Dört Ayaklı Bir Robot için YOLO Algoritması ile Yapay Zeka Görüntü İşleme

Artificial Intelligence Image Processing with YOLO Algorithm for a Quadruped Robot

Memduh Köse
Elektrik-Elekronik Mühendisliği
Kırşehir Ahi Evran Üniversitesi
Kırşehir, Türkiye
memduh.kose@ahievran.edu.tr

Bünyamin Mert
Elektrik-Elekronik Mühendisliği
Kırşehir Ahi Evran Üniversitesi
Kırşehir, Türkiye
mert.bunyamin@ogr.ahievran.edu.tr

Melikhan Kaygusuz
Elektrik-Elekronik Mühendisliği
Kırşehir Ahi Evran Üniversitesi
Kırşehir, Türkiye
kaygusuz.melikhan@ogr.ahievran.edu.tr

Hakkı Alparslan Ilgın
Elektrik-Elekronik Mühendisliği
Ankara Üniversitesi
Ankara, Türkiye
ilgin@eng.ankara.edu.tr

2025 33rd Signal Processing and Communications Applications Conference (SIU) | 979-8-3315-6655-5/25/$31.00 ©2025 IEEE | DOI: 10.1109/SIU66497.2025.11111751

***Özetçe*—Bu çalışmada, bir dört ayaklı (quadruped) robot projesi ele alınmış söz konusu robot sisteminde, Raspberry Pi 4 Model B ile uyumlu Raspberry Pi V4 kamera modülü kullanılarak nesne tespiti ve tanımlaması gerçekleştirilmiştir. Bu amaç doğrultusunda, YOLO (You Only Look Once) ailesinin YOLOv3 versiyonu tercih edilmiş; Joseph Redmon tarafından geliştirilen ve CUDA platformu üzerinde çalışan Darknet çerçevesi ile entegre edilerek tek bir yapay zeka ağı altında birleştirilmiştir. Dört ayaklı robotlar, hareket esnasında doğal ve yapay pek çok engelle karşı karşıya kalmaktadır. Bunlar arasında yükseklik farkları, arazi koşulları, mesafe değişkenleri, katı ve sıvı cisimler ile canlılar yer almaktadır. Bu araştırma kapsamında, robotun yürüme sürecinde karşılaşabileceği bu tür engellerin tanınması ve konumlandırılması amacıyla YOLO modeli ve geliştirilen yapay zeka algoritmaları kullanılmıştır. Nihai hedefi, nesneleri ve insanları tespit edebilen, konumlandırabilen ve çevresel faktörleri analiz edebilen bir sistem geliştirmek olan çalışmada bu doğrultuda, dört ayaklı robotlarda görüntü işlemeye yönelik araştırmalar gerçekleştirilmiş ve YOLOv3 ile nesne tespiti başarıyla sağlanmıştır.**

Anahtar Kelimeler — Yapay Zeka, Dört ayaklı robot, YOLO V3, Raspberry Pi 4 Model B, DarkNET

***Abstract*— This study focuses on a quadruped robot project where object detection and identification are carried out using a Raspberry Pi 4 Model B and the Raspberry Pi V4 camera module. To achieve this, the YOLO (You Only Look Once) family's YOLOv3 version was chosen, and it was integrated with the Darknet framework, developed by Joseph Redmon, which operates on the CUDA platform, under a single artificial intelligence network. Quadruped robots encounter various natural and artificial obstacles while moving, such as height differences, terrain conditions, distance variations, solid and liquid objects, as well as living beings. In this research, YOLO model and developed AI algorithms were used to recognize and locate these types of obstacles the robot may face during its walking process. The ultimate goal of the study is to develop a system that can detect and position objects and humans and analyze environmental factors. To this end, research on image processing for quadruped robots was conducted, and object detection was successfully achieved using YOLOv3.**

Keywords —Artificial Intelligence, Quadruped robot, YOLO V3, Raspberry Pi 4 Model B, DarkNET

