**T.C.**

**SAKARYA ÜNİVERSİTESİ**

**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**

**BSM 401 BİLGİSAYAR MÜHENDİSLİĞİ TASARIMI**

Acil Durum Araçları İçin Akıllı Trafik Kontrolü: Görüntü İşleme ve Mikrodenetleyici Kullanarak Trafik Işıklarının Dinamik Yönetimi

B201210373- Fatma GÜNER

|  |  |  |
| --- | --- | --- |
| Fakülte Anabilim Dalı  Tez Danışmanı | :  : | BİLGİSAYAR MÜHENDİSLİĞİ  Dr. Öğr. Üyesi Hüseyin DEMİRCİ |

2023-2024 Bahar Dönem

# ÖNSÖZ

Tez çalışmamın planlanmasında, araştırılmasında, yürütülmesinde ve oluşumunda ilgi ve desteğini esirgemeyen; engin bilgi ve tecrübelerinden yararlandığımız, yönlendirme ve bilgilendirmeleriyle çalışmamızı bilimsel temeller ışığında şekillendiren sayın hocamız Dr. Öğr. Üyesi Hüseyin DEMİRCİ’ye sonsuz teşekkürlerimi sunarım.

Fatma GÜNER

Sakarya 2024

# İÇİNDEKİLER

[ÖNSÖZ ii](#_Toc167970785)

[İÇİNDEKİLER iii](#_Toc167970786)

[ŞEKİLLLER LİSTESİ iv](#_Toc167970787)

[ÖZET v](#_Toc167970788)

[BÖLÜM 1: GİRİŞ 6](#_Toc167970789)

[1.1. Kuralların Belirlenmesi 2](#_Toc167970790)

[1.2. Projede Kullanılan Teknolojiler ve Araçlar 3](#_Toc167970791)

[BÖLÜM 2: 5](#_Toc167970792)

[LİTERATÜR TARAMASI 5](#_Toc167970793)

[BÖLÜM 3: 6](#_Toc167970794)

[PROJENİN GERÇEKLEMESİ 6](#_Toc167970795)

[3.1. VERİ SETİ OLUŞTURMA 6](#_Toc167970796)

[3.1.1 Veri seti nedir? 6](#_Toc167970797)

[3.1.2. Bu projede veri seti kullanımı 6](#_Toc167970798)

[3.2. Model Eğitimi 9](#_Toc167970799)

[3.2.1. Projede YOLO kullanımı 9](#_Toc167970800)

[3.2.2. Augmentation işlemi 10](#_Toc167970801)

[3.2.3. GPU kullanımı 10](#_Toc167970802)

[3.2.4. YOLOv8 modelini yapılandırma 11](#_Toc167970803)

[3.2.5. Gerekli Kütüphaneleri İçe Aktarma 12](#_Toc167970804)

[3.2.6. Veri Seti Yapılandırma Süreci 12](#_Toc167970805)

[3.2.7. Modelin Eğitimi Aşaması 13](#_Toc167970806)

[3.2.8. Eğitim Süreci Performans Değerlendirmesi 15](#_Toc167970807)

[BÖLÜM 4: 19](#_Toc167970808)

[DENEY DÜZENEĞİ VE TRAFİK IŞIK DEVRESİ 19](#_Toc167970809)

[4.1. Simülasyonun İşlevi 20](#_Toc167970810)

[4.2. Entegrasyon: 20](#_Toc167970811)

[4.3. Donanım Gereksinimleri 20](#_Toc167970812)

[4.4. Trafik Işığı Devresi Kurulumu 20](#_Toc167970813)

[4.5. Arduino Kodu Yazma ve Yükleme 21](#_Toc167970814)

[BÖLÜM 5: 25](#_Toc167970815)

[PROJE ENTEGRASYONU VE KODLAR 25](#_Toc167970816)

[5.1 Proje Entegrasyonu 25](#_Toc167970817)

[5.1.1 Görüntü işleme ve nesne tespiti 25](#_Toc167970818)

[5.1.2 Arduino ile trafik ışığı kontrolü 25](#_Toc167970819)

[5.1.3 Tinkercad ile devre simülasyonu 26](#_Toc167970820)

[5.2. Kodlar 26](#_Toc167970821)

[BÖLÜM 6: 28](#_Toc167970822)

[SONUÇ VE ÖNERİLER 28](#_Toc167970823)

[6.1 Sonuçlar 28](#_Toc167970824)

[6.2 Öneriler 28](#_Toc167970825)

[KAYNAKÇA 32](#_Toc167970826)

# ŞEKİLLLER LİSTESİ

Şekil 1. Ambulans Veri Seti İçin Örnek Birkaç Görsel 6

Şekil 2. Veri Setinde Etiketlenmiş Görsel Örnekler 7

Şekil 3. Verilerin Train- Valid- Test Yüzdelik Ayrımları 8

Şekil 4. Projede Yayılan Augmentation İşlemi 9

Şekil 5. Hızlandırma Anlamında GPU Kullanabiliyoruz 10

Şekil 6. Colab Ortamında YOLOv8 Kütüphanesini Kurup Yapılandırma Aşaması 11

Şekil 7. Gerekli Kütüphaneleri İçe Aktarmaktadır 12

Şekil 8. Veri Setini İndirme ve Yapılandırma Sürecini Otomatikleştirir 13

Şekil 9.YOLOv8 Modelini Belirli Parametrelerle Eğitmek İçin Kullanılır 14

Şekil 10.Belirli Bir Dizindeki Dosya ve Alt Dizinlerin Listesini Gösterir 15

Şekil 11. Confusion Matrix Grafik Modeli 16

Şekil 12. Performans Metriklerini ve Kayıplarını (loss) Görselleştirme 17

Şekil 13. Model Performans Sonuçları 18

Şekil 14. Proje Sanal Çıktıları Elde Etmek İçin Trafik Işık Devresi 21

Şekil 15. Devrenin Şematik Görünümü 22

# ÖZET

Anahtar kelimeler: Yapay Zekâ, YOLO Projesi, Nesne Tanıma

Bu proje, belirli durumlarda trafik ışıklarının nasıl ayarlanacağına dair kararları belirlemek için bir yöntem geliştirmeyi amaçlamaktadır. Öncelikle, acil durum araçları; ambulans ve itfaiye, diğer araçlar olmak üzere üç ana grup aracı tanımlayan bir sınıflandırma yapılır. Acil durum araçlarının öncelikli geçiş hakkına sahip olduğu durumlar için belirli kurallar belirlenir. Örneğin, acil durum araçlarının siren açıkken ve trafik ışığı kırmızıyken öncelikli geçiş hakkına sahip olduğu durumlar belirlenir.

Projede, acil durum araçlarının siren durumu ve trafik ışığı durumu gibi girdi değişkenleri kullanılarak bir model oluşturulur. Bu model, belirli bir durumda trafik ışıklarının nasıl ayarlanacağına dair kararlar almak için kullanılır. Örneğin, acil durum aracının varlığına ve siren durumuna göre trafik ışığının yeşil ışığa dönüp dönmemesi gibi kararlar bu model üzerinden verilir. Projede ayrıca, güvenlik ve trafik düzeni için ek kuralların veya istisnaların dahil edilebileceği belirtilir. Bu proje, görüntü işleme ve yapay zeka tekniklerini kullanarak trafik yönetiminde etkili bir yaklaşım sunmayı hedeflemektedir.

# BÖLÜM 1: GİRİŞ

Acil durumlar, zamanın kritik öneme sahip olduğu durumlardır. Bu durumlarda, hızla yardıma giden ambulans ve itfaiye araçlarının olay yerine ulaşması hayati önem taşır. Ancak, zamanın bu kadar kıymetli olduğu durumlarda dahi, trafikte beklemek zorunda kalan acil durum araçlarıyla karşılaşmak oldukça mümkündür. Trafik ışıklarında veya yoğun trafikte geçen her saniye, insan hayatının kurtarılması veya felaketlerin önlenmesi açısından kritik bir kayıptır. Bu nedenle, acil durum araçlarının ulaşımında her türlü engelin minimize edilmesi ve en etkin biçimde kullanılması gerekmektedir. Zira, hızlı müdahale zamanı saniyelerle yarışırken, trafikteki her türlü gecikme, potansiyel olarak hayat kurtarılacak bir fırsatın kaçırılmasına veya ciddi sonuçlar doğurabilecek bir krizin büyümesine sebep olabilir.[1] Bu bağlamda, acil durum araçlarının geçiş üstünlüğü ve trafikteki hareket kabiliyeti, zamanı en etkin şekilde kullanarak insan hayatını korumak ve felaketleri minimize etmek adına kritik bir rol oynamaktadır.

