Uluslararası Sürdürülebilir Mühendislik ve Teknoloji Dergisi
International Journal of Sustainable Engineering and Technology
Sayı 2, Cilt 6, (2022), 39-47

OTONOM ARAÇLAR İÇİN MİKRODENETLEYİCİ TABANLI ÇEVRESEL GÜVENLİK SİSTEMİ TASARIMI

Umutcan TÜZÜN[*1], Merdan ÖZKAHRAMAN[1], Bekir AKSOY[1]

[1]Isparta Uygulamalı Bilimler Üniversitesi, Teknoloji Fakültesi, Mekatronik Mühendisliği Bölümü, Isparta

Makale Bilgisi
Geliş tarihi: 22.07.2022
Kabul Tarihi: 26.12.2022
Yayın tarihi: 30.12.2022

ÖZET

Günümüzde sağlık alanı, video işleme, robot görüşü, sürücüsüz araç gibi pek çok örnekte karşımıza çıkan görüntü işleme sürekli büyüyen bir daldır. Görüntü işleme uygulamaları, otonom araçlarda nesne algılama ve şerit takibinin yapılabilmesinde oldukça fayda sağlamaktadır. Bu çalışmada trafik kazalarındaki artış ve sürücü hataları göz önünde bulundurularak sürücü hatası sebebiyle oluşan kazaları en aza indirgemek ve otonom araçlara olan güveni artırmak hedeflenerek bir otonom araç sistemi geliştirilmiştir. Görüntü işleme, Raspberry Pi'a OpenCv kütüphanesi kurularak gerçekleştirilmiştir. Böylece şerit takibi yapılmış ve hareketli nesneler algılanmıştır. Yol sınırları ve kenarların algılanması için Canny kenar algoritmasından yararlanılmıştır. Yazılımlar için Python programlama dili kullanılarak kodlar yazılmıştır. Çalışmanın trafikte güvenli sürüşe ve araçlardaki görüntü işleme uygulamalarının artırılmasını sağlamaya yardımcı olacağı düşünülmektedir.

Anahtar Kelimeler; Görüntü işleme, OpenCV, Otonom araç, Python, Raspberry Pi.

MICROCONTROLLER BASED ENVIRONMENTAL SAFETY SYSTEM DESIGN FOR AUTONOMOUS VEHICLES

Article Info
Received: 22.07.2022
Accepted: 26.12.2022
Published: 30.12.2022

ABSTRACT

Today, image processing is an ever-growing branch that we encounter in many examples such as healthcare, video processing, robot vision, and driverless vehicles. Image processing applications are very useful for object detection and lane tracking in autonomous vehicles. In this study, an autonomous vehicle system has been developed with the aim of minimizing the accidents caused by driver error and increasing the confidence in autonomous vehicles, taking into account the increase in traffic accidents and driver errors. Image processing was carried out by installing OpenCv library in Raspberry Pi. Thus, lane tracking was performed and moving objects were detected. Canny edge algorithm is used to detect road boundaries and edges. Codes were written for the software using the Python programming language. It is thought that the study will help safe driving in traffic and increase the image processing applications in vehicles.

Keywords; Autonomous vehicle, Image processing, OpenCV, Python, Raspberry Pi.

1. Giriş

Teknolojinin gelişmesiyle birlikte dünyada sektörler arası yenilik ve rekabette sürekli artış görülmektedir. Otomotiv sektörü de diğer sektörler gibi bu gelişmelerden etkilenerek her defasında kullanımı ve rahatlığı göz önünde bulundurduğu yeniliklere yönelmektedir. Yapılan yeniliklerde önceliğin rahatlık ve güvenlikli kullanım olduğu görülmektedir.

Dünya'da her yıl yaklaşık 1.2 milyon insan trafik kazası sonucu hayatını kaybetmektedir (More vd., 2019). Yapılan araştırmalar sonucu trafik kazalarının meydana gelmesindeki en büyük nedenin insan hatası olduğu görülmektedir (More vd., 2019).

TÜİK verilerine göre trafik kazalarının sebepleri arasında yolcu kusuru, yaya kusuru, yol kusuru ve araç kusuru gibi etkenler bulunsa da en önemli etkenin sürücü hatası olduğu tespit edilmektedir. Araçların otonom hale gelmesi durumunda trafik kazalarının önemli ölçüde azalacağı öngörülmektedir (Seçkin, 2021).

Son yıllarda, otonom araçların, insan müdahalesi olmadan hareket edebilen araçların tasarlanması ve test edilmesine artan bir ilgi görülmektedir (Rossi vd., 2020). Günümüzde Tesla, Google ve Uber gibi dünyaca ünlü markalar, kendi sürücüsüz arabalarını yaratmaya çalışmaktadır (Tian, 2019). Pek çok analist, önümüzdeki 5 yıl içinde şehirlerimizde tam otonom arabalara sahip olmaya başlayacağımızı ve 30 yıl içinde neredeyse tüm arabaların tamamen otonom olacağını tahmin etmektedir (Tian, 2019).

