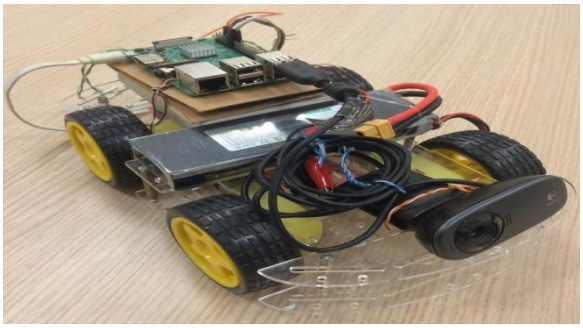
Olan lipo pil kullanılmıştır. Taşınabilir şarj aleti ise Raspberry Pi3/B'yi beslemek için kullanılmıştır. Raspberry Pi3/B'nin 4 adet GPIO pini motor sürücü devresine sinyal göndermek için kullanılmıştır. Bu pinlerden 11 ve 14'nolu pinleri motor sürücü devrenin 1 ve 2'nolu girişlerine; 15 ve 16'nolu pinler ise motor sürücü devrenin 3 ve 4'nolu girişlerine bağlanmıştır. Motor sürücü devre iki adet motoru ileri ve geri hareket ettirebilmektedir. 1'nolu pine sinyal geldiğinde soldaki

motoru ileri 2'nolu pine sinyal geldiğinde ise soldaki motoru geri hareket ettirmektedir. Motor sürücü devrenin 3 ve 4 numaralı girişleri de aynı şekilde sağdaki motoru hareket ettirmektedir. Robot üzerinde toplam dört adet motor kullanılacağından sağ ve sola paralel şekilde bağlı ikişer motor yerleştirilmiştir. Böylelikle robotun sağında ve solunda bulunan motorlar birbirinden bağımsız ancak sağda ve solda bulunan ikili motorlar birbirine bağımlı hareket emiş olacaklardır. Şekil 2'de robotun ileri, geri, sağa ve sola hareketleri gösterilmiştir.

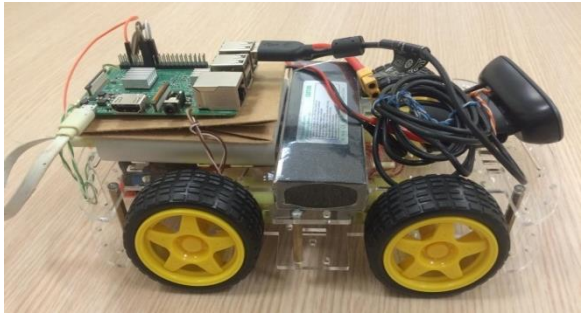


Şekil 2. Robot hareket yönleri

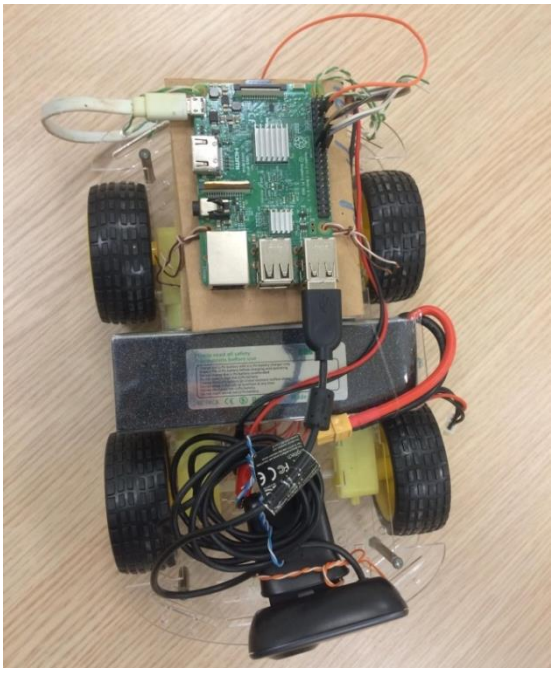
Şekilden de görüleceği üzere robot ileri, geri, sağa dönme ve sola dönme olarak toplamda dört farklı hareket yapmaktadır. İleri ve geri hareketlerde tüm motorlar ileri veya geri olarak hareket ettirilmektedir. Dönme hareketinde ise sağa dönerken sağ motorlar geri sol motorlar ise ileri hareket ettirilmektedir; sola dönerken ise soldaki motorlar geri sağdaki motorlar ise ileri hareket ettirilmektedir. Robotun tamamlanmış haline ait görseller Şekil 3-5'de gösterilmektedir.