Acil durum araçlarının geçişi sırasında, ilgili yolun trafik ışığı yeşil yanar ve bu durum geçiş tamamlanana kadar devam eder. Ancak, acil durum araçlarının geçişine öncelik tanınabilmesi için belirli koşulların karşılanması gereklidir. Öncelikle, aracın acil duruma müdahale etmek üzere hareket ettiğini belirlemek için siren lambasının yanıp yanmadığı görsel işleme ve yapay zekâ yöntemleriyle tespit edilmelidir. Ambulanslar, polis araçları ve itfaiye araçları gibi acil durum araçlarının ortak özelliği, acil durum anlamını belirleyen siren ışıkları ve siren sesidir.[1]

Bu bağlamda, belirli bir trafik kavşağında veya ışıklı bir yolda trafik akışıyla ilgili veri toplandı. Acil durum araçlarının geçiş sıklığı incelendi ve şehir hastanelerinin veya araştırma hastanelerinin yakınındaki kavşaklarda ve ışıklarda genel olarak bir yoğunluk tespit edildi. Bu veriler ışığında, belirli bölgelerin gözlemlenerek analiz edilmesi kararlaştırıldı.

Daha sonra, toplanan verileri kullanarak bu bulanık mantık modeli geliştirildi. Bu modelin amacı, belirli bir durumda trafik ışıklarının nasıl ayarlanacağına dair kararlar almak için kullanılmaktır. Acil durum araçlarının varlığı, trafik ışıklarının doğrudan buna göre ayarlanmayacağını; ancak acil durum araçlarının sireninin aktif olduğu durumlarda, yani bir önemli vaka veya olay sırasında, ışıkların bu duruma göre ayarlanması gerektiği bulanık mantık kuralıyla belirlenir. Bu doğrultuda, kararlar alınarak trafik ışığı durumu belirlenir.

## 1.1. Kuralların Belirlenmesi

1. Eğer araç acil durum aracı ise;

* Eğer siren açıksa ve ışık kırmızı yanıyorsa sinyalizasyonda yeşil ışığa döndür.
* Eğer siren kapalıysa ışık fark etmeksizin normal sinyalizasyon devam eder.

1. Eğer araç binek araç ise;

* Eğer ışık yeşil yanıyorsa, normal sinyalizasyon devam eder.
* Eğer ışık kırmızı veya sarı yanıyorsa, normal sinyalizasyon devam eder ve aracın geçmesi yasaktır.

1. Eğer araç ticari araç ise;

* Eğer ışık yeşil yanıyorsa, normal sinyalizasyon devam eder.
* Eğer ışık kırmızı veya sarı yanıyorsa, normal sinyalizasyon devam eder ve aracın geçmesi yasaktır.

Bu kurallar, acil durum araçlarının öncelikli geçiş hakkına sahip olduğu durumları belirtmektedir. Ancak, güvenlik ve trafik düzeni için ek kurallar veya istisnalar da dahil edilebilir. Bu kuralları bir algoritma veya karar ağacı şeklinde programlayıp, görüntü işleme ve yapay zekâ teknikleri kullanarak uygulayacağım.

## 1.2. Projede Kullanılan Teknolojiler ve Araçlar

Bu proje, trafik ışıklarının acil durum araçlarına öncelik verecek şekilde dinamik olarak kontrol edilmesi amacıyla çeşitli teknolojiler ve araçlar kullanarak gerçekleştirilmiştir. Projede kullanılan başlıca teknolojiler ve araçlar aşağıda detaylı olarak açıklanmıştır.

**1. Python Programlama Dili**

Python, bu projenin temel programlama dili olarak kullanılmıştır. Python, geniş kütüphane desteği ve kolay okunabilir kod yapısı sayesinde veri işleme, makine öğrenimi ve mikrodenetleyici kontrolü gibi çok çeşitli görevler için idealdir. Projede, görüntü işleme ve model eğitimi için Python kullanılmıştır.

**2. PyCharm**

PyCharm, projenin geliştirilmesi için kullanılan entegre geliştirme ortamıdır (IDE). PyCharm, Python projelerini yönetmek ve kod yazmak için güçlü araçlar sunar. Bu proje kapsamında PyCharm, YOLOv8 modelinin entegrasyonu ve görüntü işleme kodlarının geliştirilmesi için kullanılmıştır.

**3. YOLOv8 (You Only Look Once)**

YOLOv8, gerçek zamanlı nesne tespiti için kullanılan bir derin öğrenme modelidir. Bu projede, acil durum araçlarını (ambulans ve itfaiye) trafikte tespit etmek için YOLOv8 modeli kullanılmıştır. YOLOv8, hızlı ve yüksek doğrulukla nesne tespiti yapabilmesi sayesinde bu tür uygulamalar için idealdir. Model, Google Colab üzerinde eğitilmiş ve PyCharm projesinde kullanılmıştır.

**4. OpenCV**

OpenCV (Open Source Computer Vision Library), görüntü işleme ve bilgisayarla görme görevleri için kullanılan bir kütüphanedir. Bu projede OpenCV, video akışını almak, YOLOv8 modelini kullanarak nesne tespiti yapmak ve sonuçları görselleştirmek için kullanılmıştır.

**5. Google Colab**

Google Colab, Python kodlarının bulut tabanlı olarak çalıştırılmasını sağlayan bir platformdur. Bu proje kapsamında, YOLOv8 modelinin eğitimi Google Colab üzerinde gerçekleştirilmiştir. Colab, yüksek hesaplama gücü ve GPU desteği ile model eğitimi için uygun bir ortam sağlar.

**6. Roboflow**

Roboflow, görüntü işleme ve makine öğrenimi projeleri için veri setleri oluşturmayı, etiketlemeyi ve yönetmeyi sağlayan bir platformdur. Bu projede, acil durum araçlarına ait veri setleri Roboflow üzerinde oluşturulmuş ve etiketlenmiştir. Veri setleri, YOLOv8 formatında indirilip model eğitimi için kullanılmıştır.

**7. Arduino**

Arduino, mikrodenetleyici projelerinde yaygın olarak kullanılan açık kaynaklı bir platformdur. Bu projede, trafik ışıklarının kontrolü için bir Arduino UNO mikrodenetleyici kullanılmıştır. Arduino, gelen acil durum sinyallerini alıp trafik ışıklarının yeşile dönmesini sağlayacak şekilde programlanmıştır.

**8. Tinkercad Circuits**

Tinkercad Circuits, elektronik devrelerin simülasyonu ve tasarımı için kullanılan bir araçtır. Bu proje kapsamında, trafik ışığı devresinin sanal simülasyonu Tinkercad Circuits kullanılarak oluşturulmuştur. Devre, Arduino ile entegre edilerek trafik ışıklarının doğru şekilde çalışması sağlanmıştır.

**9. PySerial**

PySerial, Python ile seri haberleşmeyi sağlayan bir kütüphanedir. Bu projede, Python ile Arduino arasındaki iletişimi sağlamak için PySerial kullanılmıştır. Acil durum aracı tespit edildiğinde, Python kodu üzerinden Arduino'ya sinyal gönderilerek trafik ışıkları kontrol edilmiştir.

Bu projede kullanılan teknolojiler ve araçlar, modern trafik yönetim sistemlerinin dinamik ve akıllı hale getirilmesinde önemli rol oynar. Python ve PyCharm, gelişmiş kodlama ve proje yönetimi sağlar. YOLOv8 ve OpenCV, yüksek doğrulukla nesne tespiti yaparak acil durum araçlarını tanımlar. Arduino ve Tinkercad Circuits, fiziksel trafik ışıklarının simülasyonunu ve kontrolünü gerçekleştirir. Bu bileşenlerin entegrasyonu, acil durum araçlarının geçişini hızlandırarak trafik güvenliğini artıran etkili bir çözüm sunar.