Otonom araçlar, otomobillerde geçirilen zamanın kalitesini ve üretkenliğini artırma, ulaşım sisteminin güvenliğini ve verimliliğini artırma ve ulaşımı herkesin her an erişebileceği bir yardımcı programa dönüştürme potansiyeline sahiptir (Schwarting vd, 2018). Bir otonom araç için çevresinden bilgi almanın ilk yolu sensörlerdir. Günümüzde, sensör teknolojisinin gelişmesi ile birlikte, mobil robotların ve otonom araçların ihtiyaç duyduğu sensörler verimli bir şekilde kullanılabilmektedir (Bingöl vd., 2019). Nesne algılama ve tanıma, otonom sürüş uygulamalarının en önemli araştırma konularından biridir. Bunun sebebi, otonom sürüşte bir kontrol hareketinin öncelikle nesneyi algılaması ve sonra da o nesneyi tanımlamasının gerekliliğidir (Bingöl vd., 2019). Son zamanlarda, gerçek hayatta kullanılan araçlar için nesne tanıma uygulamaları hızla büyümüştür. Bahsedilen uygulamalara örnek olarak, şerit algılama asistanı, stereo görüntü kullanarak engel algılama, kızılötesi görüntü kullanarak yaya algılama uyarı sistemi, lazer radar ve tek lensli kamera sistemi birleşimi kullanarak araç algılama sistemleri verilebilir (Bingöl vd., 2019).

2. Kaynak Özetleri

Son yıllarda teknolojinin gelişmesi ile paralel olarak otonom araç uygulama çalışmalarında da bir artış görülmektedir. Birçok sistemin bulunduğu otonom araç uygulamalarında, farklı şerit tanıma ve görüntü algılama sistemleri kullanılsa da aynı amaç doğrultusunda birçok çalışma ortaya koyulmuştur. Yapılan sistemlerde araç hatasını en aza indirerek güvenli bir yol görüşü sağlamak amaçlanmıştır.

Assidiq (2008), çalışmasında, değişen yol koşullarında şeritleri tespit etme yeteneğine sahip görüş tabanlı bir şerit tespit yaklaşımı sunmayı amaçlamıştır. Oluşturulan sistemde görüntüleri elde etmek için araca bir kamera monte edilmiştir. Elde edilen veriler, kenarları elde etmek için gri tonlamalı dönüştürme, gürültü giderme ve otomatik eşikleme ile kenar algılamaya tabi tutulur. Sınırlı arama alanı ile satırlar Hough Dönüşümü yöntemi kullanılarak çıkarılır. Şerit sınırı taraması, yolun çizgilerini veya sınırlarını temsil eden sağ ve sol taraftaki bir dizi noktayı döndürmek için önceki aşamalardan gelen bilgileri kullanır. Son olarak, şerit sınırlarını temsil etmek için bu veri noktalarına bir çift hiperbol takılır. Önerilen şerit algılama sistemi, farklı hava koşullarında kavisli ve düz yol yapısına sahip boyalı yollarda uygulanabilmektedir. Deneysel sonuçlar, önerilen şemanın farklı koşullar altında yol şerit işaretlemesini gündüz %96,6 ve gece %92,6'ya ulaşarak verimli bir şekilde tespit edebildiğini göstermektedir. Sonuç olarak sistemin, mevcut diğer sistemlere göre daha üstün ve sağlam sonuçlar verdiği belirtilmektedir.

Ujjainiya ve Chakravarthi (2015), USB web kamerası ile yakalanan test görüntüsü üzerinde açıklayıcı bir karşılaştırmalı analiz yapılmıştır. Python programlama dili benimsenmiş ve Raspberry – Pi kartı üzerinde OpenCV'de kenar algılama işlemleri gerçekleştirilmiştir. Elde edilen sonuçlara göre canny kenar algılama tekniğinin diğer muadillerine göre daha iyi ve okunaklı kenarlar sağladığı sonucuna varılabilmektedir.

Mandlik vd. (2016), oluşturdukları sistemin temel amacı şeritten ayrılmayı tespit ederek yol kazalarını önlemek ve yayaların güvenliğini sağlamaktır. OpenCV platformu kullanılarak, şerit tespiti Hough dönüşüm algoritması kullanılarak yapılır ve kenarlar, canny kenar dedektörü kullanılarak tespit edilir. Sonuçlar, gerçek zamanlı şerit belirleme ve izlemenin verimli bir şekilde yapıldığını kanıtlamaktadır. Şerit, düz ve virajlı yollar için doğru şekilde algılanmaktadır. Araçların şeritten çıkmaya çalışması durumunda sürücüye gerekli uyarı verilmektedir. Şerit değiştirme uyarı sistemi

için önerilen teknikler de çekilen videolar ile başarılı bir şekilde kontrol edilmiştir.