Şekil 3. Robot resmi I



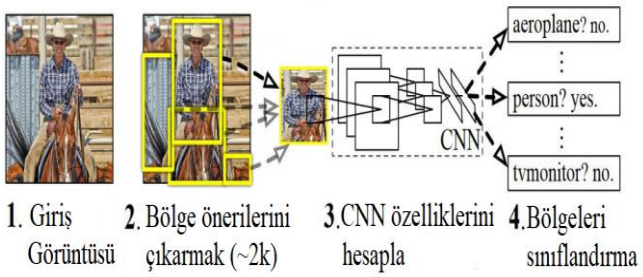
Şekil 4. Robot resmi II



Şekil 5. Robot resmi III

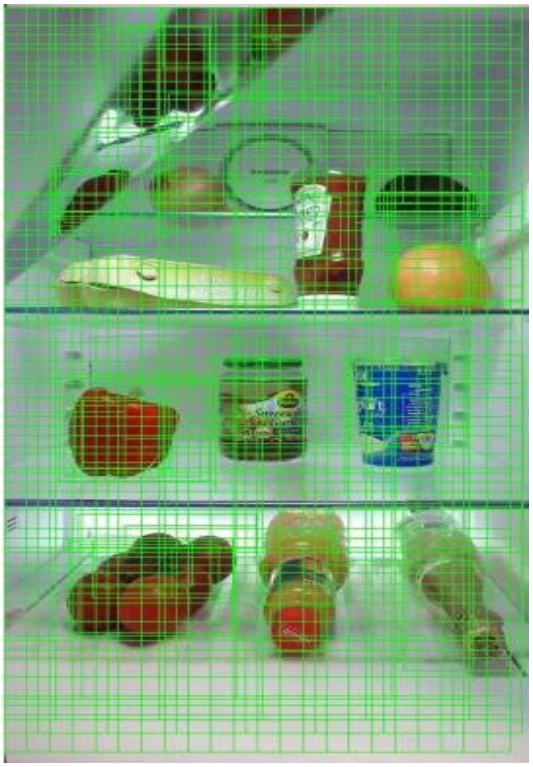
3. Derin Öğrenme Kütüphanesi (TensorFlow)

TensorFlow Kütüphanesi, veri akış grafiklerini kullanarak sayısal hesaplama için açık kaynaklı bir yazılım kütüphanesidir. TensorFlow derin öğrenme işlemleri gerektiren birçok alanda kullanılabilir. Bu çalışma kapsamında TensorFlow Kütüphanesi nesne tanımak için kullanılmıştır. TensorFlow nesne tanıma işlemi yaparken bölge önerisi yaparak tüm resimde nesne aramak yerine resmi bölgelere ayırıp sadece bu bölgelerde nesne arayarak yapacağı işlem sayısını azaltmaktadır. Şekil 6'da TensorFlow R-CNN yapısı ile nesne algılama işlemi gösterilmiştir.



Şekil 6. TensorFlow R-CNN yapısı ile nesne algılama işlemi (Object Detection using Fast R CNN | Azure AI Gallery, 2021).

R-CNN'nin temel fikri, başlangıçta milyonlarca görüntü kullanarak görüntü sınıflandırması için eğitilmiş derin bir sinir ağı nesne algılama amacıyla değiştirmektir. Şekilden de görüleceği gibi bir giriş görüntüsü verildiğinde ilk olarak çok sayıda bölge önerisi üretilmektedir. Bu bölge önerileri bağımsız bir şekilde CNN modeline gönderilir. Son olarak, her bir bölge için bir etiket ve sınıflandırıcı sonucu çıkış olarak döndürülür. Şekil 7'de yine bir örnek görüntü üzerinde TensorFlow Kütüphanesi ile bulunan bölge önerileri gösterilmiştir. Bu görselde TensorFlow Kütüphanesi ile bölge önerilerinde bulunulmuş ve önerilen bölgeler yeşil kutularla işaretlenmiştir.



Şekil 7. TensorFlow kütüphanesi ile bulunan bölge önerileri (Object Detection using Fast R CNN | Azure AI Gallery, 2021).

