GEBZE TEKNİK ÜNİVERSİTESİ

Elektronik Mühendisliği

ELM463 – Görüntü İşleme

Dönem Projesi Raporu

‘Görüntüdeki Aracın Tespit Edilmesi’

Ad Soyad: Öykü Arslan

Öğrenci No: 1801022013

1. Giriş :

Araçların kaldırımlara park edilmesi yayalar için ciddi bir sorun ve tehlike oluşturmaktadır. Bu sorundan yola çıkılarak görüntü işlemeyle araba tespiti problemi üzerinde durulmuştur.

Bu projede amaç, arabaların tespit edilmesidir. Bu problemin çözümünde eşikleme, ayrıt saptama, morfolojik operatörler, çizgi (line) tespiti, filtreleme yöntemleri kullanılarak problemin çözülmesi hedeflenmiştir.

2. Tasarım:

2.1. Literatür Araştırması ve Methodların Uygulanması

İlk olarak projede uygulanacak methodlar için bir literatür araştırması yapılmıştır. Araç tespiti için canlı video üzerinden projeler yapılmış olup kullanıcak methodlara ait bir kaynağa erişilemedi. Bu nedenle methodların artı ve eksi yönleri araştırılmıştır. Uygulandıkları alanlar tespit edilmiştir. Bu projede etkilerini görmek adına kullanılacak methodlar tek tek denenmiş olup algoritmanın oluşturulmasında önemli rol oynamıştır. Bu proje Python programında OpenCv kütüphanesi kullanılarak gerçekleştirilmiştir. Üzerinde metodların denendiği test görüntüsü Şekil gibidir:

2.1.1. Eşikleme Methodu (Thresholding)

Verilen kaynak görüntü ikili (binary) görüntüye çevirmek istenirse (görüntünün siyah ve beyaz olarak tanımlanması) ve görüntü üzerindeki gürültülerin azaltılması veya üzerinde nesne belirlenmesi gibi farklı amaçlarla kullanılmaktadır. Giriş olarak verilen görüntü üzerinde uygulanan eşikleme tipine bağlı olarak, pikselleri verilen eşik değerine göre siyah ya da beyaz olarak güncellemektedir.

2.1.2. Ayrıt Saptama (Edge Detection)

Ayrıt saptama, bir görüntü içindeki nesnelerin veya bölgelerin sınırlarını (kenarlarını) belirlemek için kullanılan bir görüntü işleme tekniğidir. Ayrıtlar, görüntülerle ilişkilendirilen en önemli özelliklerden

biridir. Bir görüntünün temel yapısı kenarlarından tanınır. Bu nedenle nesne tespiti yapılırken ayrıt saptamanın önemli bir method olduğu gözlenmiştir.

Bunun için Canny Edge Detection yöntemi uygulanmıştır. Bu yöntem çok adımlı bir algoritma olup birçok görüntü işleme projesinde kullanılmaktadır. İlk olarak gürültü azaltılıp ardından görüntünün yoğunluk gradyanı bulunur ve daha sonra Gradyan büyüklüğü ve yönü alındıktan sonra, kenarı oluşturmayabilecek istenmeyen pikselleri çıkarmak için tam bir görüntü taraması yapılır. Bunun için her pikselde pikselin gradyan yönünde komşuluğunda yerel bir maksimum olup olmadığı kontrol edilir. Bütün bu adımları OpenCv'de bulunan cv2.Canny() fonksiyonu sağlamaktadır.

2.1.3. Morfolojik Operatörler

2.1.3.1. Erozyon : Ön plandaki nesnenin sınırlarını aşındırır. Kernel, görüntü boyunca kayar. Sınırın yakınındaki tüm pikseller, Kernelin boyutuna bağlı olarak atılır. Böylece görüntüdeki ön plan nesnesinin kalınlığı veya boyutu azalır veya sadece beyaz bölge azalır. Küçük beyaz gürültüleri gidermek, birbirine bağlı iki nesneyi ayırmak vb. için kullanışlıdır.