# BÖLÜM 2:

# LİTERATÜR TARAMASI

Trafikte özellikle son yıllarda yapay zeka teknikleri ve nesne tanıma uygulamalarının kullanımı artmıştır. Bu alandaki çalışmalar, trafik akışının verimliliğini artırmak, kazaları en aza düşürmek ve acil durumlar için hızlı ve etkili çözümler sağlamak gibi önemli amaçlar için yapılmaktadır.[3]

Trafik yönetiminde yapay zeka tekniklerinin kullanımıyla ilgili literatürde çeşitli örnekler bulunmaktadır. Örneğin; Chen ve arkadaşı (2017), kavşaklardaki araç trafiğini yönetmek için derin öğrenme tabanlı bir yöntem önermişlerdir. Bu çalışmada, nesne tanıma algoritmaları kullanılarak trafik yoğunluğunu analiz eden bir sistem geliştirilmiştir.[12]

Acil durum araçlarının geçişiyle ilgili bir örnek ise Türkiye’de halihazırda öncelikli araç geçiş sistemi olarak hizmet veren bir teknoloji şirketidir. Bu şirket; ambulans ve itfaiye gibi geçiş üstünlüğüne sahip araçların trafikte zaman kaybetmeden en hızlı biçimde hedefine varmasına yardımcı bir uygulamadır. Geçiş üstünlüğü olan araçlara takılan bir GPS ile araçların konumlarını gerçek zamanlı takip etmekte ve bu araçlar herhangi bir sinyalize kavşağa yaklaştığı anda aracın ilerleyiş istikametine göre yeşil ışık yanmasını sağlamaktadır. Bu yapmak istediğim projeyle amaç olarak aynıdır fakat farklı bir uygulamayla soruna çözüm geliştirilmiştir. [13]

Literatürdeki çalışmaların çoğu genellikle belirli senaryolara veya kısıtlı veri setlerine dayanmaktadır. Bu nedenle, geniş kapsamlı ve gerçek dünya verilerine dayalı çalışmalara olan ihtiyaç açıktır.

# BÖLÜM 3:

# PROJENİN GERÇEKLEMESİ

**Acil Durum Araçlarının Tanınması İçin Görüntü İşleme ve Makine Öğrenimi**

## 3.1. VERİ SETİ OLUŞTURMA

### 3.1.1 Veri seti nedir?

Belirli bir amaç için toplanmış ve organize edilmiş veriler bütününe denir. Veri setleri, makine öğrenimi modellerini eğitmek, test etmek ve doğrulamak için kullanılır. Veri setleri, çeşitli formatlarda olabilmektedir ve genelde etiketlenmiş ya da etiketlenmemiş veriler içerir. Etiketlenmiş veri setleri, her bir veri noktasının doğru yanıtını veya kümesini içerir. Bu da denetimli öğrenme için gereklidir.

### 3.1.2. Bu projede veri seti kullanımı

Yapılan bu projede, acil durum araçlarının yani ambulans ve itfaiye araçlarının tespiti için kullanımı amaçlanmıştır. Bu tespiti gerçekleştirmek için makine öğrenimi ve görüntü işleme teknikleri kullanılacaktır.

İlk olarak bir model eğitimi gerçeklemesi amaçlanmış ve yapılmıştır. Bunun için görsellerin toplanması gerekmektedir. Ambulans ve itfaiye araçlarının farklı açılardan ve koşullardan çekilmiş çok sayıda görüntüsünün toplanması gerekmektedir. Bu toplanan veriler, farklı açılardan ve uzaklıklardan çekilmiş birçok görüntüyü içermelidir. Bu görsellere Google görsellerden ulaşıp ufak bir görsel veri kümesi oluşturdum.

metin, dış mekan, taşıt, araç, kara taşıtı içeren bir resim

Açıklama otomatik olarak oluşturuldutaşıt, araç, kara taşıtı, dış mekan, metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, taşıt, araç, kara taşıtı, dış mekan içeren bir resim

Açıklama otomatik olarak oluşturuldukara taşıtı, dış mekan, tekerlek, gece içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 1. Ambulans Veri Seti İçin Örnek Birkaç Görsel

Daha sonrasında bu veri setindeki verilerin etiketlemeleri doğru bir şekilde yapılmaktadır. Bu görüntülerde hangi aracın bulunduğunun yani ambulans veya itfaiye araçlarının belirtilmesi gereklidir. Örneğin, bir görüntüdeki aracın ambulans olduğu etiketlenmelidir.

Tüm bunlar tamamlandıktan sonra bu etiketlenmiş görüntüler, makine öğrenimi modelini eğitmek için kullanılır. Model, bu görselleri ve etiketlerini kullanarak acil durum araçlarını tanımayı öğrenir.

metin, ekran görüntüsü, grafik tasarım, tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 2. Veri Setinde Etiketlenmiş Görsel Örnekler

Bu veri setleri doğrulama seti ve test seti olarak ayrılır. Yani veri setinin bir kısmını doğrulama seti olarak bir diğer kısmını ize test seti olarak kullanmaktayız. Model doğrulama ve test aşamaları, makine öğrenimi projelerinin kritik bileşenleridir. Bu aşamalar, modelin eğitim sırasında öğrenilen bilgileri ne kadar iyi genelleştirebildiğini değerlendirmek için kullanılır.

• Model Doğrulama (Validation): Modelin eğitim sırasında nasıl performans gösterdiğini izlemek ve bu performansı optimize etmek amacındadır. Modelin aşırı öğrenme (overfitting) veya eksik öğrenme (underfitting) yapıp yapmadığını tespit etmesini sağlar. Ayrıca hiperparametre ayarlarını belirlemek ve modelin genel performansını iyileştirmesini de sağlamaktadır. Bu işlemde, veri seti genellikle eğitim seti (%70-80), doğrulama seti (%10-15) ve test seti (%10-15) olarak bölünmektedir.[8]

• Model Testi (Testing): Modelin gerçek dünya performansını değerlendirmek amacındadır. Modelin genelleme yeteneğini (unseen data) ve pratikte nasıl çalıştığını belirlemeyi sağlar. Modelin son haliyle ne kadar güvenilir oluğunu anlamayı sağlamaktadır.

metin, ekran görüntüsü, yazılım, bilgisayar simgesi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 3. Verilerin Train- Valid- Test Yüzdelik Ayrımları

Bu aşamalar, modelin güvenilir ve kullanışlı olmasını sağlamak için hayati öneme sahiptir.

Bu aşamada eğitilmiş modeli, doğrulama seti ile değerlendirildi ve modelin performansı optimize edildi. Model, test seti ile test edilerek gerçek dünya performansı değerlendirildi. Bu model, trafik ışıklarını kontrol ederek acil durum araçlarına öncelik tanıyan bir sistemin temelini oluşturur.

## 3.2. Model Eğitimi

Bu aşamada, YOLO (You Only Look Once) modelini kullanarak acil durum araçlarının tespiti için model eğitimi gerçekleştireceğiz. YOLO, gerçek zamanlı nesne tespiti için yaygın olarak kullanılan hızlı ve etkili bir derin öğrenme modelidir.

Bu projede YOLO modelini kullanmamın en önemli nedenlerinden biri, görüntüyü bir kez analiz ederek nesneleri tespit eder. Bu, nesne tespitini iki aşamalı yaklaşımlar (önce bölge önerisi, sonra sınıflandırma) yerine tek bir aşamada gerçekleştirir. Bu özelliği ayrıca hız açısından da avantaj sağlamaktadır.

### 3.2.1. Projede YOLO kullanımı

YOLO, bir görüntüyü grid'lere böler ve her grid hücresinden nesne tahminleri yapar. Her hücre, belirli sayıda bounding box ve her box için bir sınıf tahmini yapar. Model, bu tahminleri birleştirerek nihai nesne tespitlerini oluşturur.[3]

Bu projede kullanımı şu şekilde gerçeklenmektedir:

• İlk olarak ambulans ve itfaiye araçlarının görüntüleri toplanır ve etiketlenir. Daha sonra, etiketlenmiş veri seti kullanılarak YOLO modeli eğitilir. Model, acil durum araçlarını doğru bir şekilde tanımayı öğrenir. Veri seti kullanılarak model eğitilir ve doğrulama verileriyle performansı izlenir. Eğitimli model, canlı kamera görüntülerinde acil durum araçlarını tespit etmek için kullanılır. Son olarak da tespit edilen acil durum araçlarına göre trafik ışıkları dinamik olarak kontrol edilir.