Ariyanto vd. (2019), yaptıkları çalışmada, yol görüntüsünün kenarını, çizgisini ve köşesini algılayabilen ve ayrıca trafik ışığı görüntüsünün kırmızı rengini algılayabilen bir araba modelinin geliştirilmesi sunmaktadır. Araba modeli, bilgisayarla görme amaçlı kullanılan bir kamera ile donatılmıştır. Kameradan gelen görüntü Raspberry Pi tek kartlı bilgisayar kullanılarak okunmuştur. Trafik ışığının çizgi, kenar, köşe ve kırmızı rengini algılamak için gömülü algoritmalar test edilmiştir. Test sonuçlarına göre, gömülü görüntü işleme algoritmaları, yol görüntülerinin çizgisini, kenarını ve köşesini başarıyla algılayabildiği ve trafik ışığı görüntüsünün kırmızı rengini algılayabildiği görülmektedir.

Rossi vd. (2020), şerit algılayan ve herhangi bir insan müdahalesi olmaksızın hareketini planlayan bir araç prototipi oluşturmuştur. Yaptıkları çalışmada, Raspberry-Pi 3.0 Model B tarafından görüntüler yakalanarak, görüntü işleme algoritmaları OpenCV 4.2 ile Python 3.7.4'te yazılmıştır. Şeritleri tespit etmek için Canny kenar algılama algoritması ve Hough dönüşümü kullanılmıştır. Şerit tespitinden sonra Kalman filtre tahmin yöntemi kullanılarak izlenmiştir. Prototip, gerçek zamanlı olarak test edilmiş ve prototipin yol şeritlerini algılayabildiği ve hareketini başarılı bir şekilde planlayabildiği sonucuna ulaşılmıştır.

Sevgi (2020), yaptığı çalışmada, sürücüsüz araç yazılımları geliştirmek için küçük boyutlu akülü bir araç üzerinde modifiyeler yaparak test düzeneği oluşturmuştur. OpenCV kütüphanesi ile birlikte görüntü işleme yöntemi kullanılarak, Canny algoritması ile şerit takibi yapılmıştır. Tabela tanıma sistemi için ise Derin öğrenme yönteminden yararlanılmıştır. Deneysel çalışmalarda otonom sürüş için tasarlanan şerit takibi ve tabela tahmini görevlerinin yüksek doğrulukta yapıldığı sonucuna ulaşılmıştır.

Küçük, Yavşan ve Gökçe (2021), yaptıkları çalışmada, otonom araçların trafikte sürücüye yardımcı olması, trafik işaretlerini tanıması ve sürücüye uyarıda bulunmasını sağlamak için bir sistem geliştirilmesi amaçlanmıştır. Sistem geliştirilirken araca kamera bağlanarak yol görüntü ve videolarına ait verilerle trafikte bulunan işaretlerin ve trafik lambalarının tanınması sağlanmıştır. Yüksek tanıma doğruluğu sağlamak amacıyla çeşitli yapay zeka algoritmalarından yararlanılmıştır. Araçtan elde edilen veri seti kullanılarak bir yapay sinir ağı modeli geliştirilmiştir. Deneysel olarak yapılan bu çalışmada %90 oranında doğruluk elde edilmiştir. Trafik işaret ve lambalarını tanıma sisteminden sonra otonom bir araç platformu için şerit tanıma ve viraj algılama sistemi geliştirilmiştir. Bu sistem sayesinde aracın direksiyon açısı hesaplanmış ve bir PID kontrolcüyle aracın direksiyonu otonom olarak kontrol edilmiştir.

Panfilova vd. (2021), yol işaretleri tespiti için bir algoritma sunmaktadır. Hızlı Hough Dönüşümü (FHT) hesaplanarak düz bir çizgi algılanır. Algılanan işaretler, sınırlı bir maksimum eğrilik açısına sahip çoklu çizgilerle yaklaştırılır. Az sayıda parametreyle, algoritma çeşitli şekillerdeki yol işaretlerini (düz, kavisli, daireler, çoklu paralel 6 çizgiler) algılayabilmektedir. Algoritma, "Kalibr" (Moskova) deney alanında sürüş yapan otonom aracın önden bakan kamerasından toplanan gerçek veriler üzerinde test edilmiştir. Otonom araç hassas konumlandırma için yeterli olan yol işaretleme dedektörünün hassasiyeti %43 ve geri çağırma %73 olarak değerlendirilmiştir. Tüm yollarda algılanan bir dizi parçanın çaprazlanması için önerilen şema sayesinde karmaşık durumlarda bile belirtilen kısımları seçebilmektedir.