2.1.3.2. Genişletme (Dilation) : Erozyonun tam tersidir. görüntüdeki beyaz bölge artar veya ön plandaki nesnenin boyutu artar. Normalde gürültü giderme gibi durumlarda erozyonu genişletme takip eder. Çünkü erozyon beyaz sesleri ortadan kaldırır ancak aynı zamanda nesneyi de küçültür.

2.1.3.3. Opening : Erozyondan sonra genişletme uygulamasıdır. Gürültüden kurtulmak için uygulanır.

2.1.3.4. Closing: Opening işleminin tam tersidir. Genişletmeden sonra erozyon kullanılması işlemidir.

2.1.3.5. Gradyan: Genişletme ve erozyonun farkıdır.

2.1.4. Çizgi(line) tespiti

Hough Transform, herhangi bir şekli tespit etmek kullanılır. Bu nedenle çizgi tespiti yapılırken bu dönüşümden yararlanılmıştır.

2.1.5. Filtre (Laplacian): Bu işlem için “Laplacian()” adında bir metot kullanılır. Bu metot içerisine sadece işlenecek görüntü ve çıktının derinliğini alır.

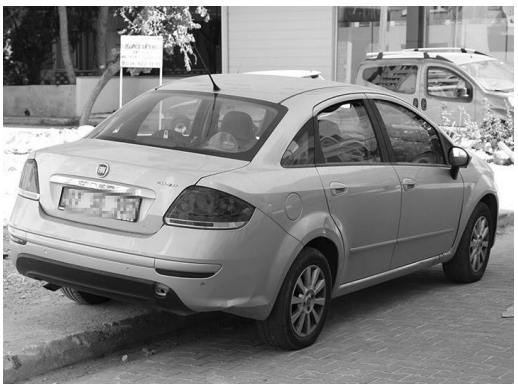
2.2. Gerçekleme

Araştırılan bu metodlar birçok farklı algoritmayla denenmiştir ve buna en uygun olacak şekilde bir algoritma oluşturulmuştur. Bu yönteme göre ilk olarak

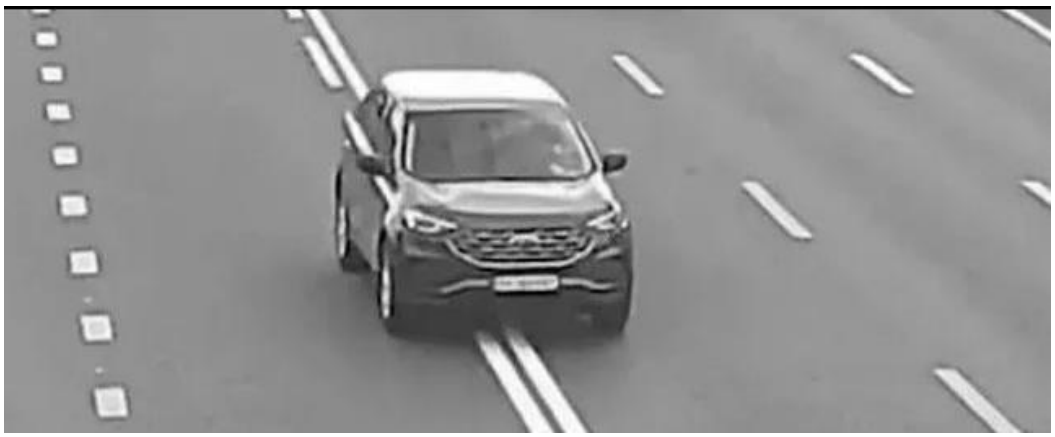
1- Median filtresi ile blurlama işlemi gerçekleştirilmiştir. Ardından
2- Laplacian filtresi kullanılarak görüntünün derinliği alınmıştır. Daha sonra elde edilen görüntünün kenarlarını saptamak için
3- Canny edge detection yöntemi uygulanmıştır.
4- Thresholding görüntüsü ile görüntü siyah beyaz olarak tanımlanmıştır.
5- Gradient görüntüsü alınarak genişletme ve erozyonun farkı alınmıştır. Daha sonrasında
6- Line detection ile sınırların belirlenmesinden önce opening yapılmıştır ve opening yapılmasıyla birlikte elde ettiğim sonucun daha doğru olduğu gözlenmiştir ve opening ardından line detection yapılarak çıktı görüntüleri elde edilmiştir. Oluşturulan bu görüntüler 10 farklı fotoğrafta düzenlenmiştir. Ancak rapora yalnızca 6’sı dahil edilmiştir.

Algoritma oluşturulma aşamasında yararlanılan eğitim seti aşağıdaki gibidir.

Şekil 1. ‘egitim_verisi4’

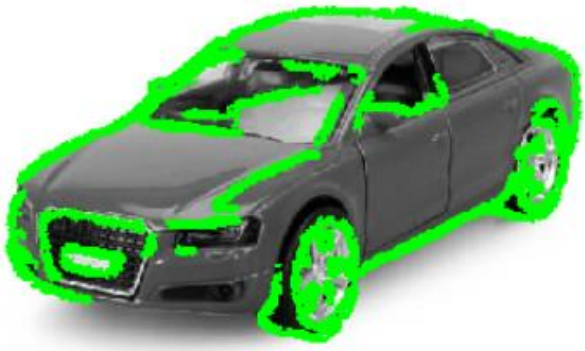


Şekil 2. ‘egitim_verisi3’



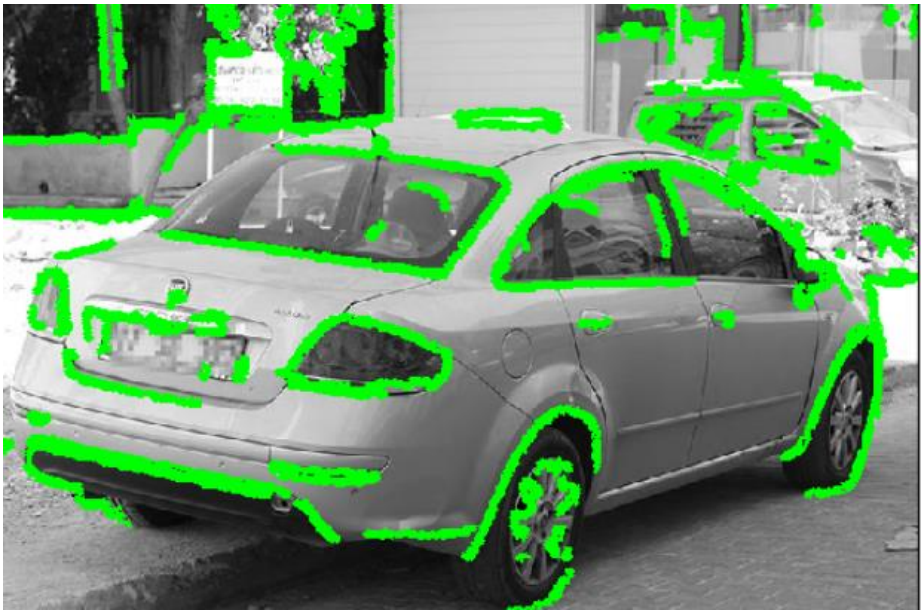
Şekil 3. ‘egitim_verisi5’

Bu eğitim verilerinin görüntülerinin kullanılmasıyla elde edilen sonuçlar ise aşağıdaki gibidir.



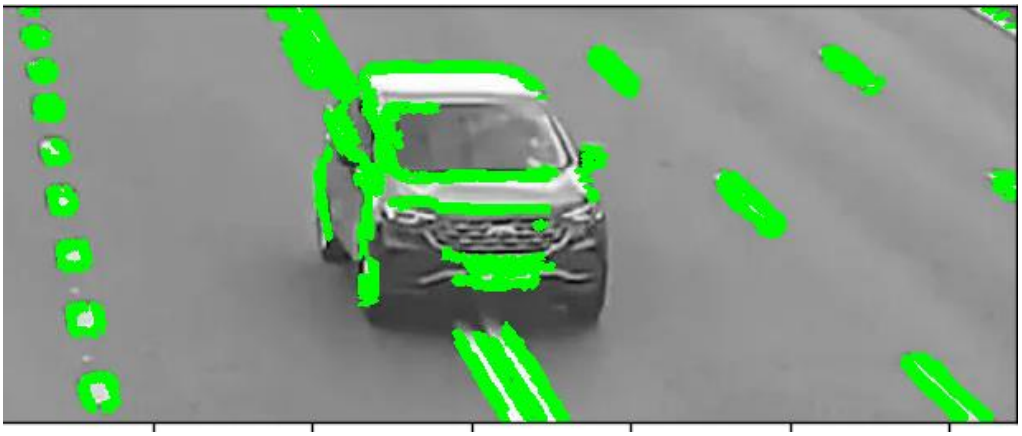
Şekil 4. ‘Şekil 1. İçin çıktı görüntüsü’

Bu görüntü eğitim seti içindeki en basit olarak seçilen görüntüdür. Bu görüntü için de yaklaşık %85 gibi bir başarı oranı söylenebilir.



Şekil 5.. ‘Şekil 2 için çıktı görüntüsü’

Bu görüntü eğitim seti içindeki en zor görüntüdür. Çünkü arka planda da birçok detay vardır. Bu görüntü için başarı yüzdesinin düşük olduğunu düşünülmektedir. Başarı oranı %20dir.



Şekil 6. ‘Şekil 13 için çıktı görüntüsü’

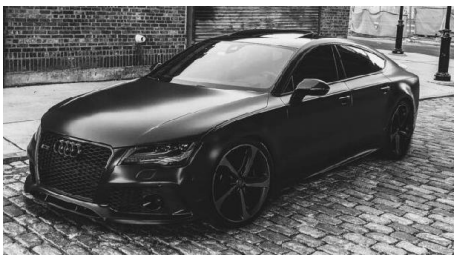
Bu görüntüde ise proje kapsamında sadece kaldırımda değil yolda olan araçların da test görüntüsü istendi. Bunun üzerine eğitim verisine eklenen bu görüntü üzerinde algoritma uygulandığında yol

üzerindeki çizgilerin de tespit edilmesiyle aracın yolda olduğu gözlenmiştir ve başarısız sayılmayacak bir görüntü elde edilmiştir. Başarı oranı %80dir.

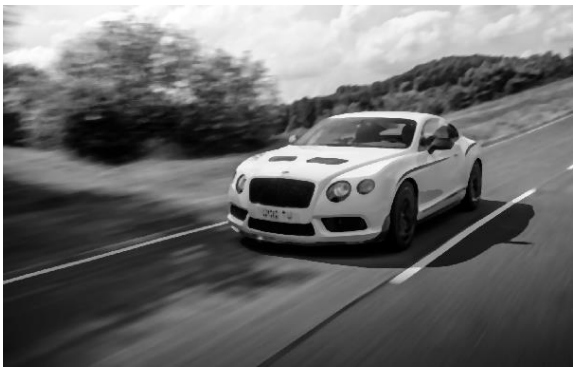
Bu yöntemle yetinilmeyip ardından birçok algoritma denenmiştir ancak en başarılısı bu yöntem olmuştur. Bu algoritmaya karar verildikten sonra test veri seti üzerinde denenmiştir.

Test veri seti aşağıdaki gibidir:

Şekil 7. ‘test_veri2’

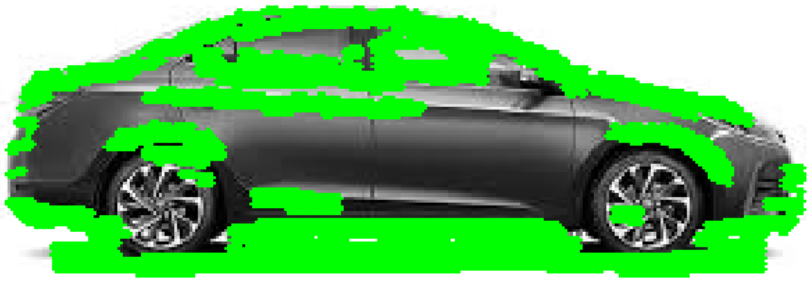


Şekil 8. ‘test_veri4’



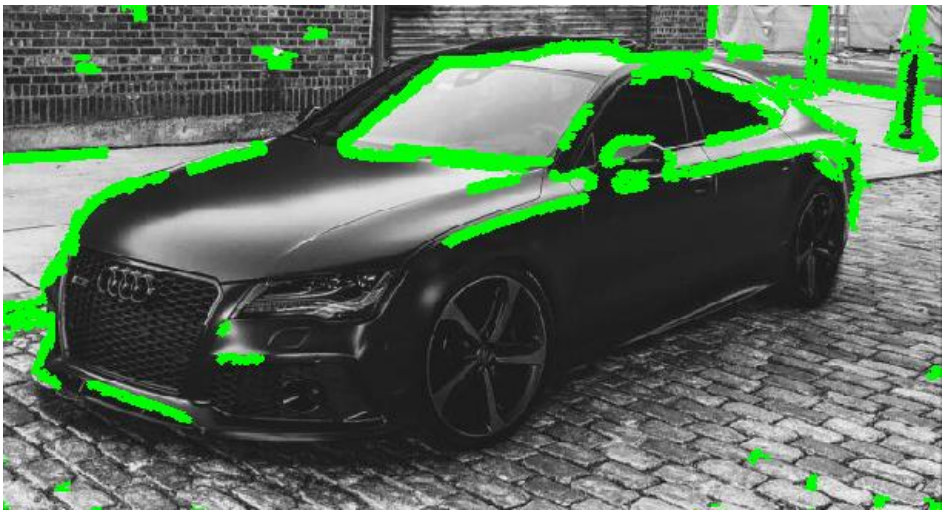
Şekil 9. ‘test_veri5’

Bu eğitim verilerinin görüntülerinin kullanılmasıyla elde edilen sonuçlar ise aşağıdaki gibidir.



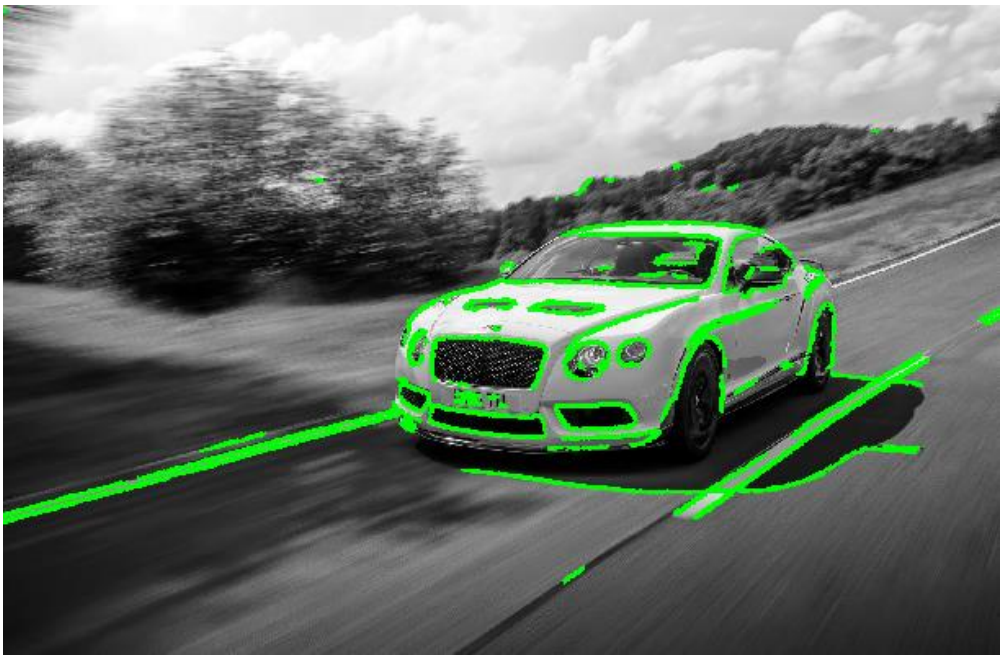
Şekil 10. ‘Şekil 7 için çıktı görüntüsü’

Bu görüntü test verilerindeki en kolay görüntüdür ve %100e yakın bir performans sergilediği düşünülmektedir.



Şekil 11. ‘Şekil 8 için çıktı görüntüsü ’

Şekil 8’deki görüntü test verileri içindeki en zor görüntüdür. Arka planla aracın renk benzerliği ve detay çokluğu tespit işlemini zorlaştırmıştır ve başarı oranı düşüktür. Başarı oranı %10 dur.



Şekil 12. ‘Şekil 9 için’ çıktı görüntüsü’

Şekil 12’de arka plan hareketli olduğundan aracın ve yolun tespiti kolaylaştırmıştır ve en başarılı görüntülerden biridir. Başarı oranı %95tir.

3. Sonuç – Yorum – Analiz

Bu projede araçların kaldırımlara park etmesi probleminden yola çıkılarak bir araba tespiti algoritması geliştirilmiştir. Python programı ve OpenCv kütüphanesi kullanılmıştır. Methodlar ilk olarak ayrı ayrı araştırılmış olup, ardından birleştirilerek bir algoritma geliştirilmiştir. Bazı görüntüler için olumlu sonuçlar elde edilirken bazı görüntüler için olumsuz sonuçlar elde edilmiştir. Test ve eğitim verilerindeki fotoğraflar zor, kolay ,orta olacak şekilde seçilmiştir. Bu projede görüntü işleme yöntemleri pekiştirilmiş olup, kullanım alanlarına daha fazla hakim olunmuştur. Ancak Çözülemeyen problemlerden biri arka plandaki gürültüden kurtulmak olmuştur. Algoritma tasarlanırken arka planın yok edilememesi büyük bir sorun olmuştur. Başta bir maske tasarlanarak gürültüden kurtulmaya çalışılmıştır ancak başarılı olunamamıştır. Çünkü her görüntü için bu yöntem değişkenlik göstermiştir ve sonunda vazgeçilmiştir. Ardından opening yöntemi uygulanarak sorunun büyük çoğunluğu giderilmiştir. Kullanılan diğer threshold yöntemleri ise derlemesi çok uzun zaman aldığından kullanılmamıştır. Başarı oranları

karşılaştırıldığında %65 - %70 gibi bir başarı oranı elde edilmiştir. Bu yüzdenin düşmesinde zor fotoğrafların etkisi çoktur. Fortoğraf çok zor olmadığı sürece algoritma kullanılabilir.

4. Kaynakça

https://docs.opencv.org/4.x/da/d22/tutorial_py_canny.html
https://learnopencv.com/edge-detection-using-opencv/
https://medium.com/analytics-vidhya/building-a-lane-detection-system-f7a727c6694
https://docs.opencv.org/3.4/d9/d61/tutorial_py_morphological_ops.html