I. GİRİŞ

Son yıllarda, teknolojinin hızla ilerlemesiyle birlikte hayatın neredeyse her alanında önemli değişimler ve gelişmeler yaşanmaktadır. Yapay zeka ve robotik sistemler, baş döndürücü bir hızla evrim geçirerek endüstriden gündelik yaşama kadar pek çok alanda kendine sağlam bir yer edinmiştir. Robotlar, kendilerine verilen görevleri yüksek hız ve verimlilikle yerine getirebildikleri için, robotik mühendisliği giderek daha önemli bir disiplin haline gelmiştir [1]-[3]. Ayrıca, insanoğlunun doğadaki canlıları gözlemleyerek ve onlardan ilham alarak teknolojik yenilikler geliştirme eğiliminde olduğu da gözlemlenmektedir. Bu durum özellikle bacaklı robot teknolojileri alanında belirgin bir şekilde kendini göstermektedir. Dünya genelinde Boston Dynamics ile özdeşleşen dört ayaklı robot tasarımları, yazılım ve elektronik endüstrisinde yaşanan ilerlemeler sayesinde önemli gelişim süreçlerinden geçmiştir. Son yıllarda, yapay zeka ve robotik teknolojilerin entegrasyonu üzerine yapılan farklı çalışmalar ve deneyler, dört ayaklı robotların yeteneklerini ve kullanım alanlarını daha da genişletmiştir. Dört ayaklı robotların, uzun yıllardır süregelen tasarım süreçleri ve üzerinde gerçekleştirilen araştırmalar neticesinde, görüntü işleme alanında tercih edilen öncü sistemlerden biri olduğu görülmektedir. Her ne kadar farklı modeller değişkenlik gösterse de, dört ayaklı robotlar genel olarak çeşitli arazi koşullarına uyum sağlayabilme, bacak yapıları sayesinde dengede kalabilme ve üstün manevra kabiliyeti sunma gibi avantajlara sahiptir. Bu temel yetenekleri sayesinde, gündelik yaşamdan sektörel uygulamalara kadar geniş bir kullanım yelpazesine sahiptirler. Olası kullanım alanları arasında şu uygulamalar öne çıkmaktadır:

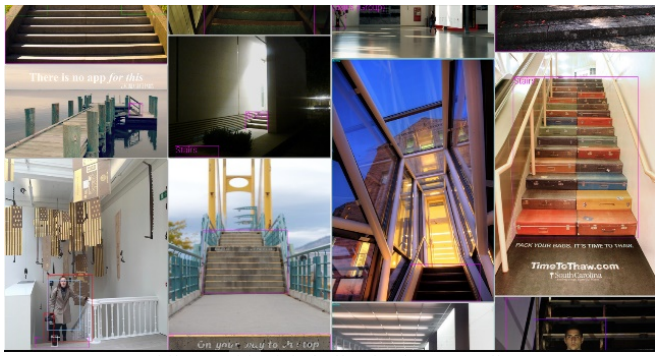
- Görme engelli bireyler için rehberlik sağlayarak çevresel algıyı desteklemek,
- Yangın, deprem, çığ gibi doğal afetlerde olay yerine hızlı müdahale etmek ve yaralıların tespitini gerçekleştirmek,
- Endüstriyel taşımacılık gibi lojistik süreçlerde görev almak.

Son yıllarda, görüntü işleme, derin öğrenme ve makine öğrenimi gibi alanlarda yaşanan büyük ilerlemeler, önceden eğitilmiş yapay zeka modellerinin belirli hedeflere yönelik tespit yapabilmesini mümkün kılmıştır. Bu bağlamda, özelleştirilmiş görevler için geliştirilen dört ayaklı robotların, kamera ve çeşitli sensörlerle donatılması gerekmektedir. Bunun yanı sıra, belirli amaçlara yönelik yapay zeka tabanlı görüntü işleme algoritmalarının optimize edilmesi büyük önem taşımaktadır. Bu çalışmada, popüler derin öğrenme tabanlı nesne tespit algoritmalarından biri olan YOLO (You Only Look Once) ailesinin YOLOv3 modeli önerilmektedir. Eğitilen bu modelin, Raspberry Pi 4 Model B üzerinde çalıştırılması planlanmış olup, performans analizleri de bu platform üzerinden gerçekleştirilmiştir. Çalışmadaki en temel hedef, YOLO algoritması kullanılarak geliştirilmiş dört ayaklı bir robotun, nesne tespit etme ve çevresini analiz etme yetisine sahip olmasını sağlamaktır [4-7].