3. Materyal – Metot

Bu çalışmada, şerit algılama ve hareket algılama işlemlerini sağlayabilmek için görüntü işleme tekniği kullanılmıştır. Görüntü işleme, Raspberry Pi'a OpenCV kütüphanesi kurularak ve sistemimize uygun Python kodu yazılarak gerçekleştirilmiştir. Çalışmada görüntü işleme kullanmamızın amacı, görünmesi zor nesneleri gözlemlemek, görüntüdeki nesneleri ayırt etmek ve otonom araçlarda görüntü işlemenin kullanılmasını yaygınlaştırmaktır.

Sistemin karşılaşacağı cisimler ile arasındaki mesafeyi ölçmek için ise HC-SR04 sensörü kullanılmıştır. HC-SR04 sensörü, ses dalgası sistemi kullanarak karşısında bulunan cisimler ile arasındaki mesafeyi iki metreye kadar ölçebilmektedir.

3.1. Görüntü işleme

Görüntü işleme, bir bilgisayarda, görüntülerden anlamlı ve faydalı verilerin işlenmesi ve çıkarılmasıyla ilgilenen bir alandır (Fernandes,

2021). Görüntüler yapılandırılmamış verileri temsil ettiğinden, tablolar ve formlar gibi yapılandırılmış verileri işlemekten çok daha zordur. Uygulamaya bağlı olarak, görüntü işleme birçok alanda ve birçok farklı yöntem ve algoritma ile uygulanabilir. Şu anda tıp alanı, video işleme, örüntü tanıma, makine/robot görüşü, sürücüsüz arabalar vb. birçok alanda kullanımda olan büyük ve büyüyen bir daldır (Fernandes, 2021).

Literatüre bakıldığında hareketli nesnelerin algılanabilmesi ve şerit takibinin yapılabilmesi için görüntü işleme tekniklerinden sıklıkla yararlanıldığı görülmektedir. Şerit takip sistemleri sayesinde sürüş güvenliği sağlanarak son zamanlarda gitgide artan trafik kazalarının ve sürücü hatalarının en aza indirgenmesi amaçlanmaktadır.

3.1.1. OpenCV

OpenCV, görüntü işleme, nesne algılama, yüz algılama, görüntü bölümlendirme, yüz tanıma ve daha pek çok bilgisayarla görme görevleri için en ünlü ve yaygın olarak kullanılan açık kaynak kitaplıklarından biridir (Gupta, 2021). OpenCV, optimize edilmiş C/C++ dilinde yazılmıştır ve modüler bir yapıya sahiptir (Nguyen, 2017). Geliştiriciler Python ve Java bağlamaları sağlamıştır. Okuması ve kullanması kolaydır (Gupta, 2021).

OpenCV kullanarak gerçekleştirdiğimiz bazı örnekler aşağıda gösterilmektedir:

1. Gri ölçekleme

Gri ölçekleme, RGB, HSV, vb. gibi 3 kanallı bir görüntüyü tek kanallı bir görüntüye, yani gri tonlarına dönüştürme yöntemidir. Son görüntü tam beyaz ve siyah arasında değişir. Gri Ölçeklendirmenin önemi, Boyut azaltma (3 kanalı tek kanallı bir görüntüye dönüştürme), Model karmaşıklığını azaltma vb. içerir (Gupta, 2021).

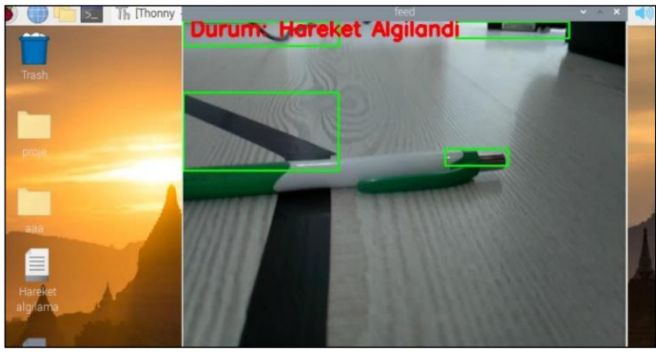
2. Yüz algılama

Yapılan çalışmada, deneme yapmak amacıyla Python'da OpenCV yüklediğimiz versiyonda gerekli kodlar girilerek ve Canny kenar algoritmasından yararlanılarak Şekil 1'deki gibi yüz algılama yapılmıştır.

Şekil 1. OpenCV'de yüz algılama örneği

3. Hareketli nesne algılama

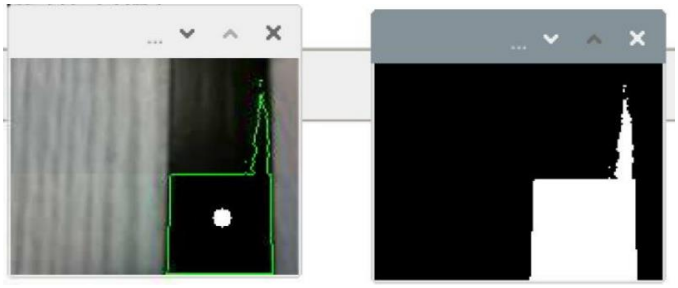
Çalışmanın hareketli nesne algılama becerisini test etmek amacıyla Python'da OpenCV yüklediğimiz versiyonda gerekli kodlar girilerek Şekil 2'deki gibi hareketli nesne algılanabilmektedir.