Resmin orijinali düşünüldüğünde bu resim üzerinde milyonlarca farklı boyutta dikdörtgen çizilip bunlar üzerinde nesne arama işlemi yapılabilirdi ancak bölge önerisi bu sayıyı 1000-2000 bölgeye ayırarak yapılacak işlem sayısını oldukça düşürmektedir.

Kameradan alınan anlık görüntülerde her bir çerçeve için yaklaşık 2000 bölge üzerinde nesne arama işlemi yapılması özellikle düşük donanımlı cihazlar için çok mümkün değildir. Literatürdeki çalışmalara bakıldığında kullanılacak tekniğe göre bölge önerisinde bulunmayan veya bulunduktan sonra CNN, R-CNN, Hızlı R-CNN, SSD gibi farklı konvolüsyonel sinir ağları ile yapılan uygulamaların nesne tanıma işlemini hızlandırdığı görülmüştür. Google ekibi tarafından, bu teknikler için hazır modeller geliştirilmiştir. Geliştirilen modeller Github üzerinden açık kaynak kodlu olarak sunulmaktadır. Tablo 1'de hazır sunulan ön eğitimli modellerin örnek birkaç tanesine ait çalışma hızları ve çıktı türleri verilmiştir (Github, 2021).

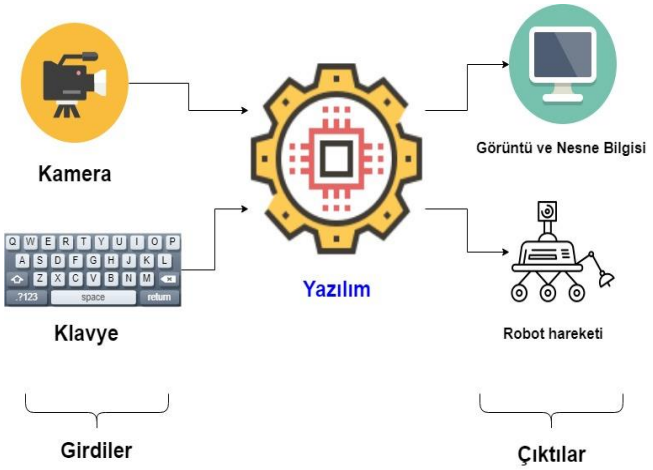
Tablo1. Github'ta sunulan eğitilmiş ağların hız bilgileri.

Model ismi	**Süre (ms)**	**Çıkış Türü**
ssd mobilenet v1 coco	30	Kutular
ssd resnet 50 fpn coco	76	Kutular
ssd mobilenet v2 coco	31	Kutular
ssdlite mobilenet v2 coco	27	Kutular
ssd inception v2 coco	42	Kutular
faster cnn inception v2 coco	58	Kutular
faster rcnn resnet50 coco	89	Kutular
rfcn resnet101 coco	92	Kutular
mask rcnn inception v2 coco	79	Maskeler

Github'da sunulan modeller arasında düşük seviyeli donanımlar için tavsiye edilenleri SSD modelleridir. SSD modeller diğerlerine göre nesne bulma isabeti daha düşük ancak yaptığı işlem sayısı daha az olduğu için işlemci üzerinde görüntü işleme yapmak için daha uygundur. Çalışma kapsamında Raspberry Pi3/B üzerinde nesne tanıma işlemi yapılacağından, SSD Mobilenet V1 Coco modeli seçilmiştir. Seçilen model hazır ön eğitimli bir derin öğrenme modelidir. (Abadi, Agarwal ve diğerleri, 2016; Tensorflow, 2019).

4. Robotun Yazılımı

Çalışmanın tamamında Python dili kullanılmıştır. Kameradan görüntüyü elde etmek ve görüntü işleme için OpenCV kütüphanesinden faydalanılmıştır. Görüntü üzerinden nesne tanımak için ise TensorFlow kütüphanesinden faydalanılmıştır. Geliştirilen yazılım Raspberry Pi3/B mini bilgisayarı Rasbian işletim sistemi üzerinde çalıştırılmıştır. Yazılıma başlamadan önce Raspberry Pi üzerine sırasıyla Rasbian işletim sistemi, OpenCV ve TensorFlow kurularak çalışma ortamı hazır hale getirilmiştir. Robota uzaktan erişim sağlamak için kart üzerine RDP kurulumu gerçekleştirilmiştir. RDP ile uygulamanın hem geliştirilme hem de robotu çalışırken kumanda edebilmek için PC ile uzaktan erişim sağlanmıştır. Şekil 8'de çalışmanın yazılım girdi-çıktı diyagramı gösterilmiştir.