• YOLO, gerçek zamanlı nesne tespiti için güçlü bir araçtır ve bu projede trafik ışıklarını kontrol eden bir sistemde acil durum araçlarını tespit etmek amacıyla kullanılmıştır. YOLO'nun hızlı ve doğru nesne tespiti yetenekleri, bu tür uygulamalarda kritik öneme sahiptir. Model, trafik ışıklarına öncelik vererek acil durum araçlarının daha hızlı geçiş yapmasını sağlar, böylece trafik güvenliğini ve etkinliğini artırır.

* Formun Üstü

### 3.2.2. Augmentation işlemi

Veri etiketleme ve Train/Test Split işlemini tamamladıktan sonra, augentation işlemi gerçekleştirilecektir. Bu işlem veri çoğaltma olarak geçmektedir. Bu projede gri ölçekle çoğaltma (Grayscale) işlemi, bulanıklaştırma yaparak çoğaltma (Blur) işlemi ve gürültü ekleme yaparak çoğaltma (Noise) işlemi yapılmıştır. [8]

metin, yazı tipi, ekran görüntüsü, beyaz içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 4. Projede Yayılan Augmentation İşlemi

YOLOv8 eğitimi, özellikle gerçek zamanlı nesne tespiti için güçlü bir araçtır. Google Colab kullanarak YOLOv8 modelini eğitmek, yüksek hesaplama gücüne erişim sağladığı için oldukça avantajlıdır. Veri setini doğru şekilde hazırlayıp modelin performansını izleyerek, nesne tespiti için güçlü bir model elde edilebilir.

Roboflow models yazarak Roboflow’un modeller sayfasına ulaşılabilmektedir. Buradan ‘YOLOv8 Object Detection’ kısmından Colab sayfasına ulaşabiliriz. Buradan kendi Drive’ımıza bunu kopyalayıp artık modelimizi eğitmeye buradan devam edeceğiz. Bundan sonra yapmamız gereken en önemli ve ilk işlem GPU’yu aktifleştirmeliyiz. [8]

### 3.2.3. GPU kullanımı

Model eğitimine başlamadan önce GPU’ya erişimimiz olduğundan emin olalım. Bunu yapmk için ‘nvidia-smi’ komutunu kullanabiliriz. [8]

metin, ekran görüntüsü, ekran, görüntüleme, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 5. Hızlandırma Anlamında GPU Kullanabiliyoruz

### 3.2.4. YOLOv8 modelini yapılandırma

Bu aşamadan sonra Google Colab ortamında YOLOv8 kütüphanesini kurup yapılandırırız. Bu adımların amacı, YOLOv8 modelini kullanmaya başlamadan önce gerekli kütüphaneleri kurmak ve ortamın doğru şekilde yapılandırıldığını doğrulamaktır. [8]

• Bu aşamada ilk olarak Ultralytics kütüphanesini kuracak kod yazılıdır. Ultralytics, YOLOv8 modelini içeren ve makine öğrenimi ile derin öğrenme modellerini kolayca kullanmanızı sağlayan bir Python kütüphanesidir.

• **import ultralytics:** Ultralytics kütüphanesini Python ortamına dahil eder, böylece kütüphanenin işlevlerini kullanabilirsiniz.

• **ultralytics.checks():** Bu işlev, ortamın doğru şekilde yapılandırıldığını ve gerekli bağımlılıkların doğru yüklendiğini kontrol eder. Ayrıca, GPU'nun mevcut olup olmadığını ve doğru çalışıp çalışmadığını doğrular.

metin, yazılım, multimedya yazılımı, grafik yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 6. Colab Ortamında YOLOv8 Kütüphanesini Kurup Yapılandırma Aşaması

### 3.2.5. Gerekli Kütüphaneleri İçe Aktarma

YOLO (You Only Look Once) modelini ve görüntüleme işlevlerini kullanmak için gerekli kütüphaneleri içe aktarmaktadır. Bu adım, YOLOv8 modelini eğitmek ve sonuçları görselleştirmek için gereklidir.

**‘ultralytics’** kütüphanesi YOLO sınıfını içe aktarır. YOLO sınıfı, YOLOv8 modelini oluşturmak, eğitmek ve tahminlerde bulunmak için gerekli fonksiyonları sağlar. Aşağıdaki kod, YOLO modeli ile nesne tespiti yapmak için kullanılır. Bu model, projenizde ambulans ve itfaiye araçlarını tespit etmek için eğitilecektir.

**IPython.display** modülü **display** ve **Image** işlevlerini içe aktarır. Bu işlevler, Colab veya Jupyter Notebook ortamında görüntülerin ve diğer çıktıların görselleştirilmesi için kullanılır. Modelin eğitim ve test sonuçlarını görselleştirmek için kullanılır. Modelin tahmin ettiği sonuçları ve görüntüleri notebook üzerinde göstermek için bu işlevlere ihtiyaç duyulur.

Bu kod parçası, YOLO modelini eğitmek ve bir görüntü üzerinde tahmin yaparak sonuçları görselleştirmek için kullanılabilir. Böylece, projede ambulans ve itfaiye araçlarının trafikteki tespiti ve izlenmesi için gerekli olan adımlar gerçekleştirilmiş olur.[8]

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 7. Gerekli Kütüphaneleri İçe Aktarmaktadır

### 3.2.6. Veri Seti Yapılandırma Süreci

Google Colab ortamında Roboflow'dan veri seti indirip YOLOv8 modeliyle kullanmak üzere hazırlamaktadır. Bu adım, modelinizi eğitmek için gerekli olan veri setini indirme ve yapılandırma sürecini otomatikleştirir.

Aşağıda yazılan bu kod, YOLOv8 modelini eğitmek için gerekli olan etiketlenmiş veri setini indirir. Bu veri seti, ambulans ve itfaiye araçlarının tespiti için kullanılacaktır. Roboflow API'sini kullanarak veri setini otomatik olarak indirmek ve yapılandırmak, manuel işlemleri ortadan kaldırır ve süreçleri hızlandırır. İndirilen veri seti, YOLOv8 formatında olduğundan, modelin eğitimi için doğrudan kullanılabilir hale gelir. Bu, veri hazırlama aşamasında zaman kazandırır ve hata riskini azaltır.

Yani kısaca bu kod bloğu, Roboflow üzerinden veri setini indirip hazırlayarak, YOLOv8 modelinin eğitimi için gerekli olan temel veri hazırlık işlemlerini gerçekleştirir. Bu sayede, modelin eğitimi için gerekli olan tüm veriler uygun formatta ve düzenli bir şekilde hazır hale getirilir. Bu adımlar, projenizin verimli ve doğru bir şekilde ilerlemesini sağlar.

**metin, ekran görüntüsü, yazılım, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu**

Şekil 8. Veri Setini İndirme ve Yapılandırma Sürecini Otomatikleştirir

### 3.2.7. Modelin Eğitimi Aşaması

Aşağıda Şekil 9’da verilen bu kodu, YOLOv8 modelini özelleştirilmiş (custom) veri setiyle eğitmek için kullanılır. Bu kodu kullanmamızın amacı, YOLOv8 modelini özelleştirilmiş veri setiyle eğitmektir. Bu komut, YOLOv8 modelini belirli parametrelerle eğitmek için kullanılır.[8]

**Komutun Parametreleri**

• **task=detect:** YOLOv8 modelinin tespit (detect) görevi için kullanılacağını belirtir.

• **mode=train:** Modelin eğitim modunda çalışacağını belirtir.

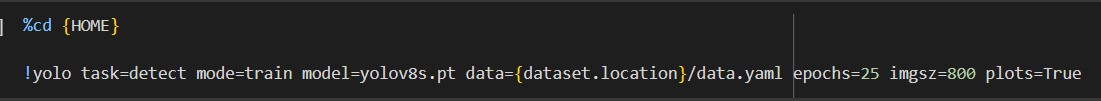
• **model=yolov8s.pt:** Kullanılacak modelin önceden eğitilmiş YOLOv8s ağırlıkları olduğunu belirtir.

• **data={dataset.location}/data.yaml:** Eğitim veri setinin yolunu belirten dosya. Bu dosya, veri setinin nasıl yapılandırıldığını ve nerede bulunduğunu belirtir.

• **epochs=25:** Modelin 25 epoch boyunca eğitileceğini belirtir. Epoch, veri setinin tamamının modele kaç kez gösterileceğini ifade eder.