Şekil 2. OpenCV'de hareketli nesne algılanması

4. Şerit algılama

Yaptığımız çalışmada, OpenCV kütüphanesi yardımıyla gerekli kodlar girilerek Şekil 3'deki şerit algılaması başarılı bir şekilde gerçekleştirilmiştir.



Şekil 3. OpenCV'de şerit algılama örneği

3.1.2. Kenar algılama

Yapılan çalışmada, yol sınırlarını ve şeritleri belirlemek için bir Canny kenar algılama algoritması kullanılmıştır.

Yol sınırlarının tespiti, akıllı bir aracın yapması gereken en zor ve önemli görevdir. Yüksek gradyan değerlerine sahip pikseller, bazılarının yol sınırları koordinatlarına karşılık geldiği görüntü kenarlarıdır. Kenarları belirlemek için Sobel, Prewitt, zero cross, Roberts, Laplacian of Gaussian ve Canny gibi çalışabilecek birkaç kenar detektörü vardır (Bounini vd., 2015).

Canny kenar detektörü, görüntülerdeki çok çeşitli kenarları algılamak için çok aşamalı bir algoritma kullanan bir kenar algılama operatörüdür. John F. Canny tarafından 1986 yılında geliştirilmiştir (Rastogi, 2020).

Canny kenar algılama algoritması aşağıdaki adımlardan oluşmaktadır:

1. Gürültü azaltma: Canny Kenar algılama algoritması çok hassas çalıştığından görüntüdeki istenmeyen gürültüleri de kenar olarak algılayabilir. Bunun için yapılması gereken ilk adım "Gaussian Filtresi" uygulanarak resimdeki istenmeyen gürültülerin kaldırılmasıdır (Anonim, 2017).

2. Gradyan hesaplaması: Filtre uygulanarak düzleştirilmiş görüntü daha sonra her iki yönde birinci türevi elde etmek için hem yatay hem de dikey yönde bir Sobel çekirdeği ile filtrelenir. Bu iki görüntüden, her piksel için kenar gradyanı ve yönü aşağıdaki gibi bulunmaktadır (Rastogi, 2020):

$$Edge\_Gradient\ (G) = \sqrt{G_x^2 + G_y^2} \quad (1)$$

$$Angle\ (\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \quad (2)$$

3. Maksimum olmayan bastırma: Gradyan büyüklüğü ve yönü alındıktan sonra, kenarı oluşturmayan, istenmeyen pikselleri kaldırmak için görüntünün tam bir taraması yapılır. Bunun için her bir pikselin eğim yönündeki alanında yerel maksimum olup olmadığı kontrol edilir (Rastogi, 2020).

4. Histerezis ile Kenar İzleme: Bu aşama, hangi kenarların gerçekten kenar olduğuna ve hangilerinin olmadığına karar verir. Bunun için minVal ve maxVal olmak üzere iki eşik değerine ihtiyaç duyulmaktadır. Yoğunluk gradyanı maxVal'den fazla olan kenarlar kesinlikle kenar olarak kabul edilir. Yoğunluk gradyanı minVal'in altındakiler kenar değildir, bu nedenle atılır. İki eşik arasında kalanlar, bağlantılarına göre kenarlar veya kenar olmayanlar olarak sınıflandırılır (Rastogi, 2020).

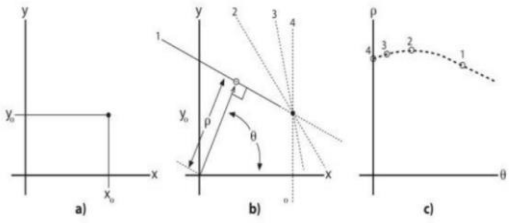
3.1.3. Hough dönüşümü

Hough Dönüşümü, bir görüntüdeki çizgiler ve daireler gibi basit matematiksel formları bulmak için popüler ve nispeten hızlı bir yöntemdir. Hough çizgi dönüşümünün arkasındaki temel fikir, ikili görüntüdeki her noktanın bazı olası çizgi kümelerine ait olabileceğidir. Genel olarak, her bir çizgi, eğim-kesme noktası biçiminde benzersiz bir şekilde bir a eğimi ve bir b kesişimi ile temsil edilir (Nguyen, 2017):