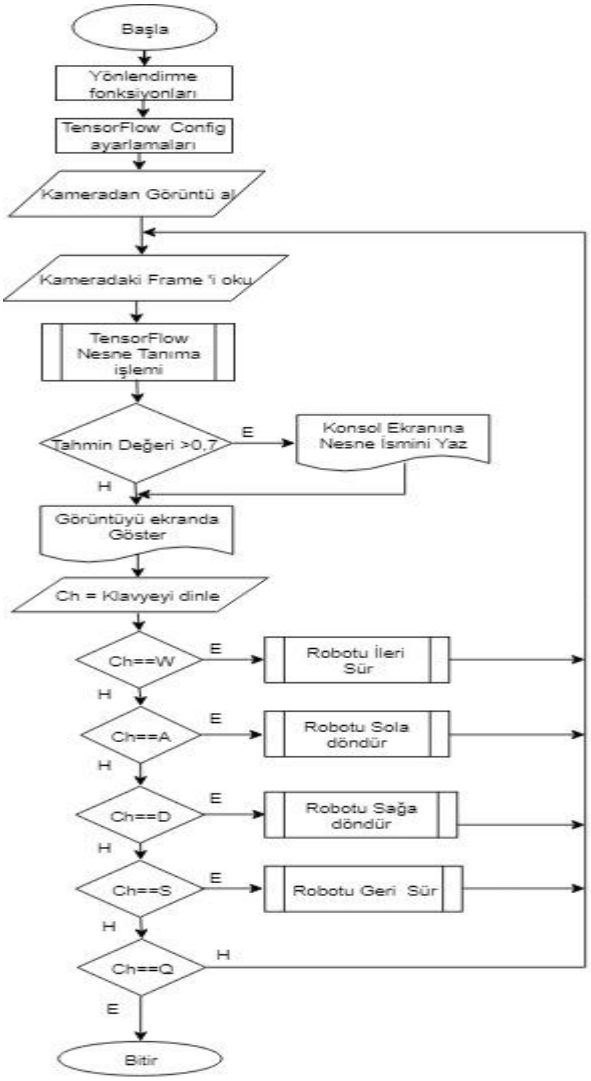


Şekil 8. Yazılım girdi çıktı diyagramı.

Geliştirilen yazılımda iki farklı kaynaktan girdi alınıp, iki farklı çıktı verilmektedir. Girdilerden biri kameradan alınan görüntüdür, diğer girdi ise klavyeden alınan robotun hareket bilgilerdir. Yazılımın çıktılarından biri kameradan elde edilen görüntüyü nesne tanıma işleminden sonra tespit edilen nesneleri işaretli olarak ekrana yansıtması ve liste olarak kontrol ekranında göstermesidir. Diğer çıktı ise motor sürücü devresine GPIO pinlerinden gönderilen kontrol sinyallerdir. Şekil 9'da çalışma kapsamında geliştirilen yazılıma ait akış diyagramı gösterilmiştir.

Akış diyagramı incelendiğinde robotun çalışmaya hazır olduğu ve görüntü işleme yaptığı yer alt bölümde bulunan döngüde gerçekleşmektedir. Bu döngü öncesinde referanslar ve tanımlamalar yapılmıştır. Tanımlamalar sonrasında döngü içerisindeki kod sayısını arttırmamak adına robot hareketini sağlayan komutlar ayrı bir fonksiyon olarak tanımlanmıştır. Fonksiyonlar ileri, geri, sağ ve sol olarak isimlendirilmiştir ve

döngü içerisinden bu fonksiyonlar çağrılarak robot hareketi sağlanmaktadır. Fonksiyonların altında TensorFlow için gerekli olan tanımlamalar yapılmaktadır. Bu tanımlamalarda hangi model kullanılacağı bu modelin dosya yolu gibi ayarlar yapılmaktadır. Döngü öncesinde son olarak OpenCV yardımıyla kameradan görüntü alınması sağlanmıştır.