• **imgsz=800:** Eğitim sırasında kullanılacak görüntülerin boyutunu (800x800 piksel) belirtir.

• **plots=True:** Eğitim süresince performans grafikleri ve diğer görselleştirmelerin oluşturulmasını sağlar.



Şekil 9.YOLOv8 Modelini Belirli Parametrelerle Eğitmek İçin Kullanılır

**Custom Training (Özelleştirilmiş Eğitim) Nedir ve Neden Yapıyoruz?**

• **Custom Training (Özelleştirilmiş Eğitim):** Makine öğrenimi modelini, spesifik bir veri setiyle eğitme sürecidir. Bu, modelin genel bir veri seti yerine, belirli bir kullanım senaryosu veya özel bir veri kümesi üzerinde eğitilmesi anlamına gelir.

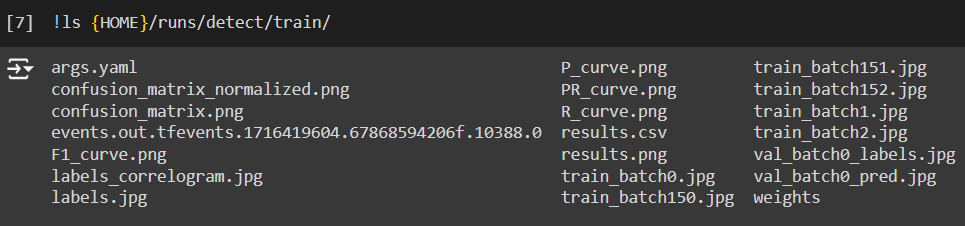
**• Neden Yapıyoruz:**

* **Özgünlük:** Projenizin spesifik gereksinimlerine uygun bir model elde etmek için. Örneğin, bu projede trafik kameralarından ambulans ve itfaiye araçlarını tespit etmek.
* **Doğruluk:** Özelleştirilmiş veri seti kullanarak modelin doğruluğunu artırmak. Genel veri setleri, spesifik senaryolar için her zaman en iyi performansı göstermez.
* **Performans:** Modelin performansını ve tespit doğruluğunu optimize etmek için belirli veri örnekleri üzerinde eğitim yapılır.

Kısacası Şekil 9’da verilen bu kod, YOLOv8 modelinizi özel bir veri setiyle eğitmek için kullanılır. Bu eğitim süreci, modelin belirli görevler (örneğin, ambulans ve itfaiye araçlarını tespit etme) için optimize edilmesini sağlar. Bu sayede, projede yüksek doğruluk ve performans elde edebiliriz. Custom training, modelin spesifik ihtiyaçlara göre uyarlanmasını ve daha iyi sonuçlar elde edilmesini sağlar. Bu kod çalıştığı zaman bize /runs/detect/train/weights klasöründe best.pt oluşturur. Bu ağırlık dosyasıdır, bu eğitilmiş hazır modelimiz olacaktır.

### 3.2.8. Eğitim Süreci Performans Değerlendirmesi

Şekil 10’da gösterilen bu kod, YOLOv8 modelinin eğitim sürecinde oluşturulan sonuç dosyalarını ve görsellerini görüntülemek için kullanılmıştır. Bu komut, belirli bir dizindeki dosya ve alt dizinlerin listesini gösterir. Model eğitimi sırasında oluşturulan tüm sonuç dosyalarını ve görselleri listeleyerek kullanıcıya eğitim sürecinin detaylarını sağlar. [8]



Şekil 10.Belirli Bir Dizindeki Dosya ve Alt Dizinlerin Listesini Gösterir

Bu dosyalar ve görseller, modelin eğitim sürecinde nasıl performans gösterdiğini izlememizi sağlar. Eğitim ve doğrulama kayıpları, doğruluk, hassasiyet, geri çağırma gibi metrikleri analiz ederek modelin iyileştirilmesi gereken alanlarını belirleyebiliriz. Ayrıca, karmaşıklık matrisleri ve örnek tahmin görüntüleri, modelin hangi sınıflarda daha fazla hata yaptığını görmemizi sağlar. Bu bilgi, eğitim veri setinin veya model hiper parametrelerinin nasıl iyileştirileceği konusunda yol gösterir. Eğitilmiş model ağırlıkları, modeli daha sonra yeniden kullanmak veya yeni veri setleri üzerinde ince ayar yapmak için saklanır.

Şekil 11’deki grafik, modelin performansını değerlendirmek için kullanılan bir karmaşıklık matrisini (confusion matrix) göstermektedir. Karmaşıklık matrisi, modelin gerçek sınıfları ile tahmin edilen sınıflar arasındaki ilişkiyi görselleştirir. Bu matris, modelin hangi sınıflarda doğru tahminler yaptığını ve hangi sınıflarda hata yaptığını anlamamıza yardımcı olur.

Her hücredeki sayı, belirli bir sınıf için doğru veya yanlış tahmin sayısını gösterir:[8]

* **True Positive (TP):** Doğru pozitifler (doğru tahmin edilen sınıflar).
* **False Positive (FP):** Yanlış pozitifler (yanlış tahmin edilen sınıflar).
* **False Negative (FN):** Yanlış negatifler (modelin tahmin edemediği gerçek sınıflar).

ekran görüntüsü, dikdörtgen, diyagram, kare içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 11. Confusion Matrix Grafik Modeli

**Matrisin Değerlendirilmesi**

**Ambulans:**

* **Doğru Tahminler (TP):** 7 (Model, 7 ambulansı doğru şekilde tespit etti)
* **Yanlış Pozitif (FP):** 1 (Model, 1 kez arkaplanı yanlışlıkla ambulans olarak tahmin etti)

**İtfaiye:**

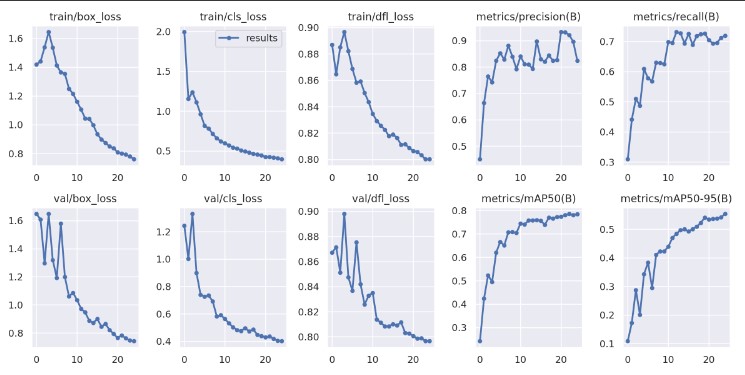
* **Doğru Tahminler (TP):** 8 (Model, 8 itfaiye aracını doğru şekilde tespit etti)
* **Yanlış Negatif (FN):** 4 (Model, 4 itfaiye aracını arkaplan olarak tahmin etti)
* **Yanlış Pozitif (FP):** 3 (Model, 3 arkaplanı yanlışlıkla itfaiye aracı olarak tahmin etti)

**Background (Arkaplan):**

* **Doğru Tahminler (TP):** 6 (Model, 6 arkaplanı doğru şekilde tespit etti)
* **Yanlış Negatif (FN):** 1 (Model, 1 ambulansı arkaplan olarak tahmin etti)

Bu grafik, modelin genel olarak iyi bir performans sergilediğini ancak bazı sınıflandırma hataları yaptığını gösterir. Daha fazla etiketlenmiş veri kullanarak modelin performansı artırılabilir. Ya da eğitim sürecinde kullanılan hiper parametreler optimize edilebilir. Diğer bir öneri ise, daha gelişmiş model mimarileri veya daha derin ağlar kullanılarak doğruluk artırılabilir.[8]

Bu tür bir karmaşıklık matrisi, modelin hangi alanlarda güçlü olduğunu ve hangi alanlarda iyileştirmeye ihtiyaç duyduğunu belirlemek için kritik öneme sahiptir.



Şekil 12. Performans Metriklerini ve Kayıplarını (loss) Görselleştirme

Şekil 12’deki grafikler, YOLOv8 modelinin eğitim sürecindeki çeşitli performans metriklerini ve kayıplarını (loss) görselleştirmektedir. Bu grafikler, modelin eğitim ve doğrulama aşamalarındaki başarısını ve nasıl geliştiğini anlamamıza yardımcı olur.