$$y = ax + b \quad (3)$$

Orijinal görüntüde 2.21 satırına ait bir noktayı (x0, y0) ele alıyoruz. Şimdi, a ve b'yi değiştirerek, (x0, y0) noktasından geçen bir dizi doğrumuz var. Böylece, (a, b) düzlemine doğru hareket eden (x0, y0) noktası, yukarıdaki doğru grubuna karşılık gelen bir noktalar kümesine (burada bir eğri haline gelir) dönüştürülür. Bu prosedürü orijinal görüntünün sıfır olmayan tüm piksellerine uygularsak ve bu tür tüm eğrileri biriktirirsek, çıktıdaki (yani (a, b) düzlemi - genellikle akümülatör düzlemi olarak adlandırılır) yerel maksimumlar girişteki çizgileri temsil eder (yani, (x, y) düzlemi). Ancak eğim-kesişim formu dikey çizgileri temsil edemez. Bu nedenle, tercih edilen temsil kutupsal biçimdir (ρ, θ), θ doğrunun açısıdır ve ρ, çizgiden koordinat orijinine olan mesafedir. Kutupsal formun denklemi (Nguyen, 2017):

$$\rho = x\cos(\theta) + y\sin(\theta) \quad (4)$$



Şekil 4. Orijinal görüntüdeki (a) bir nokta, her biri farklı bir (ρ, θ) ile parametrelendirilen birçok çizgi (b) ile geçirilebilir. (c)'de bulunan çizgiler (ρ, θ) düzlemindeki noktalarla temsil edilmektedir (Nguyen, 2017)

Hough Dönüşümü ile çizgi tespit algoritması birkaç adıma ayrılabilir (Nguyen, 2017):

1. Kenar algılama.

2. Her kenar noktası için, içinden geçen bir dizi çizgi tanımlayabilir ve onu bir eğri olarak Hough uzayına ((ρ, θ) düzlemine) eşleyebiliriz. Eğrileri saklamak için bir akümülatör kullanılır.

3. Akümülatörden yerel maksimumları çıkarmak için eşikleme gibi birkaç farklı yöntem uygulanabilir. Maksimumlar, orijinal görüntünün düz çizgilerini temsil eder.

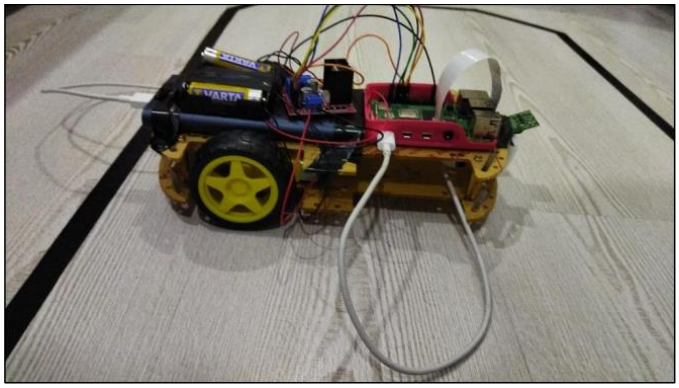
3.2. Raspberry Pi

Raspberry Pi, küçük boyutlu tek kartlı düşük maliyetli bir bilgisayardır ve mikrodenetleyici olarak kullanım esnekliği sağlamaktadır (Day vd., 2019). Gün geçtikçe modelleri yenilenen ve geliştirilen Raspberry Pi'ın akademik araştırmalarda da giderek daha fazla kullanıldığı görülmektedir. Yapılan çalışmada Arduino yerine Raspberry Pi kullanılmasının sebebi, işletim sistemi ve işlemcisinin Arduino'ya göre çok daha iyi olmasıdır. Çalışmada görüntü işleme yöntemi kullanıldığı için daha donanımlı ve hızı sebebiyle Raspberry Pi tercih edilmiştir.

VNC (Virtual Network Computing) programı ile Raspberry Pi terminaline uzaktan bağlantı sağlanılmıştır. Terminal üzerinden OpenCV kurulumu yapılmıştır.

4. Bulgular ve Tartışma

Yapılan çalışmada görüntü işleme yöntemleri kullanılarak, şerit takibi yapılması ve hareketli nesnelerin algılanması sağlanarak trafikte insan faktörünün neden olduğu kaza sebebiyetini en aza indirgemek ve otonom araçlara olan güveni artırmak hedeflenmiştir. Buna ek olarak otonom araçlara daha fazla görüntü işleme özelliği katmak ve hayatın her alanında görüntü işleme çalışmalarını artırmaya katkı sağlamak amaçlanmıştır. Oluşturulan otonom aracın görüntüsü Şekil 5'te bulunmaktadır.



Şekil 5. Otonom aracın görüntüsü

OpenCV kütüphanesi kullanılarak otonom araçlar için şerit takip sistemi ve hareketli nesne algılama sistemi uygulanmıştır.

Yaptığımız şerit takibi sisteminde iki teknik kullanılmıştır.

1- Canny kenar algılama
2- Hough dönüşümü

Gerekli OpenCV ve kamera modülü kurulumu yapıldıktan sonra Raspberry Pi'ın içine erişilerek Python kodu Thonny terminali kullanılarak yazılmıştır. Yazdığımız kodlarda Numpy ve OpenCV kütüphanelerinden yararlanılmıştır.

Şekil 6. Hazırlanan parkurda çalışmanın test edilmesi

Çalışma, Şekil 6'da görüldüğü gibi hazırladığımız parkurda 10 dakika boyunca test edilmiş ve kameranın etraftaki hareketli nesneleri istediğimiz şekilde algıladığı gözlemlenmiştir. Fakat buna ek olarak Raspberry Pi'ımızda ısınma sorunu ortaya çıkmıştır.

Assidiq (2008) yaptığı çalışmada aracın farklı yol koşullarındaki şerit tespitini ve yol görüşünü test etmiş, testlerin sonucunda gündüz %96,6 ve gece

%92,6'ya ulaşarak verimli bir şekilde tespit edebildiği sonucuna ulaşmıştır. Bizim yaptığımız çalışma sonucunda ise araç gün ışığında şerit takibinde başarılı olurken, aracın karanlık ortamda şerit takip etme özelliğini kısmen yitirdiği gözlemlenmiştir.

Literatüre bakıldığında daha önce yapılan çalışmalarda şerit takibi için Canny algoritmasına sıklıkla başvurulduğu ve başarılı sonuçlar elde edildiği görülmektedir. Örneğin; Rossi vd. (2020) ve Sevgi (2020) yaptıkları farklı çalışmalar sonucunda Canny algoritması ile şerit takibinin başarı ile gerçekleştirildiği görülmektedir. Ayrıca, Ujjainiya ve Chakravarthi (2015) elde ettikleri sonuçlar doğrultusunda Canny kenar algılama tekniğinin diğer muadillerine göre daha iyi ve okunaklı kenarlar sağladığını belirtmektedirler.

Mandlik vd. (2016) tarafından oluşturulan sistemin şerit, düz ve virajlı yolları algılayarak şeritten çıkma durumunda sürücüye uyarı verdiği sonucuna ulaşılmıştır. Bu çalışmada ise literatürden farklı olarak, aracımız düz şerit takibinde başarılı sonuçlar elde ederken keskin kenar dönüşlerinde şeridi bulmakta kısmen zorlanmıştır.

Sistemde kullanılan Pi kamera modülü sayesinde aracın etrafındaki hareketli cisimleri algılamada başarılı olduğu ve önüne çıkan engeli fark ederek hızlı bir şekilde durduğu gözlemlenmiştir.

5. Sonuç ve Öneriler

Son yıllarda otonom araç uygulamalarında artış olduğu görülmektedir. Günümüzde birçok alanda kullanılan görüntü işleme yöntemlerinin otonom araçlarda da yaygın olarak kullanıldığı bilinmektedir.

Son yıllarda yapılan çalışmalarda kullanılan görüntü işleme yöntemlerinin şerit takibi, nesne algılama, nesneleri ayırt etmek gibi işlevlerde başarılı olduğu görülmektedir.

Otonom araç uygulamaları ne kadar sürüşü ve hayatı kolaylaştırsa da sistemin karmaşık olması bazı hataların da devamında gelmesini kaçınılmaz kılmaktadır. Bu sebeple yapılan çalışmalarda daha sonradan iyileştirmeler yapılması gerekebilmektedir.

Oluşturduğumuz çalışmada Python programlama dili kullanılarak Rasperry Pi'ın içine kodlar yazılmıştır. Şerit takibi hareketli nesne algılaması ve yol sınırlarının tespiti için ise görüntü işleme yöntemlerinden olan OpenCV kütüphanesi kullanılmıştır. Araç, hazırladığımız parkurda 10 dakika boyunca test edilmiştir.

Yaptığımız çalışma için bazı iyileştirmeler ve düzenlemeler ile ilgili önerilerde bulunulabilir.

Önceki çalışmalardan farklı olarak aracın keskin virajlarda şerit bulmakta zorlandığı görülmüştür. Çalışmada daha gelişmiş Pi kamera modülü (kızılötesi vb.) kullanılması daha iyi sonuçlar alınmasını sağlayacaktır.

Hazırlanan parkurda karanlık ve aydınlık bölgeler oluşturularak araç 10 dakika boyunca test edilmiştir. 10 dakikalık süre içerisinde aracın gittiği 26 turun 21'inde şerit takibini yapabildiği gözlemlenmiştir. Sonucun %80,7'lik başarı oranına sahip olduğu görülmektedir. Karanlık bölgelerdeki kat ettiği mesafenin süresi ise aydınlık bölgeye göre daha yüksek olduğu gözlemlenmiştir.

Sonraki çalışmalarda aracın daha geniş bir parkurda ve uzun süreli test edilmesi daha net sonuçlara ulaşmayı sağlayacaktır.