II. YÖNTEM

A. Veri Setlerinin Hazırlanması

Bu çalışmada, YOLOv3 modelinin eğitimi için, Google Open Images platformundan uygun bir veri seti oluşturulmuştur. Modelin, dört ayaklı bir robot üzerinde çalışacağı göz önünde bulundurularak, robotun karşılaşabileceği çeşitli çevresel unsurların görselleri toplanmıştır. Bunlar arasında engebeli arazi yapıları, su birikintileri, merdivenler ve benzeri unsurlar yer almaktadır. Toplanan görsellerin etiketleme süreci için gerekli dosyalar indirilmiş ve her bir görselin hangi nesneye karşılık geldiğini belirlemek amacıyla Python dili kullanılarak özel bir kod geliştirilmiştir. Bu süreçte, her görselde yer alan nesnelerin tespit edilebilmesi adına, sınır kutuları ve ilgili sınıf etiketleri belirlenmiştir. Veri setinin eğitim sürecine uygun hale getirilmesi için 100 adet görselden oluşan bir veri kümesi oluşturulmuştur. Bu veri seti, modelin öğrenme sürecini optimize etmek adına %80'i (80 adet görüntü) eğitim, %20'si (20 adet görüntü) doğrulama olmak üzere iki ayrı klasöre (train ve valid) ayrılmıştır. Eğitilmiş model parametrelerinin kaydedilmesi için ise özel bir "backup" dizini oluşturulmuştur. Eğitim sürecine başlamadan önce, Darknet çerçevesinin eğitime uygun hale getirilmesi adına gerekli yapılandırma ayarları ve dosya yolları düzenlenmiştir. Algoritmada yapılan özelleştirmeler ve uygulanan matematiksel fonksiyonlar sayesinde, yalnızca 100 veri örneği ile etkili bir model eğitimi gerçekleştirmek hedeflenmiştir. Çalışmada esas amaç, sınırlı sayıda veriyle maksimum doğruluk oranına ulaşabilen yüksek verimli bir yapay zeka modelinin geliştirilmesidir. Bu çalışmada, dört ayaklı robotların karşılaştığı en önemli zorluklardan biri olan “merdiven problemi” üzerine odaklanılmıştır. Robotun merdivenleri tanıyıp doğru şekilde hareket edebilmesi için gerekli veri setleri oluşturulmuş ve eğitim sürecinde kullanılmıştır. Ancak burada dikkate alınması gereken temel zorluklardan biri, merdivenlerin farklı tasarımlara, eğim açılarının değişkenliğine ve yapısal çeşitliliğe sahip olmasıdır. Bu farklılıkların modele kazandırılabilmesi için geniş bir veri kümesi oluşturulmuş ve mümkün olduğunca çeşitli merdiven görselleri toplanmıştır. Çalışmamızda, başlangıç aşamasında “tek domain” olarak adlandırılan nesne grubuna odaklanılmış ve yalnızca merdiven kategorisine ait veriler kullanılmıştır. Gelecek çalışmalarda ise, bu veri grubunun daha geniş bir yapı içinde farklı alt kategorilerle zenginleştirilmesi planlanmaktadır. Veri setinin oluşturulması sırasında; merdivenlerin şekil, renk, boyut gibi fiziksel özellikleri, kamera ile olan mesafeleri, iç ve dış mekan görüntüleri gibi birçok değişken göz önünde bulundurulmuştur. Bu sayede modelin farklı çevresel koşullara uyum sağlayarak doğru nesne tespiti yapması amaçlanmıştır, Şekil 1. Toplanan görüntü verileri YOLO algoritmasına girdi olarak tanımlanmadan önce, bir dizi ön işleme aşamasından geçirilmiştir. Görüntüler, 416x416 piksel boyutlarına ölçeklendirilerek modele uygun hale getirilmiştir. Bunun yanı sıra, veri setinin Darknet çerçevesiyle uyumlu olması için gerekli olan stairs.data, class.names, train.txt, valid.txt gibi dosyalar oluşturulmuş ve düzenlenmiştir. Sonuç olarak, hazırlanan veri seti; görüntülerden ve bu görüntülere ait etiket dosyalarından oluşan eksiksiz bir yapı haline getirilmiştir. Modelin eğitimi sırasında algoritmaya veri girişlerinin doğru şekilde sağlanabilmesi için, nesne sınıfı sayısı, her sınıfa ait nesnelerin adları ve eğitim/test görüntülerinin konumlarını içeren bir yapılandırma dosyası hazırlanmıştır. Bilindiği üzere, YOLO algoritması öğrenme süreci boyunca sürekli olarak kendini yenileyerek performansını artıran bir sistemdir. Her öğrenim aşamasında, model kendi başarımını test görüntüleri üzerinde sınamakta ve hata oranını ortaya koymaktadır. Bu çalışma kapsamında, veri setinin %80’i eğitim için, %20’si ise test işlemleri için ayrılmıştır. Böylelikle, modelin eğitim süreci boyunca hem öğrenme hem de doğrulama aşamalarında etkin bir şekilde optimize edilmesi sağlanmıştır.



Şekil. 1. Eğitim için kullanılan bazı görseller.

B. Model Eğitimi

Veri toplama ve etiketleme aşamalarının tamamlanmasının ardından, YOLOv3 modelinin eğitim süreci başlatılmıştır. Bu süreçte Darknet çerçevesi kullanılmış ve model, güçlü bir donanıma sahip bir bilgisayar üzerinde eğitilmiştir. Eğitim işlemleri, Nvidia 1080 Ti ekran kartı ve 128 GB RAM ile donatılmış bir sistemde gerçekleştirilmiştir. Modelin verimli ve hızlı bir şekilde eğitilebilmesi için gerekli yazılımlar özenle seçilmiş ve sistem ile tam uyum içinde çalışacak şekilde konfigüre edilmiştir. Bu kapsamda, NVIDIA CUDA 12.8.0, NVIDIA cuDNN 9.7.1 ve OpenCV 4.11.0 sürümleri tercih edilmiştir. Seçim yapılırken, bu sürümlerin birbiriyle ve Visual Studio 2022 Code ortamıyla tam uyumluluk göstermesi esas alınmıştır. Gerekli kütüphaneler ve dosya uzantıları sistem değişkenlerine kaydedilmiş, ardından Darknet klasöründeki make dosyası düzenlenerek derleme işlemi gerçekleştirilmiştir. Make dosyası, eğitim sürecinde hangi bileşenlerin (CUDA,

979-8-3315-6655-5/25/$31.00 ©2024 IEEE

cuDNN, GPU ve OpenCV gibi) aktif olarak kullanılacağını belirlemektedir. Gerekli parametreler etkinleştirildikten sonra dosya kaydedilmiş ve Darknet derlenmiştir. Bunlara ek olarak, batch ve subdivision gibi parametreler de donanım kapasitesi göz önünde bulundurularak optimize edilmiştir. Eğitim sürecinde transfer öğrenme yöntemi uygulanmıştır. Daha önce büyük veri setleri ile eğitilmiş olan YOLOv3.weights dosyası, modelin başlangıç ağırlıkları olarak kullanılmış ve model, özel olarak toplanan veri kümesiyle yeniden eğitilmiştir. Eğitim aşamasına geçmeden önce, modelin konfigürasyon ayarları dikkatlice yapılandırılmıştır. YOLOv3.cfg dosyasında aşağıdaki temel parametreler, eğitim senaryosuna ve donanım kapasitesine uygun olarak düzenlenmiştir:

classes = (eğitilecek sınıf sayısı)

filtre sayısı=(classes + 5) * 3

max_batch = classes*2000

steps=(max_batch*%20),(max_batch%80)

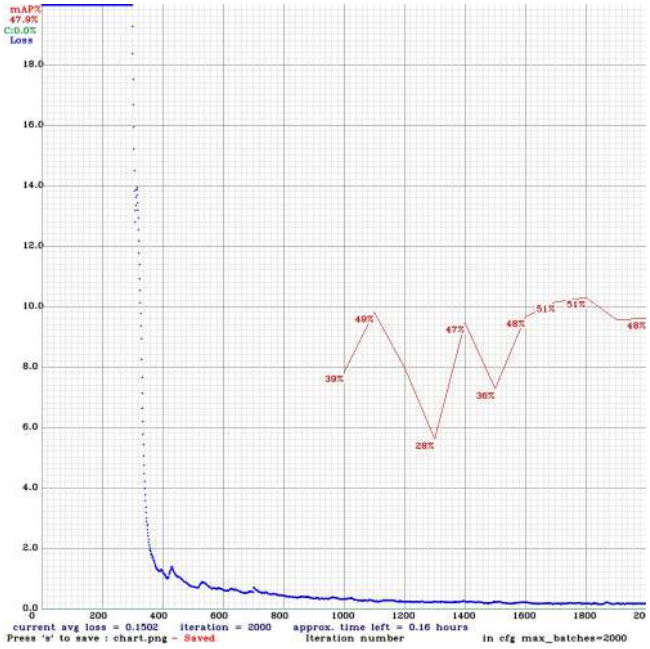
Bunlara ek olarak, batch ve subdivision gibi parametreler de donanım kapasitesi göz önünde bulundurularak optimize edilmiştir. Eğitim sürecinde transfer öğrenme yöntemi uygulanmıştır. Daha önce büyük veri setleri ile eğitilmiş olan YOLOv3.weights dosyası, modelin başlangıç ağırlıkları olarak kullanılmış ve model, özel olarak toplanan veri kümesiyle yeniden eğitilmiştir. Eğitim boyunca modelin performansı titizlikle takip edilmiş; doğruluk oranları ve kayıp değerleri (loss) gözlemlenmiştir. Loss değeri, modelin eğitim sırasında yaptığı hata miktarını gösteren kritik bir metriktir. Bunun yanı sıra, modelin nesne algılama başarısını değerlendirmede en yaygın kullanılan metriklerden biri olan mAP (Mean Average Precision) de hesaplanmıştır. Gerçekleştirilen deneysel çalışmada, mAP değeri %51 olarak elde edilmiştir (Şekil 2). Her ne kadar bu oran ideal bir doğruluk seviyesinin altında olsa da, veri kümesinin genişletilmesi ve modelin farklı parametrelerle yeniden eğitilmesi ile daha güvenilir ve yüksek doğruluklu sonuçlar elde edilmesi mümkündür. Tıpkı diğer yapay sinir ağı algoritmalarında olduğu gibi, YOLO algoritmasının da eğitilmesi kaçınılmaz bir zorunluluktur. Bu eğitim sürecinde, belirlenen hedef doğrultusunda en uygun veri kümeleri oluşturulmalı ve alaka düzeyine göre dikkatlice seçilmelidir [8].

III. Performans Derecelenrİrme Parametreleri

Eğitilen model, gerçek zamanlı nesne tespiti için Raspberry Pi ve bağlı bir kamera aracılığıyla test edilmiştir. Kamera görüntüleri OpenCV kullanılarak alınmış, her kare modelin giriş formatına uygun hale getirilerek işleme tabi tutulmuştur. YOLOv3 modelinin çıktıları, tespit edilen nesnelerin konumlarını ve hangi sınıfa ait olduklarını belirterek ekranda görselleştirilmiştir. Gerçek zamanlı algılama sürecinde, Raspberry Pi'nin işlem gücü göz önünde bulundurularak çeşitli eniyilemeler uygulanmıştır. Modelin hızını artırmak için giriş görüntüleri sıkıştırılmış, yalnızca gerekli katmanlar kullanılarak çıkarım süreci hızlandırılmıştır. Bunun yanı sıra, belirli bir eşik değerinin altındaki tahminler filtrelenerek gereksiz hesaplama yükü azaltılmıştır. Modelin başarımını değerlendirmek amacıyla Mean Average Precision (mAP), kesinlik (precision), anma (recall) ve F1 skoru gibi ölçütler kullanılmıştır. Nesne tespiti modellerinin doğruluk seviyesini belirlemek için True Positive (TP), False Positive (FP) ve False Negative (FN) gibi parametreler hesaplanarak detaylı analizler gerçekleştirilmiştir. Bir nesnenin doğru şekilde tespit edilip edilmediğini belirlemek için Intersection over Union (IoU) metriği kullanılmıştır. Eğer IoU ≥ 0.5 ise, tespit edilen nesne doğru kabul edilir ve True Positive (TP) olarak değerlendirilir. IoU ≤ 0.5 veya aynı nesne için birden fazla sınırlayıcı kutu oluşturulmuşsa, bu durum False Positive (FP) olarak kabul edilir. Öte yandan, tespit edilmesi gereken nesne gözden kaçırılmış veya yanlış sınıflandırılmışsa, False Negative (FN) olarak kaydedilir.

$$\text{Kesinlik} = \frac{TP}{TP + FP} \tag{1}$$

$$\text{Anma} = \frac{TP}{TP + FN} \tag{2}$$



Şekil. 2. Eğitim sırasında çıkarılan loss (mavi) ve mAP(kırmızı) grafikleri

Bu iki ölçütün birlikte değerlendirilmesi gerektiğinde, F1 Skoru hesaplanır. F1 skoru, kesinlik ve anmanın harmonik ortalaması olarak tanımlanır:

$$F_1 = 2 * \frac{\text{Kesinlik x Anma}}{\text{Kesinlik+Anma}} \tag{3}$$

Bunlara ek olarak, modelin başarısını daha detaylı analiz etmek amacıyla Average Precision (AP) ve Mean Average Precision (mAP) metrikleri kullanılmıştır. AP, modelin belirli bir sınıfa ait nesneleri ne kadar başarılı tespit ettiğini ölçerken, mAP, tüm sınıflar için hesaplanan AP değerlerinin ortalaması olarak ifade edilir:

979-8-3315-6655-5/25/$31.00 ©2024 IEEE

$$AP = \int_0^1 P(r)dr \tag{4}$$

$$mAP = \frac{1}{N} = \sum_{i=1}^{N} AP_i \tag{5}$$

Burada P(r), kesinliğin anmaya göre değişim fonksiyonunu, N ise toplam sınıf sayısını ifade eder. Modelin sınıflandırma başarımını daha iyi anlamak için Receiver Operating Characteristic (ROC) eğrisi ve Alan Altındaki Eğri (AUC - Area Under Curve) analizleri gerçekleştirilmiştir. ROC eğrisi, modelin farklı eşik değerlerinde TP ve FP oranlarının nasıl değiştiğini görselleştirirken, AUC değeri modelin genel ayırt ediciliğini ölçer:

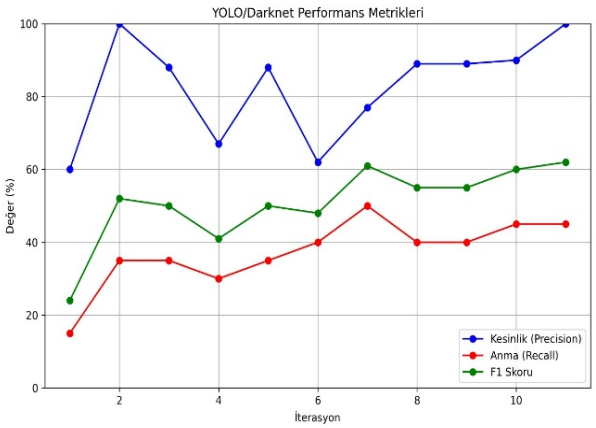
$$AUC = \int_{-\infty}^{\infty} TPR(FPR)d(FPR) \tag{6}$$

Burada TPR (True Positive Rate) ve FPR (False Positive Rate) şu şekilde hesaplanır:

$$TPR = \frac{TP}{TP + FN} \tag{7}$$

$$FPR = \frac{FP}{FP + TN} \tag{8}$$

Ek olarak, IoU metriği farklı eşik değerleriyle incelenmiş, modelin yüksek doğruluk gerektiren senaryolardaki başarımı analiz edilmiştir. IoU = 0.5 yerine IoU = 0.75 gibi daha sıkı bir eşik değeri kullanıldığında modelin doğruluk oranındaki değişimler detaylı şekilde değerlendirilmiştir. Tüm bu metriklerin birlikte ele alınması, modelin genel doğruluk seviyesinin ve farklı senaryolardaki performansının kapsamlı bir şekilde analiz edilmesini sağlamaktadır.



Şekil. 3. Kesinlik(Precision), Anma(Recall), F1 skor grafiği

IV. SONUÇ

Bu çalışmada, YOLOv3 modeli kullanılarak dört ayaklı bir robot için nesne tespiti gerçekleştirilmiştir. Modelin eğitimi, özel olarak toplanan ve etiketlenen veri seti ile tamamlanmış ve ardından gerçek zamanlı testler uygulanmıştır. Yapılan deneyler sonucunda, modelin belirlenen nesneleri başarıyla tespit edebildiği gözlemlenmiştir. Ancak, Raspberry Pi gibi sınırlı işlem gücüne sahip cihazlarda YOLOv3 modelinin çalıştırılması sırasında performans kısıtlamaları yaşandığı tespit edilmiştir. Modelin hızını artırmak amacıyla çeşitli eniyileme teknikleri uygulanmış olsa da, donanım kapasitesinin sınırlı olması nedeniyle bazı gecikmeler meydana gelmiştir. İleri çalışmalarda, modelin daha az işlem yükü gerektiren versiyonları (YOLOv4-Tiny, YOLO-Nano gibi) üzerinde testler yapılması ve eniyileme süreçlerinin derinleştirilmesi hedeflenmektedir. Ayrıca, modelin robotun hareket mekanizmasıyla entegre edilerek nesne tespiti sonrası karar alma süreçlerine katkı sağlaması planlanmaktadır. Bu sayede, robotun karmaşık ortamlarda daha etkin bir şekilde hareket edebilmesi ve engelleri algılayarak uygun manevralar gerçekleştirmesi amaçlanmaktadır.

KAYNAKLAR

[1] Liu, S., Zhang, T., Ji, H. and Wang L. “A Novel YOLOv8-pose-based Algorithm for Abnormal Behavior Analysis of Quadruped Robots” , *Proceedings of 2024 IEEE 13th Data Driven Control and Learning Systems Conference*, 2024.

[2] Amperawan, A., Andika, D., Anisah, M., Rasyad, S. And Handayani, P. “Confusıon Matrıx Usıng YOLO V3-Tıny On Quadruped Robot Based Raspberry Pı 3b+”, *Proceedings of the 7th FIRST 2023 International Conference on Global Innovations* (FIRST-ESCSI 2023), 549-562, 2024.

[3] Firdausi, R. M. A. R., Anifah, L. and Mon, A. A. “Surface Detectıon For Quadruped Robot Usıng YOLO-V3 Tıny”, *Journal of Intelligent System and Telecommunications*, 13-24, 2024.

[4] Ağdaş, M. T. ve Gülseçen, S. “Güvenlik Kameralarında Otomatik Silah ve Bıçak Tespit Sistemi: Karşılaştırmalı YOLO Modelleri” *European Journal of Science and Technology* , No. 41, 16-22, 2022.

[5] Havuç, E., Alpak, Ş., Çakırel, G. Ve Baran, M. K. “Derin Öğrenme Vasıtasıyla Masa Tenisi Topu Takibi”, *European Journal of Science and Technology* ,No. 27, 629-635, 2021.

[6] Nasrullah, M. D. ve Diker, A. “Derin Öğrenme ve YOLO Algoritmaları ile Nesne Tespiti ve Şerit Belirleme” , *2nd Internatıonal Conference On Intellıgent Transportatıon Systems*, 2021.

[7] Bayram, A. F., & Nabiyev, V. “Detection of hidden camouflaged tanks based on deep learning: comparative analysis of state-of-the-art YOLO networks” , *Gümüşhane Üniversitesi Fen Bilimleri Dergisi*, 13(4), 1082-1093, 2023.

[8] Özel, M. A., Baysal, S. S. ve Şahin, M. “Derin Öğrenme Algoritması (YOLO) ile Dinamik Test Süresince Süspansiyon Parçalarında Çatlak Tespiti”, *Avrupa Bilim Ve Teknoloji Dergisi* (26), 1-5, 2021.

979-8-3315-6655-5/25/$31.00 ©2024 IEEE