Grafikler, modelin eğitim sürecinde takip edilen farklı metrikleri ve kayıpları gösterir. Her bir grafik belirli bir metriği veya kaybı temsil eder ve epoch'lar boyunca nasıl değiştiğini gösterir. Her bir grafik, modelin belirli bir metriği veya kaybı açısından epoch'lar boyunca nasıl geliştiğini detaylandırır.

**Grafiklerin Genel Değerlendirmesi**

* **Kayıpların Azalması:** Eğitim ve doğrulama kayıplarının (loss) azalması, modelin eğitimi boyunca daha iyi performans gösterdiğini ve daha az hata yaptığını gösterir.
* **Metriklerin Artışı:** Doğruluk (precision), geri çağırma (recall) ve mAP değerlerinin artışı, modelin doğruluğunu ve genelleme yeteneğini artırdığını gösterir.
* **Eğitim Süreci:** Bu grafikler, modelin eğitim sürecindeki iyileşmeleri izlememize yardımcı olur ve modelin hangi alanlarda daha fazla eğitim gerektirdiğini belirlememize olanak tanır.

Bu grafikler, modelin eğitim sürecinde nasıl performans gösterdiğini detaylı bir şekilde incelememizi sağlar. Modelin hangi alanlarda iyi performans gösterdiğini ve hangi alanlarda iyileştirme yapılması gerektiğini anlamak için kritik öneme sahiptir.[8]

taşıt, araç, kara taşıtı, dış mekan, tekerlek içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 13. Model Performans Sonuçları

# BÖLÜM 4:

# DENEY DÜZENEĞİ VE TRAFİK IŞIK DEVRESİ

Bu projede Tinkercad ile bir trafik ışığı devresi simülasyonu oluşturuldu. Bunun amacı;

trafik yönetimini iyileştirmek ve acil durum araçlarına (ambulans, itfaiye) öncelik tanıyan dinamik bir trafik ışığı sistemi geliştirmektir. Bu simülasyon, gerçek dünyadaki trafik ışıklarının davranışını modelleyerek, acil durum aracı geçişlerinde yeşil ışık yanmasını sağlayacak şekilde tasarlanmıştır. Böylece, acil durum araçlarının hızlı ve güvenli bir şekilde hareket etmeleri sağlanarak, müdahale süreleri kısaltılabilir ve trafik kazaları azaltılabilir.

## 4.1. Simülasyonun İşlevi

Bu simülasyon, standart trafik ışığı döngüsünü ve acil durum sinyallerini yönetmek üzere programlanmıştır. Normal çalışma döngüsünde, trafik ışıkları belirli sürelerle yeşil, sarı ve kırmızı yanar. Ancak, görüntü işleme sistemi (kameralar ve YOLOv8 modeli kullanılarak) bir acil durum aracı tespit ettiğinde, Arduino'ya bir sinyal gönderilir ve bu sinyal devre tarafından algılanarak trafik ışıkları acil durum aracının geçişine izin verecek şekilde yeşile döner.

## 4.2. Entegrasyon:

* **Görüntü İşleme:** PyCharm üzerinde geliştirilen görüntü işleme kodu, kameradan alınan görüntülerde acil durum araçlarını tespit eder. Bu tespit sırasında, Arduino'ya seri port üzerinden bir sinyal gönderilir.
* **Arduino ve Simülasyon:** Tinkercad simülasyonunda, Arduino bu sinyali alarak LED'leri kontrol eder. Normal döngü durdurulur ve acil durum aracı geçene kadar yeşil ışık yanar.

## 4.3. Donanım Gereksinimleri

* **Arduino veya Raspberry Pi:** Mikrodenetleyici olarak kullanılacak.
* **LED'ler:** Yeşil, sarı ve kırmızı LED'ler.
* **Dirençler:** LED'leri korumak için uygun değerlerde dirençler.
* **Breadboard ve Jumper Kablolar:** Devreyi kurmak için.

## 4.4. Trafik Işığı Devresi Kurulumu

Bu projede, trafik ışığı devresi oluşturmanın amacı, trafik akışını kontrol etmek ve acil durum araçlarına öncelik tanıyan dinamik bir trafik ışığı sistemi geliştirmektir. Bu amaç doğrultusunda Tinkercad kullanılarak sanal bir devre oluşturulmuştur. Tinkercad, elektronik devreleri simüle etmek ve test etmek için mükemmel bir araçtır, özellikle Arduino gibi mikrodenetleyicilerle çalışırken.

İlk olarak, trafik ışığını temsil eden LED'ler (yeşil, sarı ve kırmızı) ve bu LED'leri kontrol eden Arduino UNO mikrodenetleyicisi kullanılmıştır. Her LED için uygun değerlerde dirençler kullanılmıştır. Dirençler, LED'lerin aşırı akım nedeniyle zarar görmesini önler ve doğru çalışma voltajını sağlar. Breadboard ve jumper kablolar kullanılarak tüm bileşenler düzenli ve güvenli bir şekilde bağlanmıştır. Bu bileşenler, devrenin karmaşıklığını azaltır ve bağlantıların daha kolay yapılmasını sağlar.

Devrenin kurulumu sırasında, yeşil LED Arduino'nun dijital pin 2'sine, sarı LED pin 3'e ve kırmızı LED pin 4'e bağlanmıştır. Her bir LED'in diğer ucu, 220 ohm'luk bir direnç aracılığıyla Arduino'nun GND (toprak) pinine bağlanmıştır. Bu bağlantılar, LED'lerin sırasıyla yanmasını ve sönmesini sağlayarak trafik ışığının normal döngüsünü simüle eder. Breadboard üzerindeki bu düzen, devrenin daha modüler ve değiştirilebilir olmasını sağlar.

## 4.5. Arduino Kodu Yazma ve Yükleme

Trafik ışığı devresinin kontrolünü sağlamak için Arduino kodu yazılmış ve Arduino'ya yüklenmiştir. Bu kod, trafik ışıklarının belirli bir zaman döngüsü içinde çalışmasını ve acil durum aracı tespit edildiğinde ışıkların yeşil yanmasını sağlar. Arduino IDE kullanılarak yazılan kod, mikrodenetleyiciye yüklenir ve devrenin çalışma mantığını belirler.

metin, elektronik mühendisliği, elektronik donanım, devre bileşeni içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 14. Proje Sanal Çıktıları Elde Etmek İçin Trafik Işık Devresi

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Tablo 1. Devre Bileşenleri

metin, diyagram, dikdörtgen, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 15. Devrenin Şematik Görünümü

Kodun düzgün çalışması için gerekli olan en önemli bileşenlerden biri, acil durum aracının tespit edilmesidir. Bu tespit, PyCharm üzerinde geliştirilen görüntü işleme kodu ile gerçekleştirilir. Bu kod, kamera görüntülerini analiz eder ve acil durum aracı tespit edildiğinde Arduino'ya sinyal gönderir. Bu entegrasyon, trafik ışıklarının dinamik ve akıllı bir şekilde kontrol edilmesini sağlar.

Kod, öncelikle LED'lerin bağlı olduğu pinleri tanımlar ve bu pinleri çıkış (OUTPUT) olarak ayarlar. **setup** fonksiyonu içinde, seri haberleşme başlatılır (**Serial.begin(9600)**). Bu, bilgisayar ve Arduino arasındaki iletişimi sağlar. Normal trafik ışığı döngüsü içinde, yeşil ışık 20 saniye, sarı ışık 3 saniye ve kırmızı ışık 30 saniye yanar. Bu döngü, **loop** fonksiyonu içinde kontrol edilir.

Acil durum sinyali, görüntü işleme sistemi tarafından tespit edildiğinde Arduino'ya seri port üzerinden gönderilir. Bu sinyal, **Serial.available()** fonksiyonu ile kontrol edilir ve **Serial.read()** fonksiyonu ile okunur. Eğer acil durum sinyali alınırsa (**emergency = Serial.read() == '1'**), LED'lerin durumu değiştirilir ve yeşil ışık yanar. Acil durum aracı geçene kadar yeşil ışık yanmaya devam eder. Acil durum sinyali sona erdiğinde, trafik ışıkları normal döngüsüne geri döner.

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

Sonuç olarak, bu proje, trafik yönetimini iyileştirmek ve acil durum araçlarının hızlı geçişini sağlamak amacıyla tasarlanmıştır. Tinkercad ile oluşturulan sanal devre ve Arduino kodu, bu amaca hizmet ederek trafik ışıklarının dinamik bir şekilde kontrol edilmesini sağlar. Bu sistem, gerçek dünyada uygulanabilir bir çözüm sunar ve trafik güvenliğini artırmada önemli bir rol oynar.