6. Teşekkür

Bu çalışma TÜBİTAK BİDEB birimi tarafından "2209-A Üniversite Öğrencileri Araştırma Projelerini Destekleme Programı" kapsamında maddi olarak desteklenmektedir. TÜBİTAK BİDEB birimine desteklerinden dolayı teşekkür ederiz.

7. Kaynaklar

Anonim, 2017. OpenCV Dersleri (Ders:16) Canny Kenar Algılama. http://mavienginberk.blogspot.com/2017/06/opencv-dersleri-ders16-cannykenar.html (Son erişim tarihi: 30.05.2022)

Ariyanto, M., Haryanto, I., Setiawan, J. D., Munadi, M., and Radityo, M. S., 2019, November. Real-time image processing method using raspberry Pi for a car model. In 2019 6th International Conference on Electric Vehicular Technology (ICEVT) (pp. 46-51). IEEE.

Assidiq, A. A., 2008. Vision-based road lane detection for autonomous vehicles (Master's thesis, Gombak: International Islamic University Malaysia, 2008).

Bingöl, M. S., Kaymak, Ç., ve Uçar, A., 2019. Derin öğrenme kullanarak otonom araçların insan sürüşünden öğrenmesi. Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 31(1), 177-185.

Bounini, F., Gingras, D., Lapointe, V., and Pollart, H., 2015, October. Autonomous vehicle and real time road lanes detection and tracking. In 2015 IEEE Vehicle Power and Propulsion Conference (VPPC) (pp. 1-6). IEEE.

Day, C., McEachen, L., Khan, A., Sharma, S., and Masala, G., 2019, September. Pedestrian recognition and obstacle avoidance for autonomous vehicles using raspberry Pi. In Proceedings of SAI Intelligent Systems Conference (pp. 51- 69). Springer, Cham.

Fernandes, S., Duseja, D., and Muthalagu, R., 2021. Application of Image Processing Techniques for Autonomous Cars. Proceedings of Engineering and Technology Innovation, 17, 1

Gupta, A., 2021. Top Python Libraries For Image Processing In 2021. https://www.analyticsvidhya.com/blog/2021/04/top-python-libraries-forimage-processing-in-2021/ (Son erişim tarihi: 30.05.2022)

Küçük, Ö., Yavşan, E., ve Gökçe, B., 2021. Otonom Tabanlı İşaret ve Şerit Tanımak Amacı ile Bir Öğrenme Sisteminin Geliştirilmesi. International Journal of Engineering Research and Development, 13(3), 19-25.

Mandilik, P. T., and Deshmukh, A., 2016. Raspberry-pi based real time lane departure warning system using image processing. International Journal of Engineering Research and Technology, 5(06), 755-762.).

More, C. S., Debbarma, S., Kandpal, N., and Singh, V., 2019. Open CV Python Autonomous Car. People, 6(01).

Nguyen, T.B., 2017. Evaluation of lane detection algorithms based on an embedded platform. (Master’s thesis, Chemnitz University of Technology, Chemnitz, Germany.)

Panfilova, E., Shipitko, O. S., and Kunina, I., 2021, January. Fast Hough transformbased road markings detection for autonomous vehicle. In Thirteenth International Conference on Machine Vision (Vol. 11605, p. 116052B). International Society for Optics and Photonics.

Rastogi, A., 2020. Computer Vision: Lane Finding Through Image Processing. https://medium.com/swlh/computer-vision-lane-finding-through-imageprocessing-516797e59714 (Son erişim tarihi: 30.05.2022)

Rossi, A., Ahmed, N., Salehin, S., Choudhury, T. H., and Sarowar, G., 2020. Real-time lane detection and motion planning in Raspberry Pi and Arduino for an autonomous vehicle prototype. arXiv preprint arXiv:2009.09391.

Schwarting, W., Alonso-Mora, J., and Rus, D., 2018. Planning and decision-making for autonomous vehicles. Annual Review of Control, Robotics, and Autonomous Systems, 1(1), 187-210.

Seçkin, M. E., 2021. Derin öğrenme kullanılarak trafik koşullarına uygun otonom araç uygulaması (Doctoral dissertation, Bursa Uludag University (Turkey)).

Sevgi, A., 2020. Batarya elektrikli aracın derin öğrenme ve görüntü işleme ile otonom denetiminin gerçekleştirilmesi (Master's thesis, Lisansüstü Eğitim Enstitüsü).

Tian, D., 2019. DeepPiCar-Part 1: How to Build a Deep Learning, Self Driving Robotic Car on a Shoestring Budget. https://towardsdatascience.com/deeppicar-part-1-102e03c83f2c (Son erişim tarihi: 17.05.2022)

Ujjainiya, L., and Chakravarthi, M. K., 2015. Raspberry-Pi based cost effective vehicle collision avoidance system using image processing. ARPN J. Eng. Appl. Sci, 10(7).