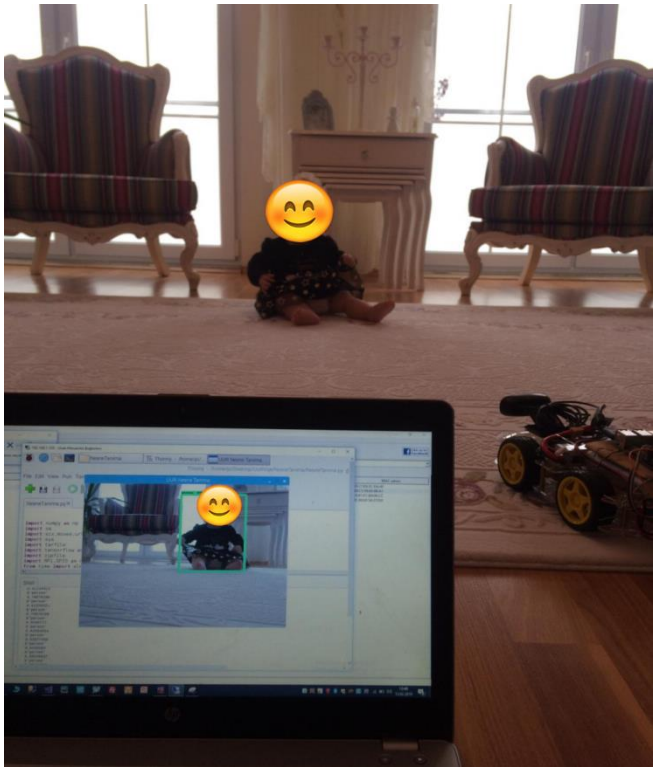


Şekil 9. Yazılım akış diyagramı.

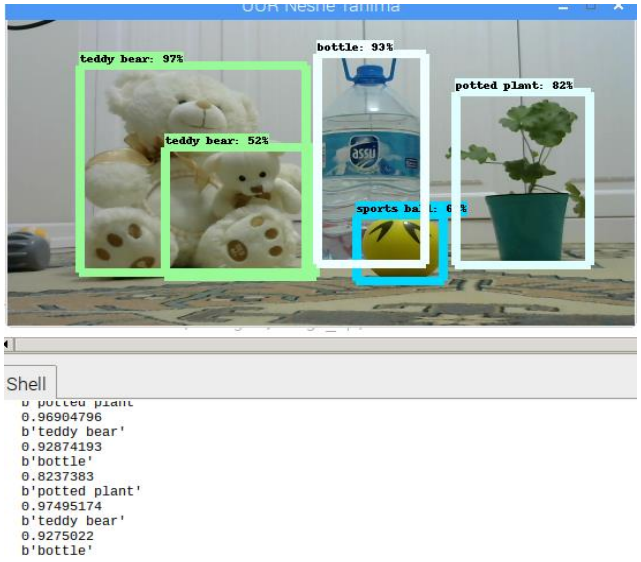
Oluşturulan döngü sonsuz olarak çalışacak şekilde planlanmıştır. Döngüyü kırmak için yazılan komut "Q" tuşuna basılmasıyla aktif olmaktadır. Döngü her iterasyonda anlık kameradan okunan çerçeveyi alıp Tensor Flow kodunu çalıştırmaktadır. TensorFlow cevap olarak bir dizi ve gönderilen çerçeve üzerinde bulduğu nesneleri işaretleyerek geri döndürmektedir. TensorFlow çerçeve üzerinde birçok tahmin yapmaktadır ve bu tahminlerin doğruluk yüzdesini vermektedir. Bu çalışmada doğruluk yüzdeliğinin eşik değeri %70 olarak belirlenmiştir. Yüzde 70'lik doğruluk oranının altında bulunan tahminler dikkate alınmayacaktır. Yüzde 70'in üzerinde olan tahminler ise robotun uzaktan yönlendirildiği konsol ekranına yazdırılmıştır. Görüntü üzerinde bu bilgiler verilse de konsol ekranına yazdırılarak buradan da takibin sağlanmasına olanak tanınmaktadır. İşaretlenen görüntü ekrana yazdırıldıktan sonra klavye dinlenilip robot için hareket komutu verilip verilmediği kontrol edilip döngü başa dönmektedir. "Q" Tuşuna basılıp yazılım durdurulana kadar bu döngü tekrar etmektedir.

5. Uygulama Testleri

Çalışmanın donanım ve yazılım bileşenleri tamamlandıktan sonra robot canlı ortamda test edilip rototun istenilen hedeflere uygun olarak çalışıp çalışmadığı gözlemlenmiştir. Şekil 10'da robotun canlı ortamda bir bebeği tanıyıp ekranda işaretlendiği görülmektedir. Şekil 11'de bir başka uygulama çıktısı konsol ekran çıktısı ile birlikte gösterilmiştir.



Şekil 10. Nesne tanıma uygulama çıktısı I.



Şekil 11. Nesne tanıma uygulama çıktısı II.

Uygulama testleri sırasında tanıdığı örnek nesnelerin başarım yüzdeleri Tablo 2'de verilmiştir. Gerçekleştirilen bu testler sonucunda bazı olumsuzluklar gözlenmiştir. Bir çerçeve görüntüyü işlemek yaklaşık 3 saniye sürmektedir. Bu nedenle kameradan gelen görüntülerde gecikme yaşanmaktadır. Robot gerçek zamanda bir insan gördüğünde robotu kontrol eden kişi bu görüntüyü 3 saniye sonra görecektir. Bu gecikme robotun kullanım alanı kurgusundaki askeri operasyonlar için oldukça

önemli bir zaman kaybıdır. Ayrıca burada yaşanan gecikme görüntü işleme adımından sonra klavyenin dinlenmesi aşamasında da gecikmeye neden olduğundan robotun efektif bir şekilde kontrol edilmesini engellemektedir.

Tablo 2. Örnek nesne tanıma başarım yüzdeleri.

Nesne	Doğruluk Yüzdesi
İnsan	97
Şişe	93
Top	88
Bitki	82
Laptop	92

6. Sonuç

Bu çalışmada ağ üzerinden kontrol edilen ve TensorFlow derin öğrenme Kütüphanesi ile nesne tanıma işlemi yapan bir robot gerçekleştirilmiştir. Çalışmada donanım olarak Raspberry Pi3/B mini bilgisayarı kullanılmıştır. Python dili ile gerçekleştirilen yazılımda, SSD Mobinet V1 modelini kullanan TensorFlow derin öğrenme kütüphanesi kullanılmıştır. Robotun hareketi için ise DC motorlardan ve L298N motor sürücü kartından yararlanılmıştır.

Yapılan uygulamalarda nesne tanıma işlemlerinin başarıyla gerçekleştirildiği ancak bu işlemlerin 640x480 çözünürlüklü ve 24 frame per second (fps) bir görüntüde ışığın şiddetine bağlı olarak nesne tanıma işleminin 3-4 saniye gecikme ile gerçekleştiği gözlemlenmiştir. Yaşanan bu gecikme robotun uzaktan kontrolünü zorlaştırmaktadır. Kullanılan Raspberry Pi3/B donanımının nesne tanıma işlemleri için yeterli olmadığı gözlemlenmiştir. İşlem gücü daha yüksek olan donanımlar kullanıp sonuçların gözlemlenmesi gerekmektedir. Nesne tanımak için kullanılan TensorFlow'un derin öğrenme kütüphanesinin birçok modeli bulunmaktadır. Gelecek çalışmalarda, bu modeller sisteme ayrı ayrı tanımlanıp karşılaştırılarak diğer derin öğrenme modelleri ile de robot çalıştırılıp en performanslı olanı seçilmelidir. Bu karşılaştırma işlemleri yapıldığında robot istenilen seviyedeki hassasiyet ile kontrol edilebilecektir. Robot genel olarak keşif amaçlı kurgulanmıştır. Robotun sahada kullanılabilmesi için düz olmayan zeminlere uygun tekerlek yapısı geliştirilmesi gerekmektedir. Robot genel anlamda keşif amaçlı kullanıma uygun tasarlanmıştır ancak robotun üzerine bir silah yerleştirilerek taarruz amaçlı kullanılması için de çalışmalar yapılabilir. Bu sayede askeri bir çatışmadaki hayati risklerde minimize edilmiş olacaktır.

Kaynakça

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... & Zheng, X. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467.

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... & Zheng, X. (2016). Tensorflow: A system for large-scale machine learning. In 12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16) (pp. 265-283).

Ataner, E., Özdeş, B., Öztürk, G., Yasin, T., Çelik, C., Durdu, A., … Öz, Y. (2020). Deep Learning Methods in Unmanned Underwater Vehicles. European Journal of Science and Technology Special Issue, 345–350. doi:10.31590/ejosat.804599

Aydın, Z. (2020). Performance Analysis of Machine Learning and Bioinformatics Applications on High Performance Computing Systems. Academic Platform Journal of Engineering and Science, 8(1), 1–14. doi:10.21541/apjes.547016

Celik, Y. ve Güneş, M. (2018). Designing an Object Tracker Self-Balancing Robot. Academic Platform Journal of Engineering and Science, 6(2), 124–133. doi:10.21541/apjes.414715

Dey, A. (2016). Machine Learning Algorithms: A Review. International Journal of Computer Science and Information Technologies, 7(3), 1174–1179.

E. R. Davies. (2005). Machine Vision: Theory, Algorithms, Practicalities, Elsevier.

Girshick, R., Donahue, J., Darrell, T. ve Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation.

Github, (2021). 20 Mart 2021 tarihinde https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf1_detection_zoo.md adresinden erişilmiştir.

Kocamaz, A. F. (2012). Makine Öğrenmesi Tabanlı Bir Uzman Sistem Tasarımı.

Köse, U. (2017). Yapay zeka tabanlı optimizasyon algoritmaları geliştirilmesi (Doctoral dissertation, Doktora Tezi, Selçuk Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği ABD).

Krishnan, S., Fox, R., Stoica, I., & Goldberg, K. (2017, October). Ddco: Discovery of deep continuous options for robot learning from demonstrations. In Conference on robot learning (pp. 418-437). PMLR.

McCarthy, J., Minsky, M. L., Rochester, N. ve Shannon, C. E. (2006). A proposal for the Dartmouth summer research project on artificial intelligence. AI Magazine, 27(4), 12–14.

Object Detection using Fast R CNN | Azure AI Gallery. (2021). 15 Mart 2021 tarihinde https://gallery.azure.ai/Tutorial/Object-Detection-using-Fast-R-CNN-1 adresinden erişildi.

Parvat, A., Chavan, J., Kadam, S., Dev, S., & Pathak, V. (2017, January). A survey of deep-learning frameworks. In 2017 International Conference on Inventive Systems and Control (ICISC) (pp. 1-7). IEEE.

Pena, D., Forembski, A., Xu, X., & Moloney, D. (2017, July). Benchmarking of CNNs for low-cost, low-power robotics applications. In RSS 2017 Workshop: New Frontier for Deep Learning in Robotics (pp. 1-5).

Phon-Amnuaisuk, S., Murata, K. T., Pavarangkoon, P., Yamamoto, K., & Mizuhara, T. (2018). Exploring the applications of faster R-CNN and single-shot multi-box detection in a smart nursery domain.

Puthussery, A. R., Haradi, K. P., Erol, B. A., Benavidez, P., Rad, P., & Jamshidi, M. (2017, June). A deep vision landmark framework for robot navigation. In 2017 12th system of systems engineering conference (SoSE) (pp. 1-6). IEEE.

Sağlam, A., Melike, T. A. Ş., & BAYKAN, N. (2020). Geri Dönüştürülebilir Atıkların Materyallerine Göre Sınıflandırılması için Raspberry Pi Tabanlı Donanım Geliştirilmesi. Avrupa Bilim ve Teknoloji Dergisi, 30-38.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal

networks. Advances in neural information processing systems, 28, 91-99.

Shatnawi, A., Al-Bdour, G., Al-Qurran, R., & Al-Ayyoub, M. (2018, April). A comparative study of open source deep learning frameworks. In 2018 9th international conference on information and communication systems (icics) (pp. 72-77). IEEE.

Tensorflow. (2019). GitHub - tensorflow/models: Models and examples built with TensorFlow. 15 Mart 2021 tarihinde https://github.com/tensorflow/models adresinden erişildi.

TensorFlow. (2021). 15 Mart 2021 tarihinde https://www.tensorflow.org/ adresinden erişildi.

Kovalev, V., Kalinovsky, A., & Kovalev, S. (2016). Deep learning with theano, torch, caffe, tensorflow, and deeplearning4j: Which one is the best in speed and accuracy?.

Yang, G., Yang, J., Sheng, W., Junior, F. E. F., & Li, S. (2018). Convolutional neural network-based embarrassing situation detection under camera for social robot in smart homes. Sensors, 18(5), 1530.

Yapıcı, M. M., & Topaloğlu, N. Performance comparison of deep learning frameworks. Computers and Informatics, 1(1), 1-11.