# BÖLÜM 5:

# PROJE ENTEGRASYONU VE KODLAR

## 5.1 Proje Entegrasyonu

Proje, görüntü işleme, nesne tespiti, mikrodenetleyici programlama ve sanal devre simülasyonunu bir araya getirerek dinamik bir trafik yönetim sistemi oluşturur. Projenin entegrasyon süreci, farklı bileşenlerin bir araya getirilmesi ve birbirleriyle uyumlu çalışacak şekilde yapılandırılması üzerine kuruludur. Aşağıda, entegrasyon sürecinin adımları detaylı olarak açıklanmıştır.

### 5.1.1 Görüntü işleme ve nesne tespiti

Görüntü işleme ve nesne tespiti, projenin temel bileşenlerinden biridir. Bu bölümde, YOLOv8 modeli kullanılarak acil durum araçlarının tespiti yapılmaktadır.

* **Model Eğitimi:** Roboflow'da oluşturulan ve etiketlenen veri seti, Google Colab üzerinde YOLOv8 modeli ile eğitildi. Eğitim tamamlandıktan sonra model dosyası (örneğin, yolov8s.pt) indirilerek PyCharm projesine eklendi.
* **Görüntü İşleme:** OpenCV kütüphanesi kullanılarak video akışları (webcam veya video dosyası) alınır ve YOLOv8 modeli ile analiz edilir. Model, görüntülerdeki acil durum araçlarını (ambulans, itfaiye) tespit eder ve bu bilgiyi kullanarak trafik ışıklarını kontrol eder.

### 5.1.2 Arduino ile trafik ışığı kontrolü

Arduino, trafik ışıklarının kontrolünü sağlayan mikrodenetleyici olarak kullanılmıştır. Arduino, gelen sinyallere göre LED'leri kontrol eder.

* **Arduino Bağlantıları:** Yeşil, sarı ve kırmızı LED'ler, uygun dirençlerle birlikte Arduino'nun dijital pinlerine bağlanmıştır. Breadboard ve jumper kablolar, devre bileşenlerinin düzenli ve güvenli bir şekilde bağlanmasını sağlar.
* **Arduino Kodlama:** Arduino IDE kullanılarak, trafik ışıklarının normal döngüsü ve acil durum sinyalleri için gerekli kod yazılmıştır. Acil durum sinyali alındığında, yeşil ışık yanar ve acil durum aracı geçene kadar bu durumda kalır.

### 5.1.3 Tinkercad ile devre simülasyonu

Tinkercad Circuits, trafik ışığı devresinin sanal simülasyonunu oluşturmak için kullanılmıştır. Bu simülasyon, fiziksel devrenin nasıl çalışacağını görsel olarak test etmeyi sağlar.

* **Devre Tasarımı:** Tinkercad üzerinde yeşil, sarı ve kırmızı LED'ler ve Arduino UNO kullanılarak devre oluşturulmuştur. Her bir LED, uygun dirençlerle Arduino'nun dijital pinlerine bağlanmıştır.
* **Simülasyon:** Arduino kodu Tinkercad'e yüklenerek devre simüle edilmiştir. Bu simülasyon, trafik ışıklarının doğru şekilde çalıştığını ve acil durum sinyallerine yanıt verdiğini doğrular.

## 5.2. Kodlar

Projenin başarılı bir şekilde entegrasyonunu sağlamak için kullanılan temel kodlar aşağıda detaylı olarak verilmiştir.

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

# BÖLÜM 6:

# SONUÇ VE ÖNERİLER

## 6.1 Sonuçlar

Bu proje, trafik yönetim sistemlerinin dinamik ve akıllı hale getirilmesi amacıyla gerçekleştirilmiştir. Projede, görüntü işleme ve mikrodenetleyici programlama teknikleri kullanılarak acil durum araçlarına öncelik tanıyan bir trafik ışığı sistemi geliştirilmiştir. Bu sistem, acil durum araçlarının hızlı ve güvenli geçişini sağlamak için trafik ışıklarını dinamik olarak kontrol eder.

**Başlıca Sonuçlar:**

**1. Görüntü İşleme ve Nesne Tespiti:** YOLOv8 modeli kullanılarak, acil durum araçlarının tespiti başarılı bir şekilde gerçekleştirilmiştir. Model, yüksek doğruluk oranıyla ambulans ve itfaiye araçlarını tanımlayabilmiştir. OpenCV kütüphanesi ile video akışlarının işlenmesi ve nesne tespiti yapılmıştır.

**2. Arduino ile Trafik Işığı Kontrolü:** Arduino UNO kullanılarak trafik ışıkları dinamik olarak kontrol edilmiştir. Seri haberleşme protokolü sayesinde, görüntü işleme sistemi tarafından tespit edilen acil durum sinyalleri Arduino'ya iletilmiş ve trafik ışıkları bu sinyallere göre yanıt vermiştir.

**3. Tinkercad Simülasyonu:** Tinkercad Circuits kullanılarak trafik ışığı devresinin sanal simülasyonu oluşturulmuştur. Bu simülasyon, fiziksel devrenin nasıl çalıştığını test etmek ve doğrulamak için etkili bir araç olmuştur. Simülasyon, trafik ışıklarının doğru şekilde çalıştığını ve acil durum sinyallerine yanıt verdiğini göstermiştir.

**4. Proje Entegrasyonu:** Projenin tüm bileşenleri başarılı bir şekilde entegre edilmiştir. Python kodları, YOLOv8 modeli, OpenCV, PySerial ve Arduino arasında uyumlu bir çalışma sağlanmıştır. Bu entegrasyon, sistemin gerçek zamanlı çalışmasını ve acil durum araçlarına öncelik tanımasını mümkün kılmıştır.

## 6.2 Öneriler

Bu proje, trafik yönetim sistemlerinde önemli bir adım olsa da, gelecekte daha da geliştirilmesi gereken alanlar bulunmaktadır. İşte gelecekteki projeler ve geliştirmeler için bazı öneriler:

**Model Performansını Artırma:**

* **Daha Büyük Veri Setleri:** Modelin doğruluğunu artırmak için daha büyük ve çeşitli veri setleri kullanılabilir. Farklı hava koşulları, açılar ve trafik durumlarını içeren veri setleri, modelin genelleme yeteneğini artıracaktır.
* **Model İyileştirmeleri:** YOLOv8 modelinin daha yeni sürümleri veya diğer derin öğrenme modelleri (örneğin, EfficientDet) kullanılabilir. Bu modeller, daha yüksek doğruluk ve daha hızlı işlem süreleri sunabilir.

**Sistem Entegrasyonu:**

* **Bulut Tabanlı Çözümler:** Sistem, bulut tabanlı bir çözümle entegre edilerek daha geniş bir coğrafi alanda uygulanabilir. Bulut tabanlı çözümler, yüksek hesaplama gücü ve veri depolama kapasitesi sunarak sistemin ölçeklenebilirliğini artırabilir.
* **Gerçek Zamanlı Veri Analizi:** Gerçek zamanlı veri analizi ve makine öğrenimi modelleri kullanılarak trafik yoğunluğu, kazalar ve diğer olaylar anlık olarak analiz edilebilir. Bu analizler, trafik ışıklarının dinamik olarak ayarlanmasını sağlayarak trafik akışını optimize edebilir.

**Güvenlik ve Güvenilirlik:**

* **Yedek Sistemler:** Sistem arızası durumunda devreye girecek yedek sistemler kurulmalıdır. Bu, trafik ışıklarının sürekli olarak çalışmasını ve güvenliğini sağlar.
* **Güvenlik Önlemleri:** Veri güvenliği ve sistem güvenliği için gerekli önlemler alınmalıdır. Özellikle, veri iletimi sırasında güvenli protokoller kullanılmalı ve yetkisiz erişimlere karşı korunma sağlanmalıdır.

**Kullanıcı Eğitim ve Bilgilendirme:**

* **Sürücü Eğitimi:** Sürücülerin bu tür akıllı trafik sistemlerine aşina olmaları sağlanmalıdır. Eğitim programları ve bilgilendirme kampanyaları, sürücülerin bu sistemlere uyum sağlamalarını kolaylaştırır.
* **Topluluk Katılımı:** Toplumun bu tür projelere katılımı teşvik edilmelidir. Kamuoyu anketleri ve geri bildirim mekanizmaları, sistemin sürekli olarak iyileştirilmesine yardımcı olabilir.

**Genişletilmiş Uygulamalar:**

* **Diğer Acil Durum Araçları:** Proje, yalnızca ambulans ve itfaiye araçlarıyla sınırlı kalmamalıdır. Polis araçları, yol bakım ekipleri ve diğer acil durum araçları da sisteme entegre edilebilir.
* **Trafik İşaretleri ve Levhalar:** Trafik ışıklarına ek olarak, trafik işaretleri ve levhaların dinamik olarak kontrol edilmesi düşünülebilir. Bu, yol kullanıcılarına daha fazla bilgi sunarak güvenliği artırabilir.

Bu proje, akıllı şehirler ve dinamik trafik yönetim sistemleri için önemli bir örnek teşkil etmektedir. Görüntü işleme, nesne tespiti ve mikrodenetleyici programlama teknikleri kullanılarak, acil durum araçlarına öncelik tanıyan bir trafik ışığı sistemi başarılı bir şekilde geliştirilmiştir. Bu sistem, trafik kazalarını azaltma ve acil müdahale sürelerini kısaltma potansiyeline sahiptir.

Gelecekte yapılacak geliştirmelerle, bu sistemin daha geniş bir alanda uygulanabilirliği artırılabilir ve trafik yönetimi daha verimli hale getirilebilir. Özellikle, model performansının artırılması, bulut tabanlı çözümlerle entegrasyon ve sistem güvenliğinin sağlanması gibi konular, gelecekteki çalışmalar için önemli odak noktaları olabilir. Bu tür projeler, akıllı şehirlerin vazgeçilmez bir parçası olabilir ve topluma önemli katkılar sağlayabilir.

**ÖZGEÇMİŞ**

Fatma Güner, 13.10.2001 tarihinde Sakarya’da doğdu. 2019 yılında Şehit Üsteğmen Selçuk Esedoğlu Anadolu Lisesinden mezun oldu. 2020 yılında Sakarya Üniversitesi Bilgisayar Mühendisliği Bölümü’nü kazandı. 2023 yılında Toyota Motor Europe Şirketinde yazılım stajını ve yine 2023 yılında TÜRASAŞ Şirketinde donanım stajını yapmıştır. SAÜ Bilgisayar Mühendisliği Bölümünden 2023 yılında mezun olmuştur.

# KAYNAKÇA

[1] Muhammed K. T., Funda A., Kamil O. (2020). Görüntü İşleme Yöntemleri Kullanılarak Araçlar İçin Akıllı Park Sistemi Tasarımı. Journal Article. 129-140.

[2] Hartenstein, H., & Laberteaux, K. P. (2008). A tutorial survey on vehicular ad hoc networks. IEEE Communications Magazine, 46(6), 164-171.

[3] Fatih Ş., Kemal A. (2024). YOLOv7 ve YOLOv8 Algoritmalarının Değerlendirilmesi: Savaş Uçaklarının Tespiti İçin Performans Analizi. International Artificial Intelligence and Data Science Congress (4 th). 276-282.

[4] Abdullah Sökülmez (2023). Karayolu Bakış Açılı Görüntü İle Derin Öğrenme Tabanlı Araç Algılama-Sayma, Trafik Yoğunluğu Hesabı Yapılması Ve Google Haritalara Anlık Veri Sunma Sistemi. YBS Ansiklopedi. 11(1). 1-10.

[5] <https://www.tinkercad.com/circuits> , Erişim Tarihi: 05.05.2024.

[6] İsabek T., Seda K., Ali Ş., Fatmanur K., Süleyman E., Ahmet S., Yaşar B. (2018). Konvolüsyonel Sinir Ağı Kullanarak Gerçek Zamanlı Araç Tipi Sınıflandırması. Kocaeli Üniversitesi, Kocaeli, Türkiye. 1-5.

[7] Ahmet F. B., Vasif N. (2023). Derin öğrenme tabanlı saklanan kamufle tankların tespiti: son teknoloji YOLO ağlarının karşılaştırmalı analizi. Gümüşhane Üniversitesi Fen Bilimleri Dergisi, GUFBD / GUJS (2023) 13(4): 1082-1093.

[8] <https://www.youtube.com/watch?v=_4ZhpmqD0D4> , Erişim Tarihi: 15.04.2024.

[9] <https://roboflow.com/models> , Erişim Tarihi: 17.04.2024.

[10] Pazar, Ş., Bulut, M., Uysal, C., (2020). Yapay Zeka Tabanlı Araç Algılama Sistemi Geliştirilmesi, Bilim, Teknoloji ve Mühendislik Araştırmaları Dergisi, 1(1), 31-37.

[11] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 779-788).

[12] Chen, Y., Li, X., Chen, G., & Zhang, X. (2017). Deep learning-based method for managing vehicle traffic at intersections. Journal of Advanced Transportation, 2017, 1-10.

[13] <https://ulastirma.info/pilot-uygulama-istanbul-fatihte-basladi/> , Erişim Tarihi: 10.04.2024.

**BSM 401 BİLGİSAYAR MÜHENDİSLİĞİ TASARIMI**

**Değerlendİrme ve Sözlü Sınav Tutanağı**

KONU :

ÖĞRENCİLER (Öğrenci No/AD/SOYAD):

|  |  |  |  |
| --- | --- | --- | --- |
| Değerlendirme Konusu | İstenenler | Not Aralığı | Not |
| **Yazılı Çalışma** |  |  |  |
| **Çalışma klavuza uygun olarak hazırlanmış mı?** | x | 0-5 |  |
| **Teknik Yönden** |  |  |  |
| **Problemin tanımı yapılmış mı?** | x | 0-5 |  |
| Geliştirilecek yazılımın/donanımın mimarisini içeren blok şeması (yazılımlar için veri akış şeması (dfd) da olabilir) çizilerek açıklanmış mı? |  |  |  |
| Blok şemadaki birimler arasındaki bilgi akışına ait model/gösterim var mı? |  |  |  |
| Yazılımın gereksinim listesi oluşturulmuş mu? |  |  |  |
| Kullanılan/kullanılması düşünülen araçlar/teknolojiler anlatılmış mı? |  |  |  |
| Donanımların programlanması/konfigürasyonu için yazılım gereksinimleri belirtilmiş mi? |  |  |  |
| UML ile modelleme yapılmış mı? |  |  |  |
| Veritabanları kullanılmış ise kavramsal model çıkarılmış mı? (Varlık ilişki modeli, noSQL kavramsal modelleri v.b.) |  |  |  |
| Projeye yönelik iş-zaman çizelgesi çıkarılarak maliyet analizi yapılmış mı? |  |  |  |
| Donanım bileşenlerinin maliyet analizi (prototip-adetli seri üretim vb.) çıkarılmış mı? |  |  |  |
| Donanım için gerekli enerji analizi (minimum-uyku-aktif-maksimum) yapılmış mı? |  |  |  |
| Grup çalışmalarında grup üyelerinin görev tanımları verilmiş mi (iş-zaman çizelgesinde belirtilebilir)? |  |  |  |
| Sürüm denetim sistemi (Version Control System; Git, Subversion v.s.) kullanılmış mı? |  |  |  |
| Sistemin genel testi için uygulanan metotlar ve iyileştirme süreçlerinin dökümü verilmiş mi? |  |  |  |
| Yazılımın sızma testi yapılmış mı? |  |  |  |
| Performans testi yapılmış mı? |  |  |  |
| Tasarımın uygulamasında ortaya çıkan uyumsuzluklar ve aksaklıklar belirtilerek çözüm yöntemleri tartışılmış mı? |  |  |  |
| **Yapılan işlerin zorluk derecesi?** | x | 0-25 |  |
| **Sözlü Sınav** |  |  |  |
| **Yapılan sunum başarılı mı?** | x | 0-5 |  |
| **Soruları yanıtlama yetkinliği?** | x | 0-20 |  |
| **Devam Durumu** |  |  |  |
| **Öğrenci dönem içerisindeki raporlarını düzenli olarak hazırladı mı?** | x | 0-5 |  |
| **Diğer Maddeler** |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **Toplam** |  |  |  |

Danışman :

danışman imzası: