T.C.
İSTANBUL GELİŞİM ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

Mekatronik Mühendisliği Anabilim Dalı

ÜRETİMDEKİ KALİTE HATALARININ GÖRÜNTÜ İŞLEME YÖNTEMİ KULLANILARAK TESPİT EDİLMESİ

Yüksek Lisans Tezi

ANIL KURU

Danışman
Dr. Öğr. Üyesi Kenan ŞENTÜRK

İstanbul – 2025

TEZ TANITIM FORMU

Yazar Adı Soyadı	Anıl KURU
Tezin Dili	Türkçe
Tezin Adı	Üretimdeki Kalite Hatalarının Görüntü İşleme Yöntemi Kullanılarak Tespit Edilmesi
Enstitü	İstanbul Gelişim Üniversitesi Lisansüstü Eğitim Enstitüsü
Anabilim Dalı	Mekatronik Mühendisliği
Tezin Türü	Yüksek Lisans
Tezin Tarihi	10.04.2025
Sayfa Sayısı	88
Tez Danışmanları	Dr. Öğr. Üyesi Kenan Şentürk
Dizin Terimleri	Görüntü işleme, Kalite hataları, OpenCV.
Türkçe Özet	Bu çalışmada Python programı kullanılarak üretim aşamasında ortaya çıkabilecek sorunların tespit edilmesi amaçlanmıştır
Dağıtım Listesi	1. İstanbul Gelişim Üniversitesi Lisansüstü Eğitim Enstitüsüne 2. YÖK Ulusal Tez Merkezine

İmzası

Anıl KURU

T. C.
İSTANBUL GELİŞİM ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

Mekatronik Mühendisliği Anabilim Dalı

ÜRETİMDEKİ KALİTE HATALARININ GÖRÜNTÜ İŞLEME YÖNTEMİ KULLANILARAK TESPİT EDİLMESİ

Yüksek Lisans Tezi

ANIL KURU

Danışman
Dr. Öğr. Üyesi Kenan ŞENTÜRK

İstanbul – 2025

BEYAN

Bu tezin hazırlanmasında bilimsel ahlak kurallarına uyulduğunu, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğunu, kullanılan verilerde herhangi tahrifat yapılmadığını, tezin herhangi bir kısmının bu üniversite veya başka bir üniversitedeki başka bir tez olarak sunulmadığını beyan ederim.

Anıl KURU

…/…/2025

T.C.

İSTANBUL GELİŞİM ÜNİVERSİTESİ

LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ MÜDÜRLÜĞÜNE

Anıl Kuru' nun **"Üretimdeki Kalite Hatalarının Görüntü İşleme Yöntemi Kullanarak Tespit edilmesi"** adlı tez çalışması, jürimiz tarafından Mekatronik Mühendisliği Anabilim Dalı Mekatronik Mühendisliği Bilim Dalı YÜKSEK LİSANS tezi olarak kabul edilmiştir.

Başkan

Dr Öğr. Üyesi Hakan KOYUNCU

Üye

Dr. Öğr. Üyesi Kenan ŞENTÜRK

(Danışman)

Üye

Dr. Öğr. Üyesi Haydar İzzettin KEPEKÇİ

ONAY

Yukarıdaki imzaların, adı geçen öğretim üyelerine ait olduğunu onaylarım.

.... /...../ 2025

Prof. Dr. İzzet GÜMÜŞ

Enstitü Müdürü

ÖZET

Bu tez çalışmasında, otomotiv üretim hatlarında kullanılan sac parçaların üzerinde bulunan somunların kalite kontrollerinin görüntü işleme yöntemleriyle otomatik olarak yapılabilirliği incelenmiştir. Özellikle, somunların boyutları ve yerleşimlerinin doğruluğunun denetlenmesi, yabancı cisimlerin tespiti ve eksik veya hatalı somunların belirlenmesi hedeflenmiştir. Görüntü işleme için OpenCV kütüphanesi kullanılarak, somunlar ve yabancı cisimler Hue,Saturation,Value (HSV) renk uzayı tabanlı bir yöntem ile algılanmış ve parça üzerinde tanımlı bölgelerdeki somunların sayısı ve boyutları analiz edilmiştir.

Tezde önerilen sistem, kameralar aracılığıyla alınan görüntülerin gerçek zamanlı işlenmesini sağlayarak, somunların istenilen ölçü ve yerleşim düzenine uygunluğunu denetlemektedir. Ayrıca, tanımlı alanların dışındaki kaliteyi etkileyebilecek yabancı cisimlerin tespitini de gerçekleştirmektedir. Bu sistem, kalite kontrol süreçlerindeki insan hatalarını en aza indirmeyi ve üretim verimliliğini artırmayı amaçlamaktadır. Görüntü işleme teknikleri kullanılarak parçalarda bulunan somun kontrolünün otomatikleştirilmesi, manuel denetimlerdeki hataların azaltılmasını ve üretim hattının daha sorunsuz işlemesini sağlamaktadır.

Anahtar Kelimeler: Görüntü işleme, Kalite hataları, OpenCV.

SUMMARY

In this thesis, the feasibility of automating the quality control of mechanical nut on sheet metal parts used in automotive production lines using image processing methods has been investigated. Specifically, the aim is to inspect the accuracy of the mechanical nut sizes and positions, detect foreign objects, and identify missing or defective mechanical nut. Using the OpenCV library for image processing, mechanical nut and foreign objects were detected through an Hue,Saturation,Value (HSV) color space-based method, and the number and sizes of nuts in defined regions on the parts were analyzed.

The proposed system in the thesis processes real-time images captured by cameras to check whether the mechanical nut comply with the desired dimensions and layout. Additionally, it detects foreign objects that may affect quality outside the defined areas. This system aims to minimize human errors in quality control processes and enhance production efficiency. By automating the mechanical nut inspection on parts using image processing techniques, the system reduces errors encountered in manual inspections and helps the production line run more smoothly

Keywords: Image processing, Quality defects, OpenCV.

İÇİNDEKİLER

ÜÇÜNÇÜ BÖLÜM
GÖRÜNTÜ İŞLEME TEKNİKLERİ

DÖRDÜNCÜ BÖLÜM
GÖRÜNTÜ İŞLEME SİSTEMİNİN YAZILIMI VE UYGULAMA SONUÇLARI

KISALTMALAR

CIE : Uluslararası Aydınlatma Komisyonu

Nm : Nanometre

RGB : Kırmızı, Yeşil, Mavi

BLOB : İkili Büyük Nesneler

HSV : Hue,Saturation,Value

OpenCV : Open Source Computer Vision Library

BMP : Bitmap

TIFF :Tagged Image File Format

GIF :Graphic Interchange Format

JPEG :Joint Photographic Experts Group

PNG :Portable Network Graphics

LZW : Lempel-Ziv-Welch

SUSAN : Smallest Univalue Segment Assimilating Nucleus

USAN : Univalue Segment Assimilating Nucle

TABLOLAR LİSTESİ

ŞEKİLLER LİSTESİ

EKLER LİSTESİ

GİRİŞ

Otomotiv endüstrisinde, üretim hatlarının verimliliği ve kalite kontrolü, rekabetçi kalabilmek ve maliyetleri düşürmek için kritik öneme sahiptir. Üretim hatlarında kullanılan her bir parçanın, belirlenen kalite standartlarına uygun olması, nihai ürünün güvenilirliği ve performansı açısından hayati önem taşır. Özellikle, sac parçalar üzerine monte edilen somunlar gibi küçük ama işlevsel bileşenlerin doğru boyut ve yerleşime sahip olmaları, üretim sürecinin sorunsuz ilerlemesi ve müşteri memnuniyeti için temel bir gerekliliktir. Kalitesiz üretimler, montaj hatalarında gecikmelere, maliyetli yeniden işleme süreçlerine ve nihai üründe ciddi sorunlara yol açabilir. Bu nedenle, kalite kontrol süreçlerinin titizlikle yönetilmesi, üretim maliyetlerini düşürmenin yanı sıra, güvenilir bir ürün sunma açısından da önem taşımaktadır. Bu çalışmada, otomotiv üretim hatlarında kullanılan sac parçalar üzerindeki somunların kalite kontrollerinin otomatikleştirilmesi ve insan hatalarının en aza indirilmesinin mümkün olduğunu göstermek amaçlanmaktadır. Bu doğrultuda, görüntü işleme tekniklerinin nasıl devreye sokulabileceği üzerine odaklanılmıştır. Görüntü işleme, üretim süreçlerinde hızlı ve hassas kalite kontrolü sağlayarak insan gözüyle yapılabilecek hataları ortadan kaldırmakta, ayrıca hızla değişen üretim ortamına uyum sağlayabilmektedir. Görüntü işleme teknolojilerinin doğru şekilde entegre edilmesi, üretim hatlarının verimliliğini ve güvenilirliğini artırmak için büyük fırsatlar sunmaktadır. Görüntü işleme, üretim hatlarında gerçek zamanlı analizler yaparak insan faktöründen kaynaklanabilecek hataları minimize edebilmektedir. Görüntülerin otomatik olarak işlenmesi sayesinde, üretim sürecinin daha kesintisiz ve verimli bir şekilde işlemesi sağlanabilir.

Çalışmanın temelinde, somunların doğru yerleşim ve boyutlarda olup olmadığını tespit edebilmek, eksik veya hatalı somunları belirlemek ve üretim sırasında oluşabilecek yabancı cisimleri algılamak için geliştirilen bir görüntü işleme sistemi yer almaktadır. Python ve OpenCv kullanılarak, somunlar ve yabancı cisimler belirlenerek, görüntü işleme algoritmaları sayesinde bu parçaların boyutları ve yerleşimleri analiz edilmeye çalışılacaktır. Yine bu çalışmada yeni sistemler sayesinde otomotiv üretim hatlarındaki kalite kontrol süreçlerini hızlandırmak, hataları minimize etmek ve nihayet üretim maliyetlerini düşürerek yenilikçi bir çözüm oluşturulması amaçlanmaktadır.

BİRİNCİ BÖLÜM
KALİTE

Kalite, Latince "nasıl oluştuğu" anlamına gelen "qualis" kelimesinden gelmektedir. (Kılıç ve Eleren,2009).Kalite, bir ürünün veya hizmetin belirli standartlara uygun Oluşturulduğunu ve beklentileri karşılayacağını emin olmakla eş değerdir. Üretim sektöründe kaliteli bir ürün yapabilmek, müşteri memnuniyetini sağlayacaktır. Müşteri memnuniyeti ile birlikte kaliteli üretim sayesinde ürün güvenilirliğinde de artış gözlemlenebilmektedir. Kalite, sadece ürün olarak değerlendirilen bir kavram değildir. Özellikle üretim yapabilmek için gerekli olan süreçlerde belirli bir standart ve kalitede ürün oluşturulması beklenmektedir. Bu standartlarda üretim gerçekleştiği zaman ortaya konulan ürünlerde aynı oranda başarılı olacaktır. Standart üretim yakalandıktan sonra ise kalite süreçlerinde sürekli iyileştirme prensibinin üretim sürecine entegre edilmesi ve eksik olabilecek noktalarda bu prensibin koruması beklenmektedir.

Bu konunun literatüre kazandırılmasında önde gelen bilim insanlarının kalite için yaptığı tanımlama şu şekildedir;

Taguchi'ye göre kalite, ürünün sevkiyat gerçekleştirildikten sonra toplumda problemlere neden olabilecek en az zarardır. Juran'a göre kalite, kullandığı ürünün uygun olması ve aynı zamanda müşteri tatminini sağlayan ve kusur bulundurmayan üretimdir. Crosby'ye göre kalite, ihtiyaçlara uygunluk olarak tanımlanmıştır. (Güzel ve Kurşunel, 2015).

1.1. Kalite Maliyetleri

Üretim sanayisi, değişen teknolojik gelişmelere uyum sağlamak zorundadır. Bu beklentiyle birlikte üretimin sürdürülebilir olması için hesaba katılması gereken maliyetler mevcuttur.

1.1.1. Kaliteye Ulaşmak İçin Yapılacak Maliyetler

Kaliteye ulaşmak için yapılacak maliyetler stratejik bir yatırımın parçalarını oluşturur. İyi bir kalite anlayışını yakalayabilmek için öncelikli kalite terimini anlatmak ve eğer kaliteli bir ürün çıkmaz ise bunun şirket için oluşabilecek potansiyel sorunlarını ortaya koymak gerekmektedir. Şirketlerin belirlemiş olduğu bu standartlara başta kalite çalışanları olmak üzere bütün şirket çalışanlarının uyum sağlaması zorunludur. Firmalar tarafından belirlenmiş bu standartların yazılı bir şekilde

belgelenebilmesi ve belirli aralıklarla bütün personellere eğitim vererek hatırlatılması gerekmektedir. Bu şekilde çalışanlar, konu ile alakalı yetkinliklerini güncelleyebilecek olup aynı zamanda şirket içinde yaşanabilecek olan süreçlerde aktif rol alabilecektir.

Yetişmiş kalite personeline sahip olunsa bile yeterli donanım ve ekipmanların olmaması durumunda şirketin üretim sürecinde yaşanabilecek hataların önüne geçmek mümkün olmayacaktır. Bu olası hataların maliyet çıktısı olarak şirketlere olumsuz yönde etkileri olacaktır. Gelişen ileri teknolojiler sayesinde hata oranlarını düşürmek ve kaliteyi artırabilmek mümkündür. Sürekli ileri teknolojilerle desteklenen bir iyileştirme kültürüne sahip olunmazsa belirli bir zaman sonrasında, kalite maliyetlerinde iyileşme yaşanamayacaktır. Bu durum yüzünden sürdürülebilir üretim sağlanamayacaktır.

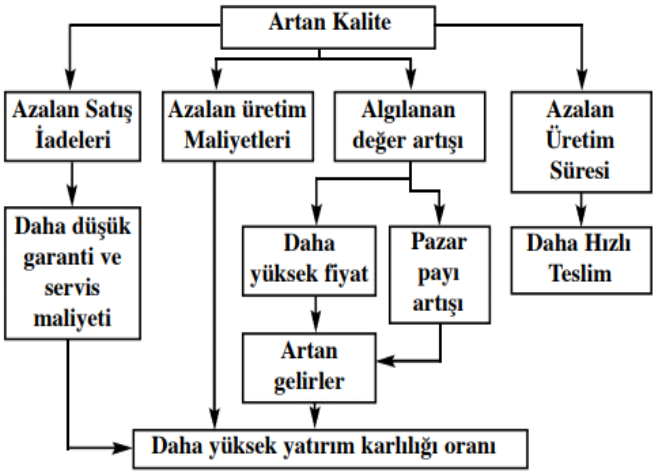
Kalite süreçlerinde dikkat edilmesi gereken önemli faktörlerden bir tanesi de sadece kalite personelleriyle yürütülmeye çalışan bu sürecin yeterli olmayacağıdır. Üretim aşamalarında yer alan üretim ve bakım bölümü başta olmak üzere şirket personellerinin aynı kültüre sahip olması gerekmektedir.

1.1.2. Kalitesizlikten Kaynaklanan Maliyetler

Saygın markaların en büyük özelliklerden bir tanesi ürünlerinde hata çıkma olasılığının az gözlemlenen bir durum olmasıdır. Otomotiv sanayisi de bu duruma önem gösteren sanayilerin başında gelir. Otomotiv parçalarında kalite standartlarını taşımayan parça gönderiminin sonuçlarını finansal anlamda ve itibar anlamında ağır sonuçlara yol açtığı bilinmektedir. Kalitesiz üretimden dolayı oluşan maliyetler, firmalar için önemli bir risk oluşturmaktadır. Kalite standartlarına uygun bir ürün tedarik edilemezse ve müşterinin beklentisine uygun olmayan ürünler hata olarak yansıtılmaktadır. Bu gibi durumların yaşanması ağır cezai yaptırımlara sebep olmaktadır.

1.1.3. Kalite Odaklı İş Süreci Ve Pozitif Etkileri

Günümüz iş dünyasında sürdürülebilir bir üretim için kaliteyi arttırmak ve müşteri memnuniyeti yakalanmak zorundadır. Bu bağlamda, birçok şirket üretim süreçlerini yüksek kalite anlayışına göre şekillendirmeye başlamıştır. Bu kalite anlayışı sayesinde şirketler, satış iadelerinin azalmasını, üretimde yaşanan hurda sorunlarının çözülmesini, şirket değerinin yükselmesini ve üretim süresinin azalmalarını hedeflemektedir (Akgün, 2005).



Şekil 1. Kalitenin yatırım etkileri

Kaynak: Akgün M, (2005, s. 33).

Şekil 1 'de belirtildiği üzere kalite kısmında yakalanan artış müşteri memnuniyetini beraber getirmektedir. Detaylı kalite kontrol süreçleri ve denetlenebilir sistemler ile daha güvenilir bir sistem ortaya konmaktadır. Yaşanan bu değer artışıyla birlikte pazar payında da artış doğru orantılı bir şekilde artacaktır. Kaliteli ürün çıkarmanın verdiği etkiyle birlikte üretim sürelerinde ve satış iadelerinde azalma gözlemlenecektir (Akgün, 2005).

1.2. Kalitenin İşletme Stratejisindeki Önemi Ve Evrimi

Rekabetin yoğun olduğu alanlarda firmaları öne geçirebilecek maddelerden bir tanesi de kaliteli üretimdir. İşletmeler ürettikleri malları hizmete sunmakla kalmayıp aynı zamanda alıcının beklentilerini karşılamak zorundadır. Bu bağlamda üretilen ürünün veya verilen hizmetin her zaman aynı veya daha önce kararlaştırılan standartlarda olması işletmeler için önemlidir. Kalite zaman zaman farklı kişiler tarafından yorumlanmıştır. Juran (1951) kaliteyi "kullanıma uygunluk" olarak tanımlarken, Crosby (1979) kaliteyi "şartlara uygunluk" olarak ifade etmiştir. Bu göreceli tanımlarda olduğu gibi işletmelerin bölümleri için bile farklılık gösterebilmektedir. Örneğin, üretim bölümü için kalite zamanında ve doğru ürünün çıkması olarak yorumlanabilirken, bakım bölümü için malzemenin dayanıklılığı, kullanılabilirlik süresi ve montaj kolaylığı olarak tanımlanabilmektedir (Kefe ve Tanış, 2014).

Geçmiş yıllarda kalite anlayışı deneylerle başlamış, ardından kalite kontrol ve kalite güvencesi gibi kavramlara evrilmiştir. Bilinçsizce yapılan kontroller zamanın geçmesiyle birlikte firmalar tarafından belirlenen özellikler ve belirli şartlara uyum sağlanması olarak dönüşmüştür.1980'lerden itibaren birçok firmanın yönetim desteğiyle bu ilkeleri kabul etmesinin yanı sıra, bu süreçleri düzeltici faaliyet olarak gören ve önem vermeyen firmalarda azımsanmayacak düzeydedir (Kefe ve Tanış, 2014).

Kalite anlayışının değişimiyle birlikte sadece üretim süreçlerine odaklanılmamış aynı zamanda tüketicilerin isteklerine ve rekabet avantajı sağlanılmaya da özen gösterilmiştir. İçinde bulunduğumuz yüzyılda ise bu durum bir ileri seviyeye taşınarak firma göstergesi olarak sunulmakta ve bu çalışmalar için yatırım yapılmaktadır (Kefe ve Tanış, 2014).

İKİNCİ BÖLÜM
GÖRÜNTÜ İŞLEMENİN TEMEL YAPISI VE LİTERATÜR TARAMALARI

2.1. Görüntü İşleme Ve Kalite Hakkında Yazılmış Literatür Taraması

Görüntü işleme ve kalite genel olarak ayrı olarak değerlendirilip yeni bir teknoloji olarak görülse de bu konu ile alakalı yapılmış birçok çalışma mevcuttur. Teknolojisinin gelişmesi ve ihtiyaçların doğrultusunda bu konuyla alakalı yapılan çalışmalar daha çok bir problemin önüne geçmek için ortaya çıkmış olup aynı zamanda iki kavramında aynı noktada kesişebileceği görülmüştür.

(Bayram,2019). tarafından gerçekleştirilen çalışmada Raspberry Pi aygıtı kullanılarak metal parçalardaki hataların bulunması amaçlanmış, morfolojik işlemler ve Hough dönüşümü kütüphaneleri ile gerçekleştirilen testler sonucunda %80'in üzerinde bir başarı elde edilmiştir. Aynı ortam şartlarında olmak koşulu ile kamera çözünürlük oranlarının değişmesinin hata tespit başarısına doğrudan etki ettiği ve çözünürlüğün hata tespit oranıyla doğru orantılı bir şekilde arttığı gözlemlenmiştir.

(Kınalı,2009). tarafından gerçekleştirilen çalışma ile anti geometrik yayınım algoritmasının, tezde uygulanan proje haricinde başka yüzey hatalarının tespitinde de etkili bir şekilde kullanılabileceği ve görüntü işleme sistemlerinin otomasyon sistemleri ile birlikte uyumlu bir şekilde çalışılabileceği ortaya konmuştur.

(Boyacıgil,2022). tarafından gerçekleştirilen bu çalışma ile uygulanan akıllı kontrol sistemleri sayesinde olası gelen hatalı parçaların hata tespitinin süresi azaltılmış olup, insan faktörünün ortadan kaldırılabileceği ortaya konmuştur. Rasberry Pi kontrol sistemiyle uygulanan bu projede kamera çözünürlüklerinin değişmesiyle birlikte doğruluk oranlarında değişiklikler gözlemlenmiş olup, kamera çözünürlüğü arttığı durumlarda sonuçların doğruluk oranında artış gözlemlenmiştir.

(Toklu,2006). tarafından gerçekleştirilen çalışma paketleme sektöründe hatalı ürün gönderme riskini 0'a indirilebileceğini göstermiş olup farklı alanlarda da etkili bir biçimde kullanılabileceğini ortaya koymuştur. Aynı zamanda üretim hattı performansında da doğru veri akışı sağlanmış ve süreç otomatik bir hale getirilmiştir.

(Tarı,2020). tarafından gerçekleştirilen çalışma ile görüntü işleme sistemlerinin sağlık üretim sanayisinde verimli bir şekilde kullanılabileceği ve zamandan tasarruf

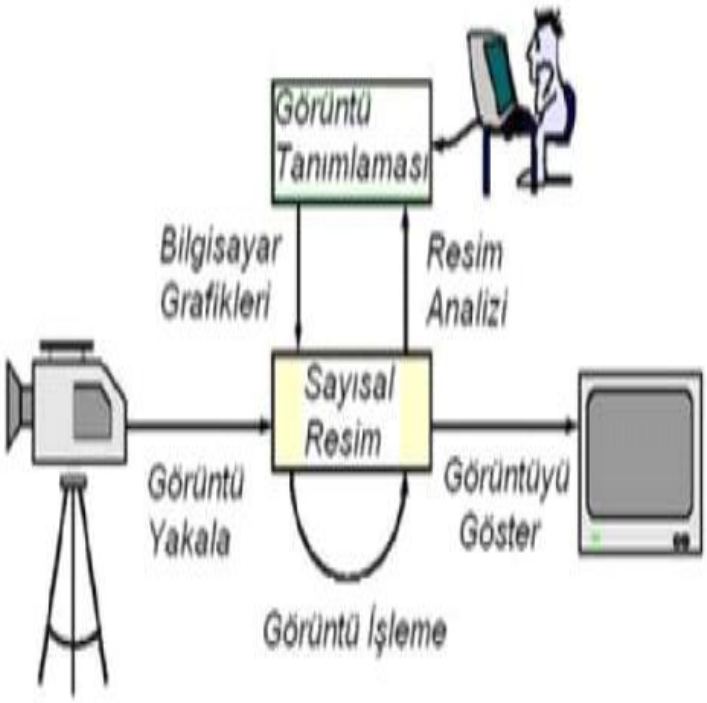
sağlandığı ortaya konmuştur. Çalışmada farklı koşullar değerlendirilmiş olup sistemin yanıt verme süresinde farklılıklar gözlemlenmiştir. Bu farklılıkların üretim sisteminde herhangi bir etkiye sahip olmadığı tespit edilmiştir. Morfolojik işlemlerden aşındırma özelliği kullanılarak yapılan çalışmalarda yapılan aşındırma sayısına göre iyileşme gözlemlenmiş ve aşındırma arttıkça doğruluk oranının da belirli düzeylerde arttığı saptanmıştır.

(Yoldaş,2021) tarafından gerçekleştirilen çalışma ile alüminyum profil sektöründe daha ince profillerin talep edilmesi sebebiyle ve geleneksel yöntemlerin daha zor olması sebebiyle görüntü işleme sistemleri kullanılmaya başlanmıştır. Yürütülen bu tez çalışmasında geleneksel yöntemler ile görüntü işleme ile yapılan çalışmalar karşılaştırılmış olup deneyi yapılan 2 yöntem de kalite kontrol işlemlerinden başarıyla geçmiştir. Görüntü işleme yöntemiyle yapılan bu projelerde zaman kazanımı olduğu ve aynı zamanda hatanın saptanma zamanının düştüğü sonucuna ulaşılmıştır.

2.2. Görüntü İşleme

2.2.1. Görüntünün Temel Yapısı Ve Oluşumu

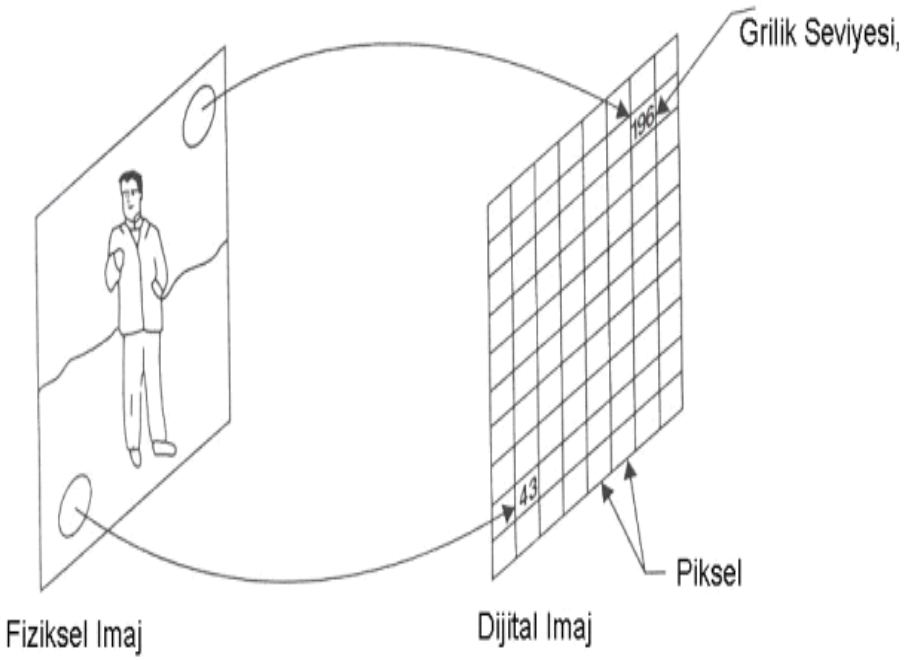
İnsan gözü 3 boyutlu olarak nesneleri algılayabilecek bir yapıdadır. İnsanlar, hücrelerini kullanarak belirli bir görüntüyü fotoseparatörler sayesinde gelen ışık uyarılarına yanıt vererek bir sinyal oluştururlar. Bu sinyaller belirli aşamalardan geçerek görme korteksine ulaşır. Görme korteksi, bu sinir sinyallerini işler ve insanlar için anlamlı hale gelebilecek bir görüntü ortaya çıkar. İnsan gözünün nesneleri algılamakta yeterli olamayacağı veya insani koşullarda çalışmanın uygun olmadığı yerlerde bilgisayar sistemleri devreye girmektedir.



Şekil 2. Görüntü işleme sistem yapısı

Kaynak: Horasan D, (2016, s. 3).

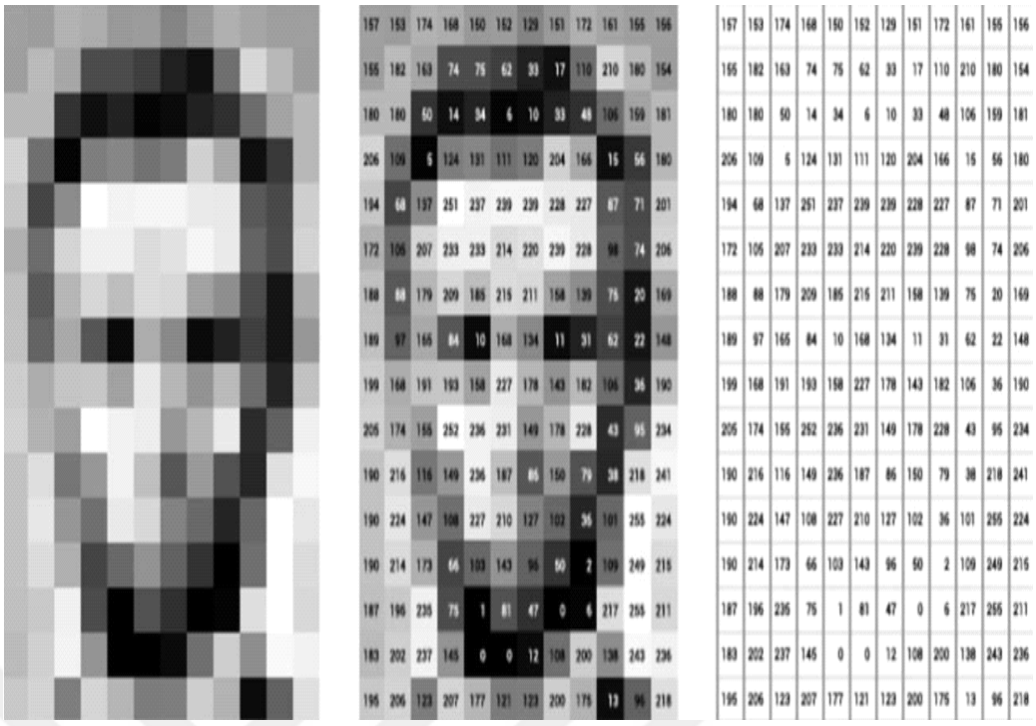
İnsan gözünün algılamasında zorluk çekeceği durumlarda kullanabilecek bilgisayar sistemleri, hızlı ve hassas bir şekilde kendine gönderilen görüntüyü analiz ederek doğru sonuçlar verebilir. Görüntü analiz sürecinin özellikle gelişmekte olan otomotiv endüstrisindeki rolü oldukça büyük olacaktır. Örneğin; saatte 2000 tane üretilen bir proseste bulunan araba sacında oluşabilecek potansiyel hataları veya kalite sorunlarını tespit etmek için görüntü işleme tekniklerini kullanmak etkili bir çözüm yolu olacaktır. Bilgisayar sistemindeki görüntülerde de benzer bir yapı mevcuttur. Şekil 3'de gösterildiği üzere kamera, fotoğraf makinesi gibi teknolojik aletler tarafından alınan analog görüntüler bilgisayarın anlayabileceği dil olan dijital formata dönüştürülür. Oluşan bu dijital görüntü analiz edilerek ve belirli filtrasyon işlemlerinden geçirilerek tekrardan insanların anlayabileceği bir yapıya ulaşır (Horasan, 2016).



Şekil 3. Fiziksel bir fotoğrafın dijital fotoğrafa dönüştürülmüş hali

Kaynak: Asmaz K, (2006, s. 2).

Bilgisayar sistemleri insan beyninden farklı olarak görüntüyü bütün olarak ele alamazlar ve görüntüleri "piksel" adı verilen bölümlere ayırmaktadır. Pikseller görüntüyü oluşturan temel yapı taşlarıdır. Bu yüzden gözümüzle gördüğümüz görüntüleri fiziksel bir resme dönüştürme işlemi ile bilgisayarların anlayabileceği bir dil olan sayısallaştırma işlemine çevirmek gerekir (Asmaz, 2006).

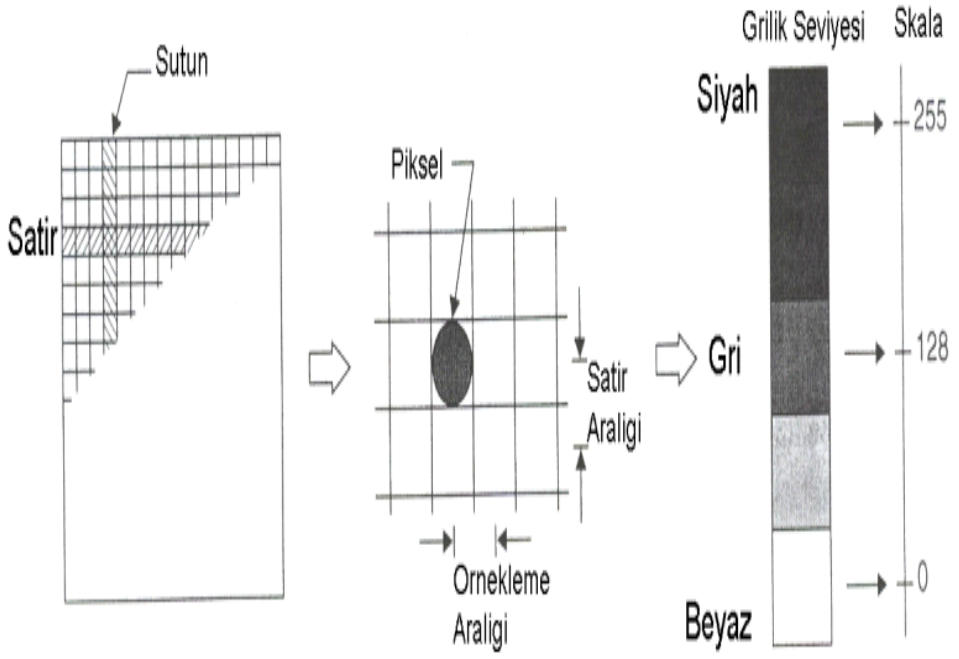


Şekil 4. Sayısallaştırılmış örnek bir görüntü

Kaynak: Çelik S, (2020).

Şekil 4'te fiziksel bir resmin bilgisayar diline çevrilmiş dijital bir görüntüsü görülmektedir. Piksellerin ilk başta parlaklık değerleri ortaya çıkarılır ve sayısal bir veriye çevrilir. Bu işlemle birlikte her pikselin renk değerleri ortaya çıkmış olur.

Her pikselin kendine özgü bir satır ve sütun numarası mevcuttur. Bu adımlar sayesinde görüntü artık işlenebilir bir biçimde olmaktadır (Asmaz, 2006).



Şekil 5. Görüntünün sayısallaştırılması

Kaynak: Asmaz K, (2006, s. 19).

Sayısallaştırma işleminden sonra ortaya çıkan bu görüntünün fiziksel görüntü olarak adlandırılan ve bilgisayar dilindeki karşılığı olan renklerin tam değeri ortaya çıkar. Bilgisayar dilindeki renkler genel olarak RGB olarak adlandırılır. RGB'nin açılımı İngilizceden gelen ve 3 ana renk olarak kabul edilen kırmızı, yeşil, mavi değerlerinden oluşur. Bu renk değerlerinin ortaya çıkması ile oluşan bütün bir sistemin matris düzeninde sıraya konmasıyla birlikte bir görüntü ortaya çıkmaktadır. Bu oluşturulan matrisin işlenmesiyle birlikte ortaya çıkan her yeni yapıya görüntü işleme adı verilir (Talan ve Aktürk, 2021).

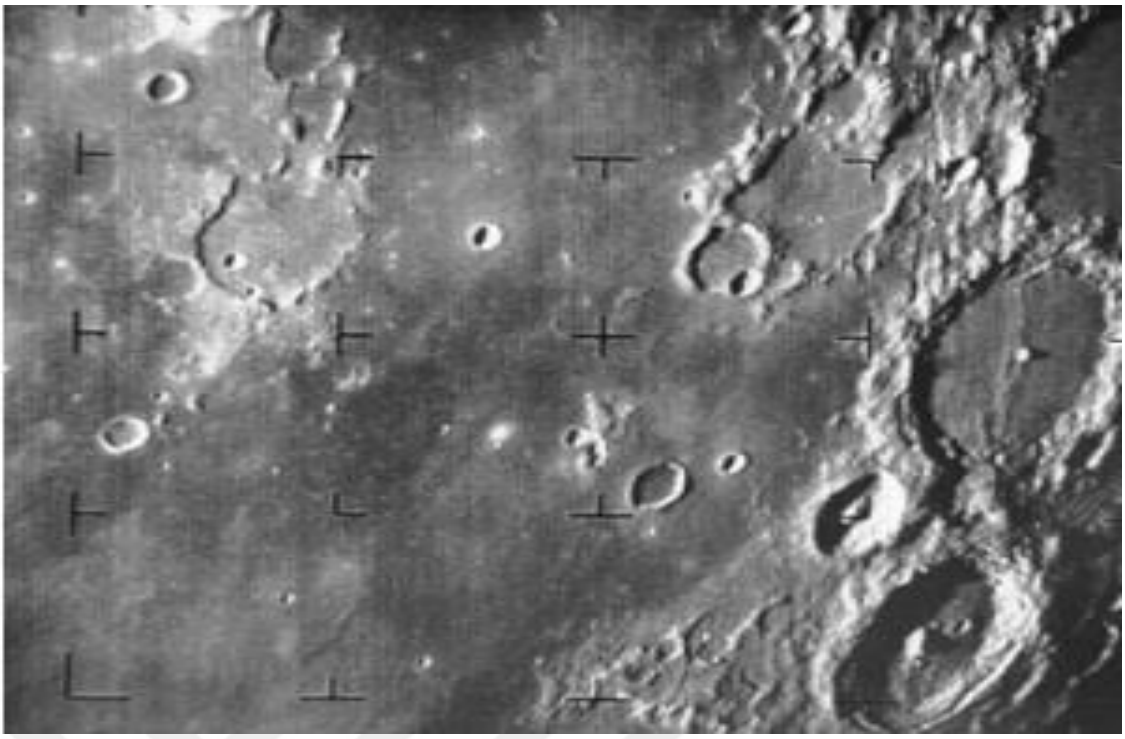
2.3. Dijital Görüntünün Tarihsel Gelişimi

Görüntünün taşınabilmesi ve işlenebilmesi ile ilgili ilk uygulamalardan biri gazete sektöründe görülmüştür. Bu uygulama, Londra'dan deniz altına döşenen haberleşme kabloları aracılığıyla verilerin iletilmesini sağlamaktadır. Gönderilen veriler, New York'taki bir merkezde çözülerek bir görüntü haline getirilmektedir. Bu

işlemin başarılı bir şekilde gerçekleştirilmesi sonucunda, 1 haftadan fazla süren resim alışveriş süreci 3 saatlik bir zamana indirilmiştir. (Gonzalez ve Wood, 2008).

Görüntü işlemede en önemli kavramların başında bilgisayar gelmektedir. Bilgisayar gelişim sürecinin başlangıcı olarak nitelendirebileceğimiz ilk olay, abaküsün yapımıdır. Son iki yüzyılda modern bilgisayarın yapımı için önemli gelişmeler yaşanmıştır. Modern bilgisayarların temeli ise 1940'lara dayanmaktadır. John Von Neumann, modern bilgisayarlara geçiş konusunda iki tane önemli temel noktaya değinmiştir. Bunlardan bir tanesi, programı ve veriyi tutabilmek için bir bellek, diğeri ise şartlı dallanmadır. Bilgisayarların şu anda milyarlarca işlemi aynı anda yapabilmesinde bu iki temelin büyük bir önemi mevcuttur (Gonzalez ve Wood, 2008).

Uzaya olan merakla birlikte, görüntülerin işlenebilmesi için gerekli olan güçlü bilgisayar sistemleri 1960'ların başında ortaya çıkmışlardır. Bu makinelerin ortaya çıkmasıyla birlikte uzay programındaki görüntülerin iyileştirme ihtiyacıyla günümüz dijital görüntü işleme mantığının temelleri atılmıştır. Ranger 7 uzay aracından gelen görüntülerin iyileştirilmesi amacıyla 1964 yılında Jet İtiş Laboratuvarı (Pasadena, California) tarafından başlatılan çalışmalar, veri toplama amaçlı gönderilen uzay araçlarından elde edilen görüntülerin kalitesini artırmaya yönelik önemli adımların atılmasına neden olmuştur. Uzay uygulamaları ile başlayan bu serüven 1970'lerin başlarıyla birlikte tıbbi görüntüleme ve astronomi alanlarında da kullanılmaya başlanmıştır (Gonzalez ve Wood, 2008).



Şekil 6. Ranger 7'den alınan ay görüntüsü

Kaynak: Gonzalez R ve Wood R, (2008, s. 5).

Teknolojinin gelişimi ile fotoğrafların da kalitesi gelişmeye başlamış ve standartlar oluşturulmuştur. JPEG, 1986 yılında ISO (Uluslararası Standartlar Organizasyonu) tarafından oluşturulan ve "Joint Photographic Experts Group" adlı bir çalışma grubu tarafından geliştirilmiştir. JPEG sisteminin geliştirilmesiyle, düşük depolama ihtiyacı sayesinde milyonlarca görüntünün aynı anda işlenebilmesi mümkün hale gelmiştir. Ayrıca, dosyaların küçük boyutları işlemlerin hızlı bir şekilde gerçekleştirilmesini ve formatın pratik bir şekilde kullanılmasını sağlamıştır. (Artuğer ve Özkaynak, 2018).

1990'lı yılların başlarında geliştirilmeye başlanan Python programının yaratıcısı Guido van Rossum'dur. İlk sürümü 1991 yılında yayınlanan nesne tabanlı program, basit ve anlaşılır bir yapıya sahip olup, aynı zamanda geniş bir kütüphane içeriği sunmaktadır. Bu geniş kütüphane özellikleri, Python yazılım dilinin gelişmesinde önemli bir rol oynamıştır (Saabith, Fareez ve Vinothraj, 2019).

Python açısından önemli gelişmelerden biri 1999 yılında Intel tarafından kurulan ve Opencv adı verilen kütüphanenin verimli bir şekilde görüntüleri analiz edebilmesidir. Bu sistem sayesinde görüntü işlemede büyük bir kilometre taşı daha geçilmiştir. 2000 yılında açık kaynaklı olarak yayınlanan kütüphane, sürekli gelişerek

ve etki alanını genişleterek güncelliğini korumaktadır. Bu güncelleştirmeler ile görüntü işleme alanında sayılı kütüphaneler arasına girmiştir. 2012 yılında Opencv.org çatısı altında toplanmış ve açık kaynaklı bir proje olarak sürekli hale gelmiştir. Ayrıca Google, Microsoft ve Intel gibi şirketlerden aldığı destekle endüstride güvenilir ve kullanılır bir hale gelmiştir (Alvi, 2013).
Bu gelişmeler sayesinde, günümüzde yoğun ve etkin bir şekilde kullanılmaya devam etmektedir.

2.4. Görüntü formatları

Günümüzde kullanılan görüntü formatları ve uzantıları şunlardır.

- BMP (Bitmap; .bmp)
- TIFF (Tagged Image File Format; .tif)
- GIF (Graphic Interchange Format; .gif)
- JPEG (Joint Photographic Experts Group; jpg, jpeg)
- PNG (Portable Network Graphics; .png)

2.4.1. BMP

Microsoft'un Windows Bitmap (BMP) dosya formatı, Windows ortamında dijital görüntüler için temel bir format olarak kabul edilmektedir. Bu format, Windows'un yerel dosya formatıdır ve bu nedenle çoğu Windows tabanlı uygulama bu formatı desteklemektedir. BMP formatı, Windows'un zamanla gelişmesiyle birlikte değişiklikler göstermiştir ve farklı BMP dosyası versiyonları mevcuttur. BMP dosyalarında piksel verilerinin satırlarının ters yönde depolandığını bilmek oldukça önemlidir. Ekranda en üstte görünen satır, bitmap dosyasında aslında en altta yer almaktadır (Anand, 2011).

Başlık kısmı, dosyanın yapısını tanımlayan ve görüntünün özelliklerini belirten bilgileri içerir. Bu bilgiler arasında görüntünün genişliği, yüksekliği, renk derinliği, piksel sırası ve sıkıştırma türü gibi önemli unsurlar yer alır. Örneğin, bir BMP dosyasının başlığı, 14 baytlık bir dosya başlığı ve ardından 40 baytlık bir bilgi başlığı içerir. Bu yapı, dosyanın işlenmesini kolaylaştırırken, farklı yazılımların BMP dosyalarını doğru bir şekilde okumalarını sağlar (Bourke,1998).

Piksel veri bölümü, görüntünün asıl içeriğini temsil eder. Her pikselin renk bilgisi, genellikle RGB formatında saklanır. BMP formatı, farklı renk derinliklerini

destekler. Her piksel için kullanılan bit sayısına bağlı olarak değişir. 1 bit, 2 renk (siyah ve beyaz) anlamına gelirken, 24 bit, toplamda 16.7 milyon farklı rengi temsil edebilir. Bu geniş renk yelpazesi, BMP formatını çeşitli uygulamalar için uygun hale getirir (Bourke,1998).

BMP'nin önemli avantajlarından biri, sıkıştırmasız bir format olmasıdır. Bu durum, görüntü kalitesinin bozulmadan saklanmasını sağlar. Ancak, sıkıştırmasız olmanın bir adet dezavantajı vardır; bu da dosya boyutunun büyük olmasına yol açmasıdır. Bu nedenle, BMP formatı, yüksek görüntü kalitesi gerektiren uygulamalar için uygun olsa da büyük dosya boyutları nedeniyle genellikle sıkıştırmalı formatlara, örneğin JPEG veya PNG gibi alternatiflere göre tercih edilmez

2.4.2. TIFF

TIFF (Tagged Image File Format), yüksek kaliteli görüntüleri kayıpsız bir şekilde saklama amacıyla kullanılan, esnek ve çok yönlü bir dosya formatıdır. Aldus Corporation tarafından 1986 yılında Adobe ile iş birliği içinde geliştirilen bu format, baskı öncesi hazırlık ve arşivleme gibi alanlarda yaygın olarak tercih edilir. Tagged (etiketli) kelimesi, TIFF'in karmaşık yapısını ifade etmektedir.TIFF formatı, görüntü kalitesini koruyarak detaylı dosya saklama özelliğine sahiptir (Wiggins,Davidson, Harnsberger, Lauman ve Goede, 2001).

TIFF 5.0 sürümü 1988 yılında yayımlanarak, popüler bir sıkıştırma algoritması olan LZW sıkıştırma tekniğini desteklemeye başladı. TIFF dosyalarının dikkat çeken bir diğer özelliği ise, tek bir dosya içerisinde birden fazla görüntüyü saklayabilmesidir (Wiggins vd., 2001).

TIFF'in başlıca dezavantajı ise, kayıpsız sıkıştırma yöntemleri ve veri aktarımını sağlamak için kullanılan etiketlerin, dosya boyutlarını oldukça büyük hale getirmesidir. Bu büyük dosya boyutu, özellikle internet tabanlı ortamlarda TIFF formatının kullanımını zorlaştırır ve performans düşüşüne neden olabilir (Wiggins vd., 2001).

2.4.3. GIF

GIF, 1987 yılında CompuServe tarafından geliştirilmiş ve özellikle internet ortamında görüntülerin kolayca paylaşılması için popüler hale gelmiş bir görüntü formatıdır. GIF, 256 renk içermektedir. Bu da onu daha az detay ve renk barındıran grafikler ve ikonlar için uygun kılar. GIF'in öne çıkan bir özelliği, kayıpsız sıkıştırma

sağlayan LZW algoritmasını kullanmasıdır, bu sayede görüntü kalitesinden ödün vermeden dosya boyutu azaltılır GIF, özellikle statik görüntülerin yanı sıra animasyonlu görseller için de kullanılır; birden fazla kareyi ardışık olarak oynatarak basit animasyonlar oluşturabilir. Dosya boyutunun küçük olması, geniş çapta desteklenmesi ve internet tarayıcılarıyla uyumlu çalışması nedeniyle GIF, hala web üzerinde sıkça tercih edilmektedir. Özellikle sosyal medya platformlarında, mesajlaşma uygulamalarında ve forumlarda yaygın olarak kullanılır (Frandsen ve Zhang, 2014).

Ancak GIF kullanılırken bazı zorluklarla karşılaşabilir. Özellikle, hesaplama maliyetlerinin yüksek olması ve gürültüye duyarlılık gibi faktörler, algoritmanın doğruluğunu etkileyebilir. Yüksek çözünürlükteki görüntülerde çalışırken, işlem süreleri uzayabilir ve bu da gerçek zamanlı uygulamalarda sınırlamalara yol açabilir.

2.4.4. JPEG

JPEG formatı, özellikle dijital fotoğrafçılık ve web için en çok kullanılan görüntü dosyası formatlarından biridir. 1986 yılında oluşturulan JPEG, görüntülerin sıkıştırılmasında ve depolanmasında yüksek bir verimlilik sağlayarak geniş bir kullanım alanına sahip olmaktadır. Bu formatın temel avantajı, kayıplı sıkıştırma algoritması sayesinde dosya boyutunu önemli ölçüde küçültmesidir. Görüntülerin daha az depolama alanı kaplamasını sağlar ve bu da hem internet üzerinde daha hızlı paylaşılabilmesini hem de depolama alanından tasarruf edilmesini mümkün kılar (Taubman ve Marcellin, 2002).

JPEG dosya formatının sunduğu sıkıştırma yöntemi, kullanıcıya dosya boyutunu dengeleyebilme imkânı sunar. Sıkıştırma düzeyi ayarlanarak görüntü kalitesi ile dosya boyutu arasında bir tercih yapılabilir. Örneğin, yüksek kalitede bir JPEG görüntü daha büyük bir dosya boyutuna sahipken, düşük kaliteli bir JPEG görüntü daha küçük olur. Ancak, kayıplı sıkıştırma tekniği, her kaydedilme işleminde görüntünün kalitesinde belirgin bir düşüşe yol açabilir (Christopoulos,Skodras ve Ebrahimi, 2000).

JPEG formatı, 24-bit renk derinliğini destekleyerek 16 milyon renk tonunu görüntüleyebilir. Bu özellik, doğal sahnelerin, manzaraların ve yüksek renk detayına sahip fotoğrafların görüntülenmesi için oldukça uygun hale getirir. Ancak, kayıplı sıkıştırma yöntemi nedeniyle, ince detaylar veya yüksek kontrastlı kenarlar gibi ayrıntılar bazen kaybolabilir. Bu durum, özellikle grafik tasarım ve baskı kalitesinin

önemli olduğu projelerde, JPEG formatını daha az kullanılmasına neden olur (Sindhu ve Rajkamal, 2009).

2.4.5. PNG

PNG, kayıpsız veri sıkıştırma sağlayan ve web üzerinde sıkça kullanılan bir resim formatıdır. 1996 yılında geliştirilmiş olan bu format, GIF formatına alternatif olarak tasarlanmıştır. PNG'nin en önemli avantajlarından biri, kullanıcıya kayıpsız bir şekilde görüntü sıkıştırma imkânı tanımasıdır. Bu, resmin görüntü kalitesinin orijinaline sadık kalmasını sağlar ve hiçbir bilgi kaybı olmadan sıkıştırılabilir. PNG, 1 bitlik siyah-beyaz görüntülerden tam renkli (24-bit) görüntülere kadar çeşitli renk derinliklerini destekler. Bu sayede, her türlü grafik ve logo gibi görsel içeriklerin yüksek kalitede saklanmasına olanak tanır. Ayrıca, şeffaflık desteği sunarak, özellikle web tasarımında arka planın transparan olmasını gerektiren durumlar için ideal bir çözüm sunar (Mao, Hu, Zhu ve Qin, 2012).

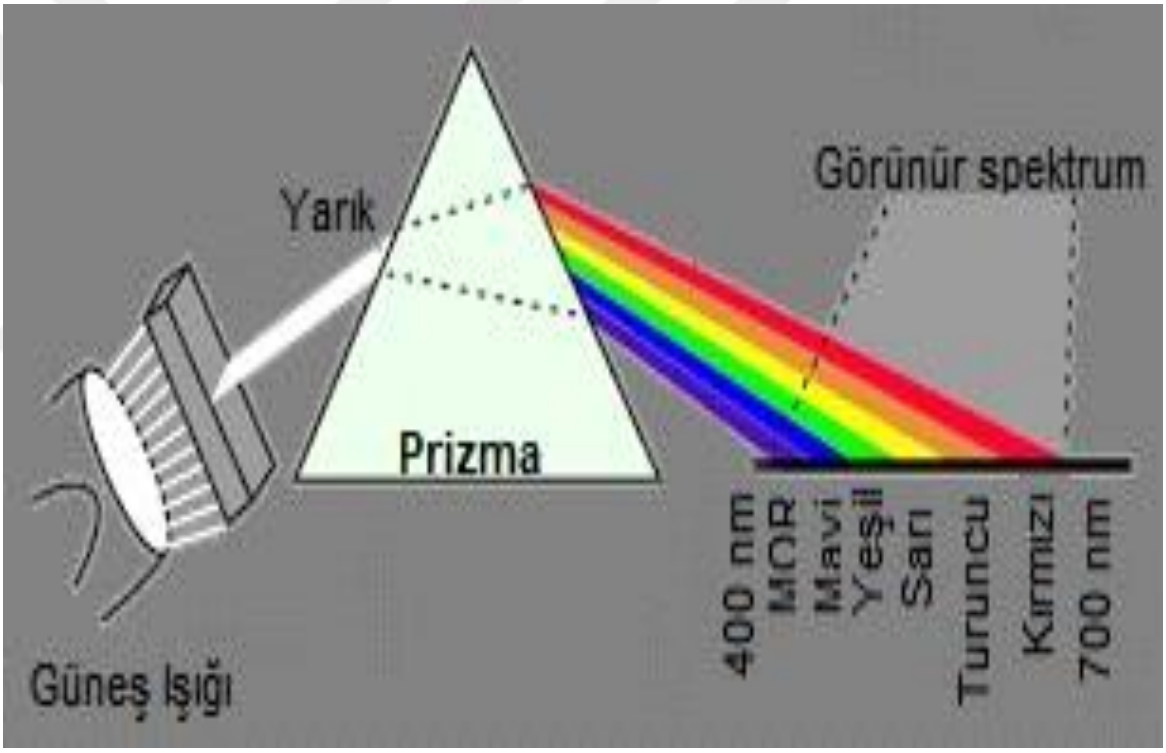
PNG formatında kullanılan interlaced sıkıştırma yöntemi, görüntülerin aşamalı olarak yüklenmesini sağlar. Bu yöntem, resmi tam olarak yüklenmeden önce, daha düşük çözünürlükte bir önizleme görüntüsü sunarak hızlı bir görüntüleme deneyimi sağlar. PNG'nin JPEG ‘ten farklı olarak, kayıplı sıkıştırma kullanmaması, onu grafiklerde ve şeffaflık gereksinimlerinde tercih edilen bir format yapmaktadır. Ancak, büyük dosya boyutları nedeniyle JPEG gibi kayıplı formatlara göre daha az verimli olabilir (Boutell,1997).

ÜÇÜNCÜ BÖLÜM

GÖRÜNTÜ İŞLEME TEKNİKLERİ

3.1. Renk Spektrumu

1931 yılında Uluslararası Aydınlatma Komisyonu (CIE) tarafından kırmızı(R), yeşil (G), mavi (B) olacak şekilde renkleri standartlaştırabilmek için özel dalga boyları belirlenmiştir. Bu dalga boyları mavi=435,8 nm yeşil =546,1 nm ve kırmızı =700nm'dir. Standartlaştırma için yapılan bu renklerden bütün renklerin çıkarılabileceği anlamı çıkmamaktadır. Birincil renkler sayesinde ikincil adı verilen macenta (kırmızı + mavi), cyan (yeşil + mavi), ve sarı (kırmızı + yeşil) üretmek için birlikte kullanılabilir (Arslan, 2011).

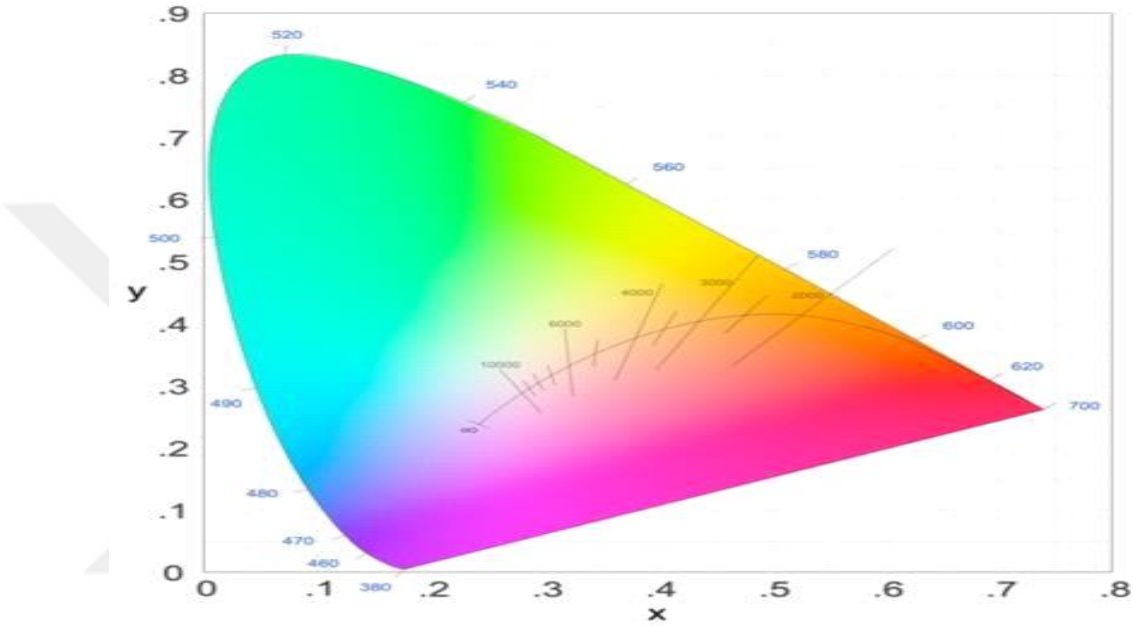


Şekil 7. Görünür ışık spektrumu

Kaynak: Pehlivan F, (2022 s. 21).

Görünür spektrumdaki herhangi bir ışık dalga boyu için, o dalga boyuna denk gelen ve renk üretmek üzere kullanılması gereken tristimulus değerleri genellikle kapsamlı deneyler sonuçlarından türetilmiş eğriler veya tablolar sonucunda ortaya çıkmaktadır. Renklerin belirlenmesinde kullanılabilecek başka bir yaklaşım ise, renk kompozisyonunu x (kırmızı) ve y (yeşil) fonksiyonları olarak temsil edilen CIE renk diyagramını kullanmaktır. Şekil 7 de gösterilen eşit enerji noktası, RGB olarak

nitelendirilen renklerin eşit bölüntüsüne karşılık gelmektedir. Bu beyaz ışığı temsil eden Uluslararası Aydınlatma Komisyonu (CIE) standardını ifade eder. Renklilik çizelgesinin sınırında bulunan herhangi bir nokta tamamen doygunluğuna ulaşmış bir renktir. Ancak bir nokta sınırı terk edip eşit enerji noktasına doğru yön aldıkça renge daha fazla beyaz ışık eklenir ve renk daha az doygun bir görüntü alır. Eşit enerji noktasında doygunluk 0'dır. Çünkü bu nokta beyaz ışığı temsil eder ve tamamen doygun bir renk içermez (Pehlivan, 2022).



Şekil 8. CIE Renk modeli

Kaynak: Pehlivan F, (2022 s. 22).

3.2. Gri Renk Dönüşümü

Günümüz dünyasında teknolojilerin ve fotoğraf makinelerinin gelişmesiyle birlikte elde edilen fotoğrafların görüntü kaliteleri sürekli gelişim göstermektedir. Bu gelişimle birlikte fotoğrafların boyutları artmakta ve görüntüler güzelleşse dahi daha karmaşık bir yapıya bürünmektedir. Bu karmaşıklık yüzünden görüntü işleme yaparken bilgisayarın bu yapıyı anlaması ve işlenmesini daha zor bir hale sokmaktadır. Renkli görüntüler bilgisayar dilinde RGB görüntü olarak nitelendirilirler. RGB görüntülerdeki baş harfler kırmızı yeşil ve mavi renklerini temsil eden 3 byte ile temsil edilirler. RGB görüntülerin gri tonlarına dönüşmesiyle birlikte bu 3 byte ile nitelendirilen görüntü artık 1 byte ile işlem görebilecektir. RGB modunda bulunan her pikselin 0-255 arasında bir değer aldığı ve RGB modundaki bir görselin gri tonlamalı

modele dönüştürülebilmesi için her pikselin 3 byte'lık değerinin işlenmesi gerektiği kabul edilmektedir. Aşağıdaki denklemler ile RGB gri dönüşümü görülmektedir (Çil, 2015).

$$x + y + z = 1 \quad (1)$$

$$\mathrm{x} * R + y * G + z * B = T \quad (2)$$

Denklem 1'de gösterildiği üzere katsayılar toplamı 1 olan x, y ve z değerleri, RGB renk modelindeki R, G ve B renk bileşenlerini ağırlıklandırarak T değerini oluşturur Bu değerin sonucunda renk değer aralığı 0-255 arasında olacaktır ve bu değer gri bir renk tonuna karşılık gelecektir (Çil, 2015).

(a) (b)

(c) (d)

Şekil 9. (a) RGB renkli görüntü (b) Açık Gri (c) Ortalama gri (d) Parlak Gri

Kaynak: Çelik A ve Tekin E, (2020, s. 262).

Görüntülerin gri tonlarına dönüştürülmesiyle birlikte resimlerin parlaklık oranları değişecek olup bu değişiklik sebebiyle görüntü işleme algoritmalarının görüntüyü işleyebilmesi ve işlenen görüntülerin depolanması konusunda avantaj sağlayacaktır. Bu avantajın küçük veri setlerindeki etkisi önemsenmeyecek düzeyde olsa bile milyonlarca fotoğrafın bulunduğu büyük veri setlerinde önemi artacaktır.

X, y ve z 'nin alacağı değerlere göre başarılı olabilecek birden çok yöntemle griye dönüşüm işlemleri yapılabilir.

3.2.1. Ortalama Değer Yöntemi

Gri tonlamayı elde edebilmek için her bir pikseldeki x, y ve z değerlerinin ortalaması alınır. Ortalama değer yöntemi genel olarak bütün piksellerin değeri aynı ise tercih edilmektedir. Fakat yöntemin çalışabilmesi için kesin bir şart değildir.

$$T = \frac{R+G+B}{3} \tag{3}$$

Renkler arasındaki dengesizlikleri ve hızlı bir sonuç elde edilmek istendiği zaman bu yöntem kullanılabilir. Fakat insan gözünün bütün renkleri aynı oranda algılayamadığı bilinmektedir. Bu nedenle, her renk kanalını aynı ağırlıkta kullanmak bazen doğal olmayan gri tonlamalarına yol açabilir.

3.2.2. Parlaklık Yöntemi

Parlaklık yönteminde CIE standartları tarafından tanımlanan ve insan gözünün parlaklık değişimlerine duyarlılığını yansıtan spesifik katsayılarla belirlenir. Bu ağırlıklar RGB renk modelindeki kırmızı (R), yeşil (G), (B) mavi kanalları için sabit değerlerdir ve denklem 4 'de belirtilmiştir (Ahsani, Sari ve Adikara, 2019).

$$T = 0{,}299 * R + 0{,}587 * G + 0.114 * B \tag{4}$$

Bu yöntem sayesinde renkler arasındaki kontrast korunur ve bu sayede belirgin detayların kaybolmaması sağlanır. Bu yöntem insan gözü için en iyi değerlerle seçilmiş olsa bile yüksek kontrastlı ve belirgin renkler içeren görüntülerde renk ayrımı konusunda dezavantaj sağlayabilir.

3.3. Kenar Belirleme

Görüntü işleme ve bilgisayarla görü süreçlerinde kenar belirleme, görüntülerdeki nesnelerin sınırlarını veya kenarlarını tespit etmek amacıyla kullanılan önemli bir tekniktir. Bu yöntemler, özellikle kalite kontrol ve nesne tanıma gibi birçok

endüstriyel ve bilimsel uygulamada temel bir rol oynar. Kenar belirleme, bir görüntüdeki ani yoğunluk değişikliklerini tespit eder ve bu sayede nesnelerin şekil, boyut ve konum bilgilerini çıkarmayı kolaylaştırır.

Görüntü işlemede kenar belirleme, nesnelerin sınırlarını belirleyerek onları arka plandan veya diğer nesnelerden ayırmayı amaçlar. Kenar belirleme teknikleri, genellikle tıbbi görüntüleme, robotik, otomatik kalite kontrol, yüz tanıma, sürücü destek sistemleri ve diğer birçok bilgisayarla görü uygulamasında kullanılır. Temel kenar belirleme teknikleri, çoğu zaman daha karmaşık ve gürültülü ortamlarda yetersiz kalabilir.

Gri seviye değerleri, bir piksel üzerinde bulunan parlaklık değerini temsil eder. Bu değerin hızlı bir şekilde değişimi, nesnedeki sınırları veya görüntüdeki önemli özellikleri belirtir. Eğer bir bölgedeki piksel değerleri düşük bir değerden yüksek bir değere doğru hızlı bir değişiklik gösteriyorsa bu genellikle nesne parçasının bir kenarı olarak yorumlanabilir. Yüksek değerden düşük değere doğru ani bir hareket mevcut ise yine bu bölgenin kenarı olarak değerlendirilebileceği söylenebilir. Kenar belirleme algoritmaları bu tür değişiklikleri tespit eder ve görüntülerdeki kenarları ortaya çıkarmaktadırlar.

Kenar tespitinde kullanılan yöntemler, türevlerin yardımıyla çalışır. Birinci derece türevler, görüntülerde daha belirgin ve kalın kenarlar ortaya çıkarırken, ikinci derece türevler daha ince detaylar ve keskin ayrıntılar üzerinde etkili olur. Ancak, ikinci derece türevler, gürültüye karşı daha duyarlıdır ve küçük hataların bile vurgulanmasına neden olabilir. Sobel, Prewitt ve Canny gibi kenar belirleme yöntemleri birinci derece türevlere dayanırken, Laplace yöntemi ikinci derece türev kullanarak daha detaylı bir kenar tespiti sağlar (Öztürk ve Öztürk, 2018).

3.3.1. Sobel Filtresi

Sobel filtresi bir görüntünün griye dönüştürüldükten sonra kenarlarını belirginleştirmek için kullanılan yöntemlerden bir tanesidir. Sobel filtresini uygulayabilmek için yatay eksene Tablo 1 de ayrı bir şekilde 3x3'lük bir matris uygulanır.

Tablo 1. Piksellerden oluşan 3x3'lük bir matris

P1	P2	P3
P4	P5	P6
P7	P8	P9

Tablo 2. Sobel kernel matrisleri Gx

-1	0	1
-2	0	2
-1	0	1

Tablo 3. Sobel kernel matrisleri Gy

-1	-2	-1
0	0	0
1	2	1

Gradyan büyüklüğünü ölçerek, bir pikseldeki yoğunluk değişiminin büyüklüğü gözükebilmektedir. Bu değer ile kenarlarda oluşan yoğunluk değişimi ifade edilebilir. Her bir piksele uygulanacak Gradyan matrisin büyüklük hesabı denklem 5,6 ve 7 'de gibi hesaplanabilir;

$$Gx = -P1 + P3 - 2P4 + 2P6 - P7 + P9 \quad (5)$$

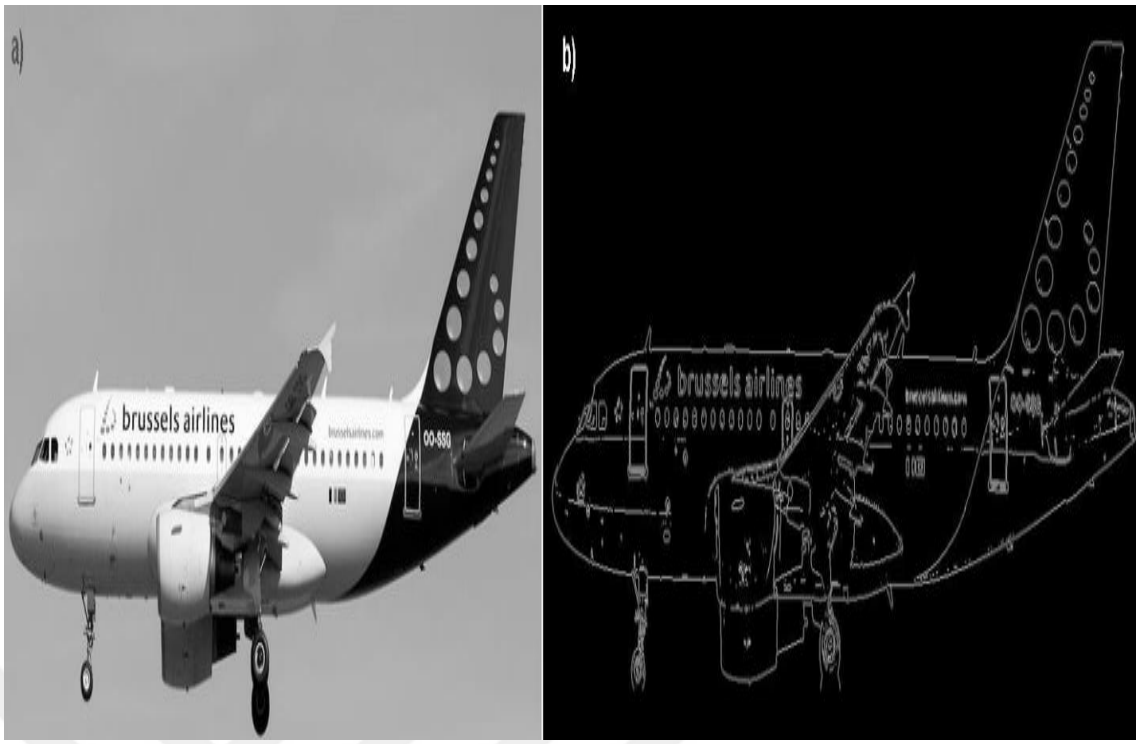
$$Gy = P1 + 2P2 + P3 - P7 - 2P8 - P9 \quad (6)$$

Son olarak ortaya çıkan matris şu şekilde olacaktır;

$$G = |Gx| + |Gy| \quad (7)$$

Gradyan büyüklüğünden elde edilen bilgileri kullanarak gradyan yönünü hesaplamak denklem 8'deki gibi mümkündür. Gradyan yönünü belirleyerek bir pikselin yönünün dikey mi yoksa yatay mı olduğu saptanmaktadır. Eğer karşılaşılan derece 90 ise dikey 0 derece ise yatay çizgi anlamına gelmektedir.

$$\theta = \arctan\left(\frac{G_x}{Gy}\right) \quad (8)$$



(a) **(b)**

Şekil 10. (a) Gri tonlamaya dönüştürülmüş resim(b) Sobel yöntemi uygulanmış resim

Kaynak: Karacan K, Uyar T ve Öztürk M. K, (2018, s. 3).

Sobel filtresi dikey ve yatay kenar kısımlarında daha fazla belirginlik arandığı durumlarda veya daha yumuşak bir geçiş istendiği durumlarda tercih edilmektedir.

3.3.2. Prewitt Filtresi

Prewitt filtresi, görüntü üzerinde uygulanırken sobel filtresine kıyasla daha hızlı çalışır, ancak genellikle daha fazla gürültü içerir ve bu nedenle daha düşük kaliteli sonuçlar üretebilir. Kenar bulma yöntemlerinden bir diğeri olan Prewitt filtresi de basit ve genel kenar yapılarını bulmak için kullanılır.

Tablo 4. Prewitt kernel matrisi Gx

-1	0	1
-1	0	1
-1	0	1

Tablo 5. Prewitt kernel matrisi Gy

-1	-1	-1
0	0	0
1	1	1

(a) **(b)** **(c)**

Şekil 11. (a) Orijinal görüntü (b) Prewitt Gx (c)Prewitt Gy.

Kaynak: Perihanoğlu G. M, (2015, s. 24).

Daha basit yapılarda, karmaşık olmayan resimlerde veya hızlı sonuç alınması gereken noktalarda bu filtreden yararlanılabilir. Bu filtreyi uygulayabilmek için dikey matriste tablo 4 matrisini ve yatay yönde tablo 5 matrisini uygulamak gereklidir.

3.3.3. Canny Filtresi

Canny, kenarları tespit etmek için bir optimizasyon süreci kullanmış ve bu sürecin bir parçası olarak, bir Gauss filtresi ile yumuşatılmış görüntüdeki gradyan büyüklüğünün en yüksek değerlerini esas alarak optimal bir dedektör önerisinde bulunmuştur (Canny, 1986).

Canny kenar dedektörü, görüntü işleme alanında en yaygın ve etkili kenar tespit algoritmalarından biridir. John F. Canny tarafından 1986 yılında geliştirilen bu

algoritma, özellikle yüksek doğruluk, iyi lokalizasyon ve tek yanıt verme özellikleriyle dikkat çekmektedir. Görüntü işleme ve bilgisayarla görme alanlarında sıklıkla kullanılan Canny algoritması, kenar tespitinin yanı sıra nesne tanıma, hareket izleme ve görüntü segmentasyonu gibi görevlerde de etkin bir şekilde kullanılmaktadır. Canny kenar dedektörünün temel ilkeleri, kullanıcıların görüntüleri daha iyi analiz etmelerine ve yorumlamalarına olanak tanır. Bu algoritmanın beş temel aşaması vardır: Gauss filtresi uygulama, gradyan hesaplama, maksimum dışlama, çift eşikleme ve histerezis ile kenar izleme (Qin, 2021). Bu aşamalar, Canny algoritmasının başarısının temelini oluşturur.

Birinci aşama olan gürültü azaltma işleminde, görüntüdeki gürültüyü minimize etmek amacıyla gerçekleştirilir. Görüntü üzerindeki gürültü, kenar tespitinde yanlış pozitiflerin ortaya çıkmasına neden olabilir. Bu nedenle, Canny algoritmasında görüntüye uygulanan Gauss filtresi, görüntüyü pürüzsüzleştirerek kenarların daha belirgin hale gelmesini sağlar. Özellikle, net görüntülerde σ değeri 0 olarak belirlenir; bu, görüntülerin pürüzsüzleştirilmesine gerek olmadığı anlamına gelir. Gürültülü görüntülerde ise, sonuçların karşılaştırılmasını kolaylaştırmak için σ değeri 1 olarak alınır (Qin, 2021). Gauss filtresi, iki boyutlu bir Gauss dağılımına dayanmaktadır ve her pikselin, komşu piksellerle birlikte bir ağırlıklandırma ile yeniden hesaplanmasını sağlar. Bu işlem, aşağıdaki matematiksel formülle tanımlanır:

$$G_{(x)} = \frac{1}{\sqrt{2\Pi\sigma}} e^{-\frac{x^2}{2\sigma}2} \quad (9)$$

Denklem 9'da G(x,y) Gauss filtresinin değeri, sigma filtre boyutunu belirleyen standart sapmadır. Bu işlem, görüntüdeki düşük frekans bileşenlerinin korunmasını sağlarken yüksek frekans bileşenlerini azaltarak gürültüyü etkili bir şekilde giderir. Gürültü azaltma aşamasının ardından, görüntüdeki kenarların belirginliğini artırmak için ikinci aşama olan gradyan hesaplama aşamasına geçilir. Bu aşamada, Sobel veya Scharr operatörleri kullanılarak her pikselin gradyan büyüklüğü ve yönü hesaplanır. Gradyan büyüklüğünün hesaplanması için aşağıdaki formüller kullanılır:

Tablo 6. Scharr Gx dikey matris

3	0	3
-10	0	10
-3	0	3

Tablo 7. Scharr Gy yatay matris

3	10	3
-10	0	10
-3	-10	-3

Burada Gx ve Gy kenarların yatay ve dikey yönlerindeki gradyanları temsil eder. Gradyan büyüklüğü, denklem 10'da bulunan formülle hesaplanır:

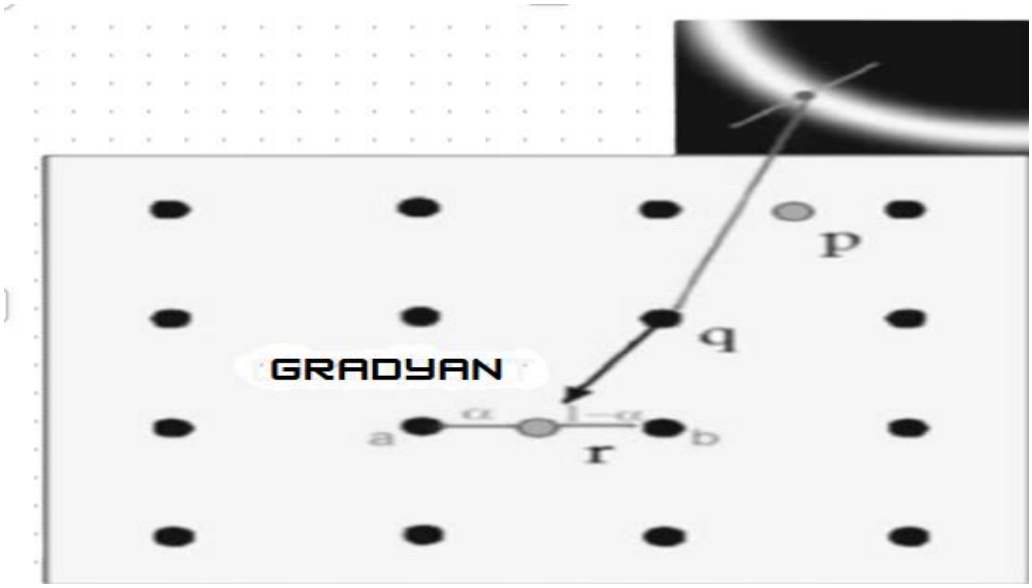
$$G = \sqrt{G_x^2 + G_y^2} \tag{10}$$

Gradyan yönü ise denklem 11'de gösterildiği gibi tanımlanır:

$$\theta = arctan\left(\frac{G_y}{G_x}\right) \tag{11}$$

Denklem 11'de iki argüman alan bir ark tanjant fonksiyonunu temsil edilir ve çıkış değerleri (-π, π) aralığında olur. Gradyan yönü genellikle 0°, 45°, 90° ve 135° gibi dört açıyı alır. (Xu, Xu ve Wu, 2017).

Gradyan büyüklüğü, görüntüdeki kenarların belirginliğini ortaya koyarken, gradyan yönü ise kenarların yönelimini belirtir. 3. Aşamada, kenar yönleri belirlendikten sonra, maksimum olmayan bastırma işlemi uygulanması gerekmektedir. Bu işlem, kenar yönünde iz sürerek kenar olarak kabul edilmeyen herhangi bir piksel değerini 0'a eşitleyerek bastırmak için kullanılır ve çıktı görüntüsünde ince bir çizgi oluşturur.



Şekil 12. İnterpolasyon ile baskılama diyagramı

Kaynak: Qin X, (2021, s.5).

Şekil 12'de, piksel q'nun yoğunluğunun, q'nun gradyan yönündeki pikseller olan p ve r piksellerinden daha büyük olduğu durum söz konusudur. Eğer bu koşul doğruysa, piksel korunur; aksi takdirde, pikselin değeri sıfırlanır (kara piksel haline getirilir). Maksimum olmayan bastırma, daha fazla doğruluk elde etmek için pikselleri interpolasyon yaparak denklem 12'deki gibi gerçekleştirilebilir:

$$r = \alpha b + (1 - \alpha) \quad (12)$$

Gradyan büyüklüğüne göre kenar piksellerinin en yüksek değerlerini koruyarak diğer piksellerin atılmasını sağlar. Bu işlem, yalnızca belirgin kenarların belirlenmesini sağlarken, zayıf ve belirsiz piksellerin göz ardı edilmesine olanak tanır. Maksimum dışlama aşaması, algoritmanın doğruluğunu artırarak yanlış pozitiflerin sayısını azaltır.

Dördüncü aşama olan çift eşikleme, belirlenen iki eşik değeri kullanılarak piksellerin güçlü, zayıf ve gürültü olarak sınıflandırılmasını sağlar. Güçlü kenarlar, yüksek eşik değeri ile belirlenirken, zayıf kenarlar düşük eşik değeri ile tanımlanır. Çift eşikleme işlemi sırasında aşağıdaki koşullar kullanılır:

Eğer $G > T_h$ ise, piksel güçlü olarak kabul edilir.

Eğer $T_l < G < T_h$ ise, piksel zayıf olarak kabul edilir.

Eğer $G < T_l$ ise, piksel gürültü olarak kabul edilir.

Burada T_h yüksek eşik değeri, T_l ise düşük eşik değeridir. Zayıf kenar piksellerinin güçlü kenarlara bağlı olup olmadığı kontrol edilerek, nihai kenar tespit işlemi gerçekleştirilir. Bu aşama, kenarların doğru bir şekilde belirlenmesini sağlarken, tespit edilen kenarların daha anlamlı bir hale gelmesine yardımcı olur (Qin, 2021). Canny kenar filtresi, bu dört aşamanın yanı sıra, kenarların bir bütün olarak algılanmasını sağlamak için son aşama olan kenar bağlantılandırma işlemini de içermektedir. Bu aşamada, zayıf kenar piksellerinin güçlü kenarlara bağlı olup olmadığı kontrol edilir; eğer bir zayıf kenar, bir güçlü kenara komşu ise, bu zayıf kenar da kenar olarak kabul edilir.

Şekil 13. (a) Normal görüntü (b) Canny filtresi uygulanmış görüntü

Kaynak: Turan B, (2017, s. 37).

Canny kenar dedektörünün avantajları arasında yüksek doğruluk oranı ve etkili kenar tespiti yer almaktadır. Canny algoritması, sadece teorik olarak değil, pratikte de önemli bir araç haline gelmiştir. Ancak algoritmanın etkin bir şekilde kullanılması, parametrelerin dikkatlice ayarlanmasına bağlıdır. Doğru eşik değerleri ve filtre boyutlarının seçimi, algılama doğruluğunu artırmakta önemli bir rol oynamaktadır. (Turan,2017). Bununla birlikte, Canny kenar dedektörünün bazı dezavantajları da bulunmaktadır. Özellikle, yüksek gürültü oranlarına sahip görüntülerde istenmeyen sonuçlar verebilir ve bu durum, yanlış pozitif tespitlerle sonuçlanabilir. Ayrıca, çok

ince kenarların tespiti sırasında, bazı önemli detaylar kaybolabilir. (Turan,2017). Bu nedenle, Canny kenar dedektörü kullanılırken, görüntülerin ön işleme aşamalarına dikkat edilmesi gerekmektedir.

3.3.4. Laplacian of Gauss (LOG) Kenar Belirleme

Laplace algoritması, görüntü işleme ve bilgisayarla görme teknolojilerinde geniş bir uygulama alanına sahiptir. Görüntüdeki kenarları netleştirmek, nesne sınırlarını belirginleştirmek ve öznitelikleri çıkarmak için kullanılan bu yöntem, özellikle görüntülerdeki ani parlaklık değişimlerini saptamak amacıyla geliştirilmiştir (Ecemiş, Güner, Kuran ve Kuran, 2023). Laplace operatörü, görüntüde yüksek frekanslı bileşenleri, yani pikseller arasındaki ani geçişleri tespit eder. Bu sayede görüntüdeki kenarlar ve nesne sınırları öne çıkar. Laplace algoritması, bulanıklık içermeyen ve net kenarların gerektiği durumlarda son derece başarılı sonuçlar sunar. Bununla birlikte, Laplace operatörü tüm yönlerde aynı anda çalışan bir ikinci türev operatörü olduğu için, tüm kenarları aynı anda netleştirir, ancak herhangi bir yön bilgisi sağlamaz.

Laplace operatörünün matematiksel temeli, bir fonksiyonun ikinci türevini almak üzerine kuruludur. Bir görüntüyü iki boyutlu bir fonksiyon olarak düşünüldüğünde, görüntüdeki parlaklık fonksiyonunu f(x,y) ile gösterilirse, Laplace operatörü bu fonksiyonun ikinci türevleriyle ifade edilir (Gonzalez ve Wood, 2008). İki boyutlu Laplace operatörü formülü şu şekildedir:

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{13}$$

Denklem 13'de, $\frac{\partial^2 f}{\partial x^2}$ ve $\frac{\partial^2 f}{\partial y^2}$ terimleri sırasıyla yatay ve dikey eksenler boyunca alınan ikinci türevleri temsil eder. İkinci türev, bir fonksiyonun eğimindeki ani değişimleri belirlemede kullanılır. Görüntü işleme açısından bu eğim değişiklikleri, görüntüdeki kenarları ve nesne sınırlarını ifade eder. Laplace operatörü, parlaklık değişimlerini tespit etmek için tüm yönlerdeki ikinci türevleri birleştirerek ani geçişleri belirgin hale getirir.

Laplace operatörünü uygularken genellikle çeşitli filtreler kullanılır. Bu filtreler, görüntünün her pikseli için ikinci türev değerlerini hesaplar. Yaygın olarak kullanılan filtrelerden biri, 3x3 boyutundaki bir çekirdek (kernel) matristir ve bu çekirdek, görüntüdeki her piksel üzerine uygulanarak ikinci türev sonuçlarını verir. Basit bir Laplace çekirdeği örneği şu şekildedir:

Tablo 8. Laplace kernel matrisi

0	-1	0
-1	4	-1
0	-1	0

Bu çekirdek, merkezdeki pikselin parlaklık değeri ile çevresindeki piksellerin parlaklık değerlerini karşılaştırarak kenarları belirginleştirir.

Laplace operatörünün diğer popüler bir versiyonu ise merkezde -8 değeri olan ve çevresindeki piksellerin +1 değerine sahip olduğu 3x3 bir çekirdek matristir.

Tablo 9. Laplace kernel matrisi 2

1	1	1
1	-8	1
1	1	1

Bu tür filtreler, görüntünün detaylarını daha belirgin hale getirir ve hassas kenar algılaması sağlar. Ancak, Laplace operatörü yüksek frekanslı bileşenlere duyarlı olduğundan, görüntüdeki gürültüye karşı da hassastır. Bu nedenle, Laplace operatörünü uygulamadan önce genellikle düşük geçiş filtresi kullanılarak görüntüdeki gürültü azaltılır (Şenel, 2007).

Laplace algoritmasının en büyük avantajlarından biri, kenarları hızlıca belirginleştirmesidir. Diğer kenar algılama algoritmalarının aksine, Laplace operatörü tüm yönlerde aynı anda çalışarak görüntüdeki kenarları ortaya çıkarır. Bu özellik, özellikle dairesel veya karmaşık yapılar içeren görüntülerde Laplace operatörünü kullanışlı hale getirir Görüntüdeki gürültü, ani parlaklık değişimlerine sebep olduğunda, Laplace operatörü tarafından kenar olarak algılanabilir. Bu nedenle, Laplace operatörünün uygulanmasından önce veya sonra bir gürültü giderme adımı eklemek faydalı olacaktır (Şenel, 2007).

Ancak, bu yöntemin en büyük dezavantajı, keskin geçişlerin yanı sıra gürültüyü de belirgin hale getirmesidir. İkinci derece Laplace kenar bulma algoritmalarının kenar

tespit filtresi olarak nadiren tercih edilmesinin nedeni, bu algoritmaların gürültüye karşı oldukça duyarlı olmalarıdır (Onat, 2017).

3.3.5. Robert Algoritması

Roberts çapraz operatörü, bir görüntüdeki kenarları tespit etmek için kullanılan basit ve bilinen en eski yöntemlerden biridir (Turan, 2017). Bu operatör, gradyan büyüklüğünü hesaplayarak görüntüdeki yoğunluk değişimlerini belirler. Matematiksel olarak, gradyan büyüklüğü denklem 14'deki gibi ifade edilir;

$$G_f(i,j) = f(i,j) - f(i+1,j+1) + f(i,j+1) + f(i+1,j) - f(i,j) \quad (14)$$

Burada, $f(i,j)$ i ve j koordinatlarındaki piksel değerini temsil eder. Bu formül, her bir pikselin komşularıyla olan farklarını hesaplayarak, o pikselin etrafındaki kenarların gücünü belirler (Shrivakshan ve Chandrasekar, 2012).

Roberts operatörü, gradyan bileşenlerini Gx_ve Gy olarak iki ayrı yönde hesaplar. Bu bileşenler, aşağıdaki konvolüsyon maskeleri ile elde edilir:

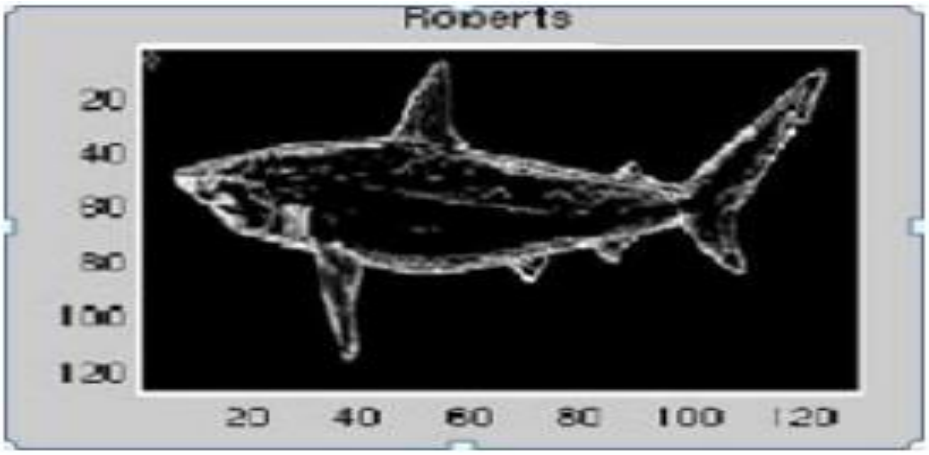
Tablo 10. x yönündeki gradyan bileşenini hesaplamak için maske

1	0
0	-1

Tablo 11. y yönündeki gradyan bileşeni için maske

0	1
-1	0

Bu maskeler, görüntünün belirli alanlarındaki yoğunluk değişimlerini ölçmek için kullanılır.



Şekil 14. Robert kenar algılama filtresi örneği

Kaynak: Shrivakshan G. T. ve Chandrasekar C. (2012, s. 271).

Görüntü işleme ve bilgisayarla görü alanında sıkça kullanılmakta olan bu filtrenin avantajları arasında basit ve hızlı uygulanabilir olması yer almaktadır. Elde edilen bu hızlı sonuçlar, görüntülerdeki önemli detayları ortaya çıkarmak için değerlidir. Örneğin şekil 14'de gösterildiği gibi bir köpekbalığı görüntüsü üzerinde bu operatörün nasıl uygulandığına dair bir örnek verilmiş ve sonuçlar gayet başarılı gözükmektedir.

Bununla birlikte, bu yöntem, diğer kenar tespit yöntemlerine göre daha az yönsel bilgi sağlamakta olup, kenarların yönlerini belirlemekte sınırlı kalabilir. Ayrıca, yalnızca 2x2'lik bir maske kullanması, daha büyük ve karmaşık kenar yapılarını yeterince tanımlayamamasına neden olabilir. Hızlı ve etkili bir kenar tespiti yöntemi sunarken, daha karmaşık ve gürültülü ortamlarda sınırlamalar olabilmektedir.

3.3.6. Harris Köşe Belirleme Algoritması

Harris köşe tespit algoritması, bir görüntüdeki köşe ve kenar noktalarını belirlemek amacıyla geliştirilmiş temel bir yöntemdir. Bu algoritma, görüntüdeki her piksel için otokorelasyon matrisi M kullanır (Onat, 2018). Bu matris, piksellerin çevresindeki küçük değişikliklere karşı nasıl tepki verdiğini ölçer ve aşağıdaki gibi hesaplanır:

$$\mathrm{M} = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \tag{15}$$

Burada, Ix ve Iy, sırasıyla görüntünün yatay ve dikey yöndeki türevleridir. Bu türevler, piksellerin çevresindeki yoğunluk değişikliklerini ölçmek için kullanılır

(Onat, 2018). Matris M'nin özdeğerleri λ_1 ve λ_2 piksellerin köşe, kenar veya düz bir bölgede olup olmadığını belirler:

Düz bölge: Eğer λ_1 ve λ_2 her ikisi de küçükse, piksel düz bir alandadır. (Onat, 2018).

Kenar bölgesi: Eğer λ_1 büyük, λ_2 küçükse veya tam tersi ise, piksel bir kenar üzerindedir. (Onat, 2018).

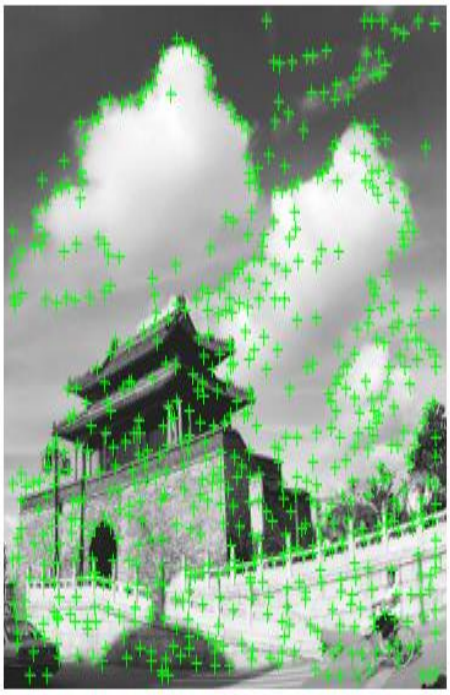
Köşe bölgesi: Eğer λ_1 ve λ_2 her ikisi de büyükse, piksel bir köşe üzerindedir. (Onat, 2018).

Harris algılama algoritmasında, her pikselin gradyanı hesaplanır. Gradyan değeri tüm yönlerde belirgin şekilde değişiyorsa, piksel köşe olarak kabul edilir (Amaricai, Gavriliu ve Oana, 2014).

$$R = det(M) - k.\left(trace(M)\right)^2 \quad (16)$$

Denklem 16'da gösterilen R, her pikselin (x, y) koordinatlarındaki köşe tepkisini ölçer. k, genellikle 0.04 olarak alınan sabit bir katsayıdır. Ix ve Iy, sırasıyla yatay ve dikey yönlerdeki birinci dereceden gri tonlama gradyanlarını ifade eder (Shen, Zhang ve Heng, 2010).

Harris algoritması, döndürme ve eğim değişmezliği gibi avantajlar sunan klasik bir yöntem olmasına rağmen, bazı önemli sınırlamaları da bulunmaktadır:



(a) (b)

(c)

Şekil 15. (a) Orjinal resim (b) Düşük thresholdlu Harris uygulanmış resim (c) Yüksek thresholdlu Harris uygulanmış resim

Kaynak: Shen vd., (2010, s. 2).

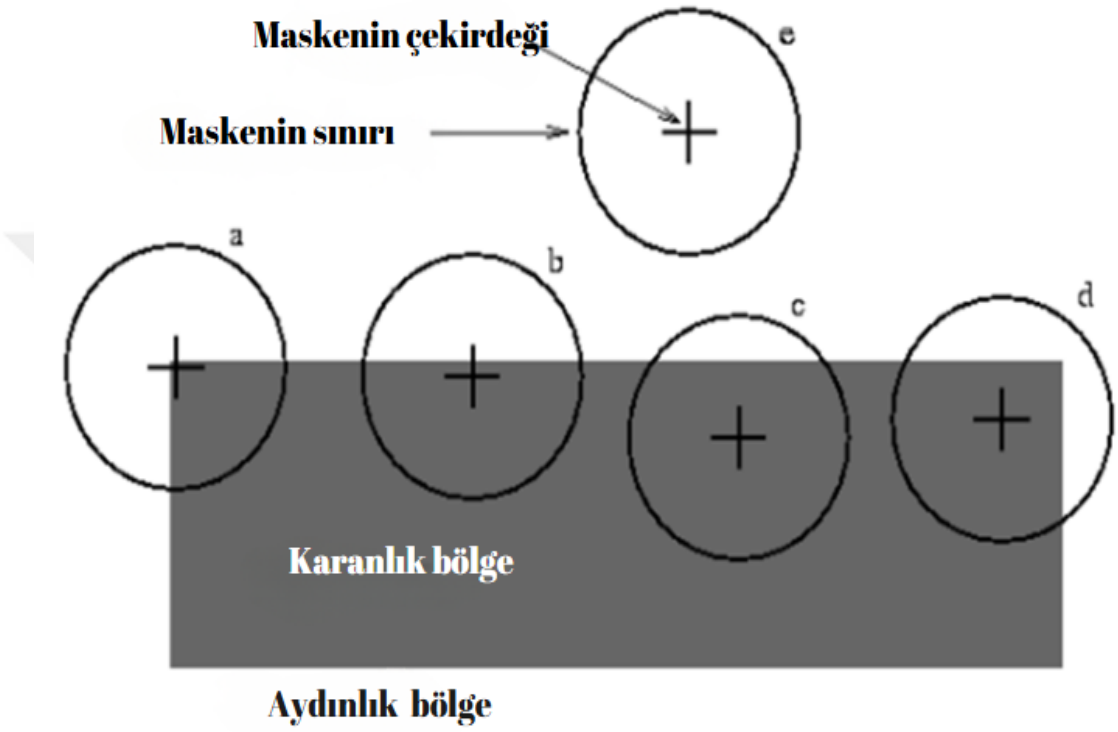
a) Harris köşe algılama performansı, büyük ölçüde kullanılan eşik değerine bağlıdır. Eşik değeri çok yüksek belirlendiğinde, tespit edilen köşe sayısı az olur; bu köşeler genellikle doğru olmasına rağmen, yeterli sayıda köşe bulmakta zorluk yaşanabilir. Tam tersi durumda, eşik değeri çok düşük olduğunda ise, sayıca fazla ama geçersiz (yanlış) köşeler tespit edilebilir (Shen vd., 2010).

b) Gri tonlama açısından belirgin bir kontrasta sahip görüntülerde, belirli bir eşik değeri bir bölgedeki köşeleri etkili bir şekilde tespit edebilirken, başka bir bölgede köşe tespitinde başarısız olabilir. Bu durum, köşe algılama sürecinin ardından gerçekleştirilecek olan kamera kalibrasyonu veya video yeniden yapılandırma gibi işlemlerde zorluk ve karmaşa yaratabilir (Shen vd., 2010).

3.3.7. Susan Kose Belirleme Algoritması

SUSAN (Smallest Univalue Segment Assimilating Nucleus) köşe tespit algoritması, bir dairesel maske kullanarak gerçekleştirilir. Bu yöntemde, maskenin içerisindeki her bir pikselin parlaklığı, maskenin merkezindeki (nükleus) pikselin parlaklığı ile karşılaştırılır. Bu karşılaştırma sonucunda, nükleusa benzer parlaklık

değerine sahip olan piksel alanı, "USAN" (Univalue Segment Assimilating Nucleus) olarak adlandırılır. Şekil 16 incelendiğinde, beyaz bir arka planda koyu bir dikdörtgen görülmektedir. Farklı pozisyonlarda yerleştirilmiş beş dairesel maske, basit bir görüntü üzerinde gösterilmiştir. USAN alanının büyüklüğüne göre köşeler tespit edilebilir. Nükleus, USAN alanının en küçük olduğu durumda bir köşe üzerindedir; örneğin, "a" pozisyonunda olduğu gibi (Chen, Zou, Zhang ve Dou, 2009).



Şekil 16. Basit bir görüntü üzerinde farklı yerlere yerleştirilmiş dört dairesel maske

Kaynak: Chen vd., (2009, s. 2).

Köşe tespiti için her maskenin içindeki piksellerin nükleusla karşılaştırılması, denklem 17'deki gibi gösterilir:

$$c(r, r_0) = exp\left(-\frac{|I(r)-I(r_0)|}{t}\right) \quad (17)$$

Burada r_0 nükleusun koordinatlarını, r, maskenin diğer noktalarının koordinatlarını, c(r, r_0), karşılaştırma sonucunu, I(r), pikselin gri değeri ve t, gürültüye karşı direnç ve SUSAN dedektörünün algılayabileceği en küçük kontrastı belirleyen gri farkı eşiğini ifade eder (Chen vd., 2009).

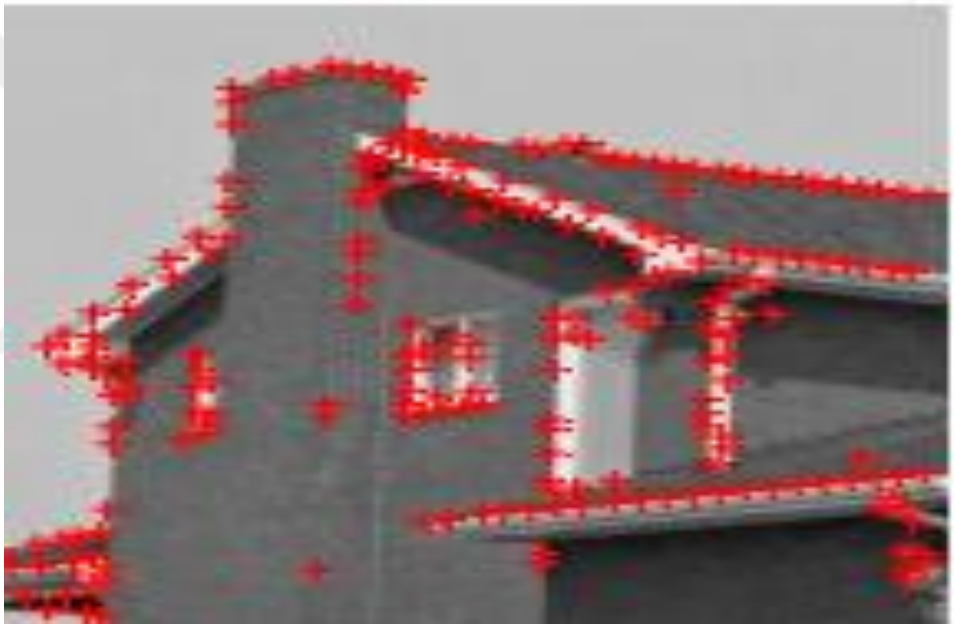
USAN bölgesinin büyüklüğü denklem 18'de belirtilen şu formül ile verilir:

$$n(r_0) = \sum_{r \in c(r)} C(r, r_0) \quad (18)$$

Bu değerler kullanılarak köşe tespitine yönelik ilk yanıt formül ile elde edilir; SUSAN'ın temel ilkesi gereğince, daha küçük bir USAN alanı, köşelere daha yüksek bir yanıt verir:

$$R(r) = \begin{bmatrix} 0, & eğer\ n(r) \geq g \\ g - n(r), & eğer\ n(r) < g \end{bmatrix} \quad (19)$$

Denklem 19'da gösterilen g, geometrik eşiği ifade eder ve köşenin keskinliğini belirler. Daha küçük bir g değeri daha keskin köşeleri ortaya çıkarır. Son olarak, köşeler maksimum olmayan baskılama yöntemi ile belirlenir (Chen vd., 2009).



Şekil 17. SUSAN köşe belirleme algoritmasıyla bir resim

Kaynak: Chen vd., (2009, s. 2).

3.4. Hough Algoritması

Hough Dönüşümü, bir görüntüde belirli şekilleri tespit etmek için kullanılan bir yöntemdir ve özellikle çizgiler ve daireler gibi geometrik şekilleri bulmada etkilidir (Parsa, Hosseinzadeh ve Effatparvar, 2015).

Bu yöntem, bir görüntüdeki nesneleri belirli bir şekil sınıfında tanımlamak için bir oylama süreci gerçekleştirir. Çizgilerin tanımlanabilmesi için öncelikle

matematiksel formüllerle ifade edilmeleri gerekir. Bir çizgi, genellikle denklem 20'de gösterilen formülle ifade edilir:

$$y = mx + c \tag{20}$$

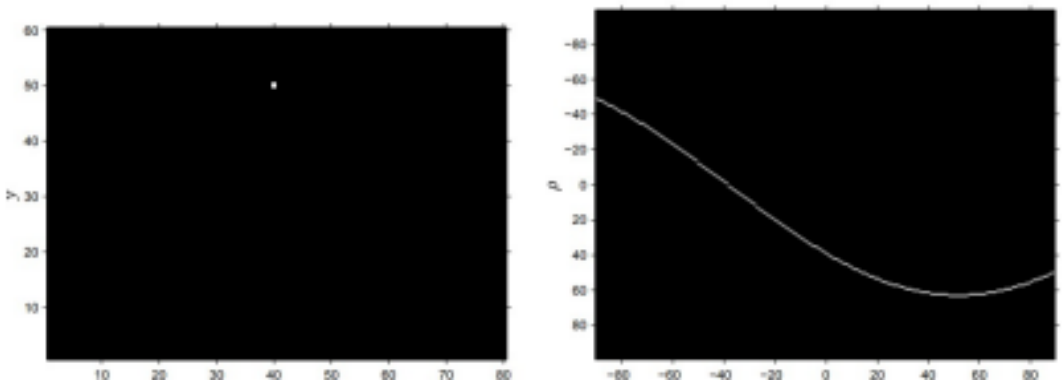
Bu formülde x ve y görüntünün yatay ve dikey koordinatlarını, m eğimi ve c kesişim noktasını temsil eder. Ancak Hough Dönüşümü, bu çizgileri daha etkili bir şekilde işlemek için alternatif bir yaklaşım kullanır. Hough uzayı, her bir çizgiyi bir nokta olarak temsil eden bir sistemdir. (Parsa vd., 2015).

$$p = x\cos\theta + y\sin\theta \tag{21}$$

Hough uzayında bir çizgi , bir parametre çiftini (ρ, θ) alır. Denklem 21 'de gösterilen ρ, orijinden çizgiye olan en kısa mesafeyi, θ ise çizginin yatay eksenle yaptığı açıyı gösterir (Lee, Wu, Park, Park ve Kim, 2015).

Görüntü alanında bir nesne noktasını (örneğin A noktası) ele aldığımızda, bu noktanın etrafında sonsuz sayıda düz çizgi geçebilir. Hough Dönüşümü, her bir aday noktayı bu çizgilere eşleyerek, görüntüdeki potansiyel çizgileri tespit etme işlemini gerçekleştirir. A noktasının Hough uzayındaki karşılığı, o noktadan geçen tüm çizgilerin denklemlerinin oluşturulmasıyla belirlenir (Lee vd., 2015).

Hough Dönüşümü, her bir aday piksel için, o noktadan geçen tüm olası çizgilerin hesaplanmasını sağlar. Hesaplamaların verimliliğini artırmak için ters bir dönüşüm de yapılabilir. Bu dönüşüm, Hough uzayındaki rastgele noktaların oylarının, o noktanın üzerinden geçen piksel sayılarak hesaplanmasını içerir (Lee vd., 2015).



Şekil 18. Hough dönüşümü yapılmış bir görsel

Kaynak: Nasseri M. H., Moradi H., Nasiri S. M., ve Hosseini, R. (2018, s. 131).

Sonuç olarak, Hough Dönüşümü, görüntüde daha fazla noktanın bulunduğu ve en uzun çizgiye karşılık gelen Hough uzayındaki en yüksek değere sahip noktanın belirlenmesiyle sonuçlanır. Bu nokta, görüntüde en çok noktayı içeren çizgi olarak kabul edilir ve böylece karmaşık görüntülerdeki çizgilerin tespit edilmesine yardımcı olur. Hough Dönüşümü, çeşitli nesne ve şekil tespiti uygulamalarında etkili bir araçtır.

3.5. Gürültü Azaltma Filtreleri

Görüntü işleme aşamalarından olan gürültü azaltma işlemleri, başarı oranı üzerinde etkisi olabilecek bir işlemdir. Filtreler kullanılarak bir görüntüde bulunan ve istenmeyen pikselleri ve detayları azaltmak bu sayede görüntü kalitesini iyileştirmek mümkündür. Filtreler genel olarak düşük ışık koşullarında olan bir resmi veya düşük çözünürlükte olan resimleri düzeltebilmek için kullanılmaktadır. Bu gürültüyü azaltabilmek için ortaya çıkan ve belirli algoritmalar ile çalışan gürültü azaltma filtreleri mevcuttur.

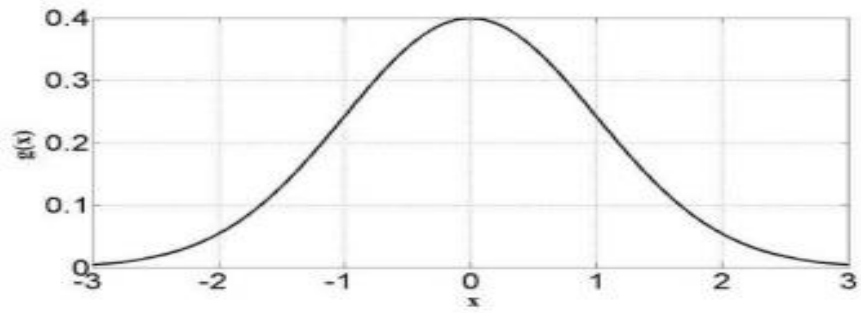
3.5.1. Gaussian Filtresi

Bu filtre, bir görüntü üzerindeki kenarları yumuşatmak ve gürültüyü azaltma işlemlerinin gerçekleştirilmesi için kullanılır. İsmi matematikte bulunan Gauss fonksiyonundan gelmektedir

Gauss filtresinde, her pikselin ağırlığı, merkez piksele olan uzaklığına ve seçilen standart sapma değerine göre belirlenir. Uzaklık arttıkça, piksellerin ağırlıkları azalır. Bu nedenle, merkezdeki piksel ve ona en yakın pikseller, yeni piksel değerinin hesaplanmasında daha fazla etkiye sahiptir (Değirmenci, Çankaya ve Demirci, 2018).

Gaussian filtresi için kullanılan bir boyutlu bir Gauss fonksiyonu denklem 22'de gösterilmiştir;

$$G_{(x)} = \frac{1}{\sqrt{2\Pi\sigma}} e^{-\frac{x^2}{2\sigma}2} \quad (22)$$



Şekil 19. Ortalaması 0 σ=1,5 olan 1 boyutlu Gaussian dağılımı

Kaynak: Çelik K, (2015, s. 21).

Gaussian filtresindeki amaç bir pikselin değerini, etrafındaki piksellerin ağırlıklı ortalaması alınarak bulunmasıdır ve bu ağırlıkların dağılımı, iki boyutlu Gauss fonksiyonu ile belirlenmektedir.

Gaussian filtresi için kullanılan iki boyutlu bir Gauss fonksiyonu denklem 23'de gösterildiği gibidir;

$$G(x,y) = \frac{1}{2\Pi\sigma^2} e - \frac{x^2+y^2}{2a^2} \quad (23)$$

Gauss filtresinde katsayıların belirlenmesinde önemli bir parametre de kullanıcı tarafından seçilen standart sapma değeridir. Standart sapma değeri küçük seçildiğinde, Gauss eğrisi daha dik bir yapı alır ve merkez pikselin katsayısı diğer piksellere göre daha belirgin hale gelir (Değirmenci vd., 2018).

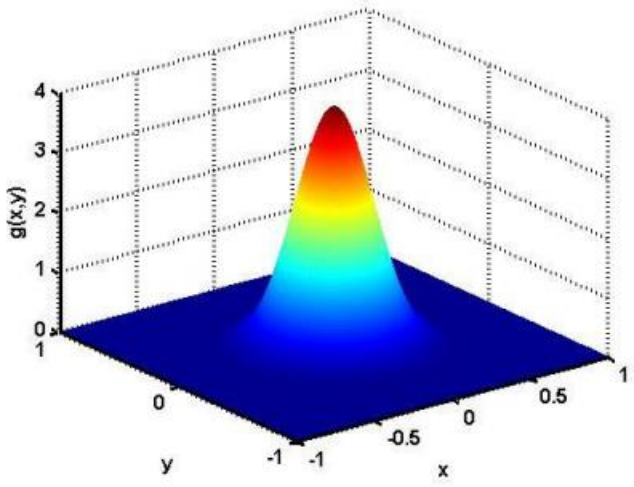
(a) (b)

(c) (d)

Şekil 20. σ =0.2 gürültü eklenmiş görüntü, (b) σ = 0.833 gürültü eklenmiş görüntü (c) σ = 1.666 gürültü eklenmiş görüntü (d) σ = 2.5 gürültü eklenmiş görüntü

Kaynak: Değirmenci vd., (2018, s. 201).

Bu durumda, komşu piksellerin filtrelenen piksel üzerindeki etkisi azalır ve görüntüdeki bulanıklık miktarı düşer. Ancak bu durum, gürültü giderme performansının da azalmasına yol açar. Standart sapma değeri büyük seçildiğinde ise Gauss eğrisi daha yatay bir dağılım gösterir ve filtre maskesindeki katsayılar birbirine daha yakın olur, bu da daha fazla bulanıklaşmaya neden olur (Değirmenci vd., 2018).



Şekil 21. 2 boyutlu Guassian dağılımı

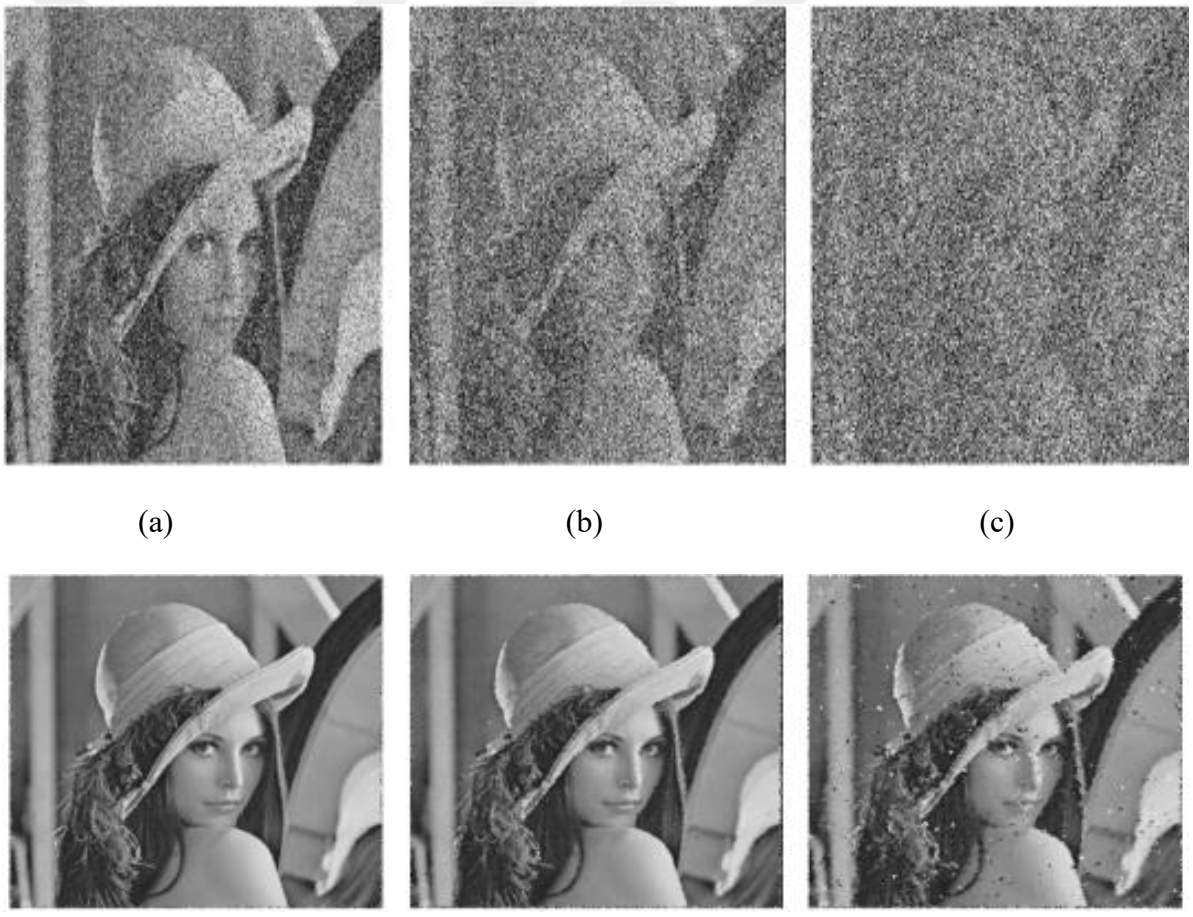
Kaynak: Çelik K, (2015, s. 22).

Şekil 21'deki x ve y değerleri bir pikselin konumunu göstermektedir. Alfa değeri Gauss fonksiyonunun standart sapmasıdır. Bu değerin büyüklük oranıyla, filtre etkisi

arasında doğru orantı mevcuttur. Standart sapma ne kadar büyükse filtre etkisi o kadar yayılmıştır.

3.5.2. Ortanca Filtresi (Medyan Filtresi)

Genel olarak görüntü temizlemede kullanılan bu filtre, tuz-biber gürültüsü bulunan görüntülerde iyi sonuçlar vermektedir. Filtreleme işlemlerinde birbirinden farklı olan büyüklük ve şekillerde şablonlar kullanılabilir. Genellikle 3x3 kare şablonunda kullanılan bu filtre de 5x5, 7x7 veya başka çeşitli geometrik şekillerde de kullanılması mümkündür. Matris sayısı artırıldığı zaman görüntüde bozulmalar meydana gelir ve detaylar ortadan kaybolmaya başlar (Çayır, 2010).



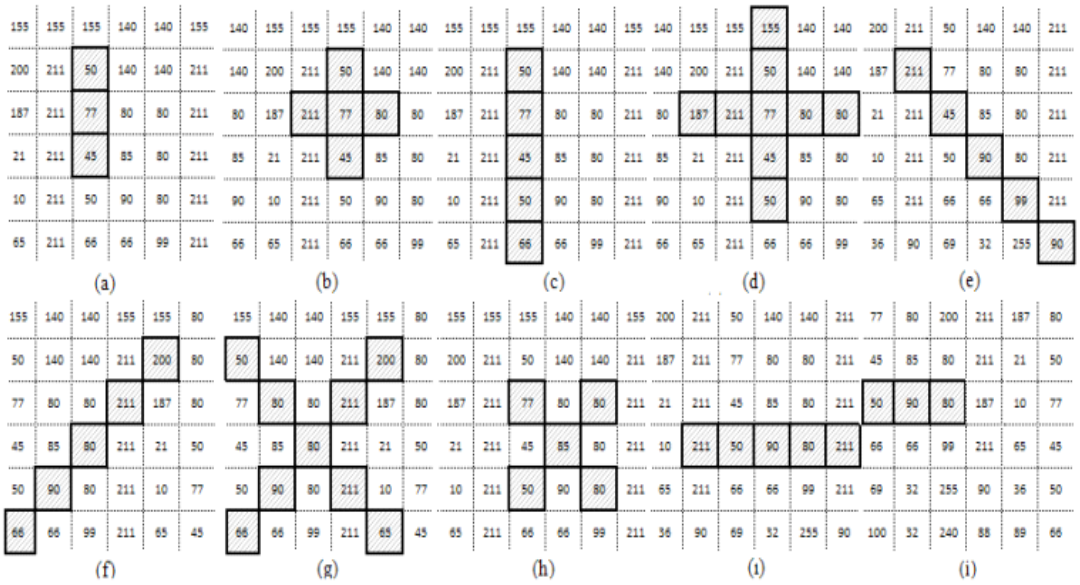
(a) (b) (c)

(d) (e) (f)

Şekil 22. (a) %30 Gürültülü (b) %60 Gürültülü (c) %80 Gürültülü (d) Medyan filtreli %30 temizlenmiş hali (e) Medyan filtreli %60 temizlenmiş hali (f) Medyan filtreli %80 temizlenmiş hali

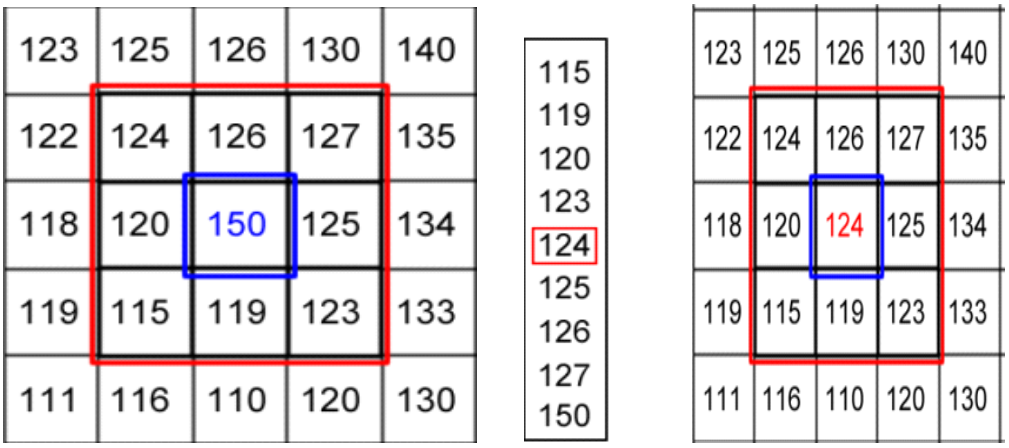
Kaynak: Erkan U ve Görkem L, (2017, s. 17).

Filtrede doğru sonuçları elde edebilmek için piksel oranıyla gürültü yoğunluğu arasında bir oran olması gerekmektedir. Piksel sayısının toplamının gürültü sayısına oranının %20'yi geçmemesi durumunda net sonuçlar elde edilir. Detayları fazla olan görüntülerde ise bazı ayrıntıların kaybolmasına yol açabilmektedir (Çayır, 2010).



Şekil 23. Farklı geometri şekillerinde medyan filtre kullanım örneği

Kaynak: Çayır T, (2010, s. 23).



Şekil 24. Medyan filtresi örneği

Kaynak: Çankaya G., Boyacı A. ve Yarkan S, (2021, s. 186).

Matematiksel olarak pikselleri X1, X2, X3, Xn olarak nitelendirildiği durumlarda medyan filtresi denklem 24'de gösterildiği gibi ifade edilebilmektedir;

$$Medyan = Ortanca\ ([X1, X2, \ldots\ldots, Xn]) \qquad (24)$$

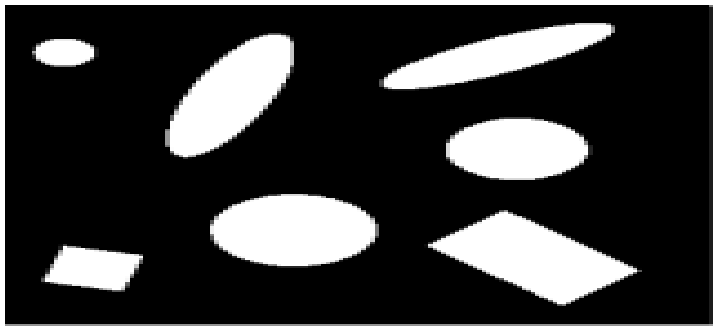
Burada bulunan sayılar eğer tek değer ise ortanca değerini bulabilmek için küçükten büyüğe sıralanır. Sıralanan bu değerlerin ortadaki sayısına denk gelen sayı medyan sayısı olarak kabul edilmektedir. Eğer elde edilen sayı çift ise, küçükten büyüğe doğru sayılar sıralanır. Orta kısma denk gelen 2 değerin aritmetik ortalamasının alınması gerekmektedir.

3.6. Blob Analiz

Blob analiz yöntemi, bilgisayar teknolojisinin ve görüntü analizinin gelişmesiyle birlikte görüntü işleme sektöründe önemli bir rol tutmuştur. Bu teknik, bir görüntüdeki nesnenin tespit edilebilmesi, takibinin yapılabilmesi ve analiz çalışmaları için kullanılmaya başlanmıştır. Blob analiz sayesinde bir nesnenin bir karedeki konumunu diğer karelerde de takip etmek mümkündür.

Şekil 25. Bir insan ve farklı şekiller içeren ikili görüntü

Kaynak: Moeslund T. B, (2012, s. 104).

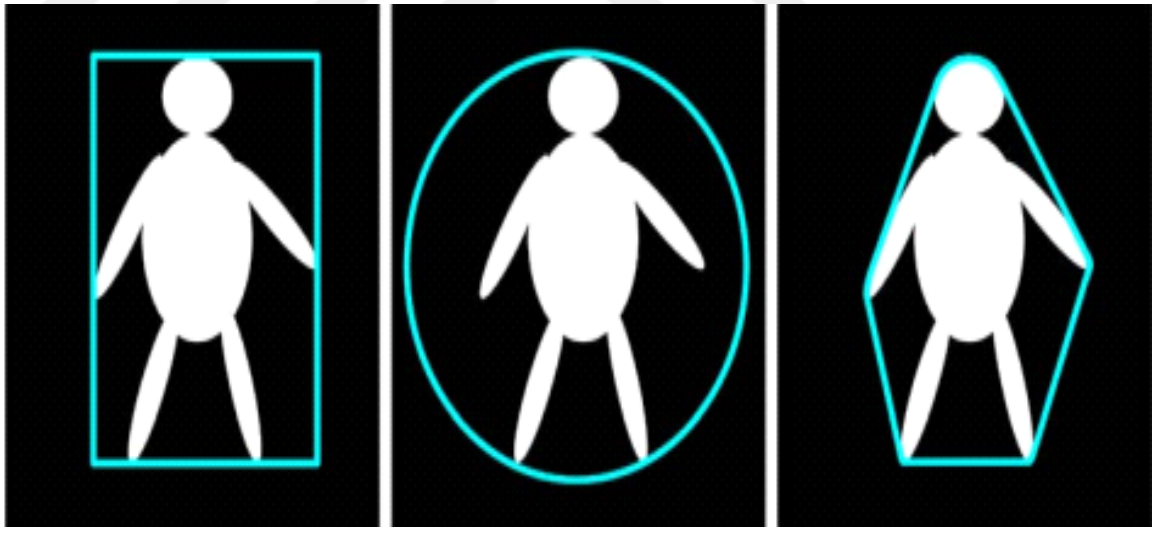


Şekil 26. Farklı şekiller içeren ikili görüntü

Kaynak: Moeslund T. B, (2012, s. 104).

Şekil 25‘de görülen yuvarlak şekli düşünebilen ve yorum yapabilen bir varlık olan insan için çok kolay bir davranıştır. Bilgisayar dilinde bu işlemin yapılabilmesi için şekil 26‘da gözüktüğü gibi farklı birer cisim olarak ayırmak, her biri için sınıf belirlemek ve bu sınıfları bilgisayarlara tanıtma işleminin yapılması gerekmektedir. Bu tekniğin başarılı olabilmesi için görüntüdeki karmaşıklığın en alt düzeyde olması zorunludur. Bu yüzden BLOB analiz öncesinde morfolojik işlemlerin yapılması gerekmektedir. Morfolojik işlemlerin ardından, ilk olarak görüntüdeki farklı nesnelerin (BLOB’ların) ayırt edilmesi gerekmektedir (Moeslund, 2012).

Bir görüntüdeki BLOB’lar, genellikle nesneleri veya ilginç bölgeleri temsil eden piksel gruplarından oluşmaktadır. Her bir BLOB bir nesne grubunu nitelemektedir. Bu nesnelerin belirleyici olan bazı özelliklerinden ayırt edebilmek için, BLOB’ların boyutlarına ve şekline bakılması gerekmektedir. Bir BLOB’un boyutlarını belirleyebilmek için piksel sayısı kullanılmaktadır. Bu şekilde çok büyük ve çok küçük BLOB’ları ayırma işlemi gerçekleştirilebilmektedir. Algılanan bu BLOB’ları göstermenin yolu, sınırlayıcı daire, kutu veya dışbükey kabuk çizmektir (Moeslund, 2012).



(a) (b) (c)

Şekil 27. (a) Sınırlayıcı dikdörtgen kutu (b) Sınırlayıcı daire (c) Konveks şekil

Kaynak: Moeslund T. B, (2012, s. 108).

Bir BLOB’un sınırlayıcı kutusu, çizilen en küçük dikdörtgen içerisinde tanımlanır. BLOB’un boyutunu belirleyerek çok büyük veya çok küçük BLOB’ları çıkarmak için kullanılır. Sınırlayıcı kutuyu kullanarak, BLOB içindeki piksellerin x ve

y koordinatları hesaplanır. Sınırlayıcı kutunun yüksekliği ve genişliği arasındaki oran, BLOB'un şeklini belirlemek için önemli bir özelliktir. Sınırlayıcı kutu aynı zamanda ROI (ilgilenilen bölge) olarak da adlandırılır. Bir BLOB'un sınırlayıcı kutu oranı, sınırlayıcı kutunun yüksekliğinin genişliğe bölünmesiyle tanımlanır (Moeslund, 2012).

$$Sıkılık = \frac{Blob\ bölgesi}{genişlik * yükseklik} \tag{25}$$

Fiziksel bir nesnenin ağırlık merkezi, nesnenin dengede kalması için parmağınızın yerleştirilmesi gerek noktadır. Bir nesnenin veya görüntünün kütle merkezi, tüm piksellerin x ve y koordinatlarının toplamının, toplam piksel sayısına bölünmesiyle bulunur. Bu nesnenin veya görüntünün ortalama konumunu ifade eder.

$$x_c = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{26}$$

$$y_c = \frac{1}{N}\sum_{i=1}^{N} y_i \tag{27}$$

Bazı durumlarda görüntüde eklenmiş görüntüler bulunabilir. Bu durumlarda medyan hesaplaması yapılması etkili bir çözüm olacaktır. Örneğin, bir kişinin gövdesinin merkezini bulurken, kolların bulunduğu konum sonucu etkileyebilir. Bu durumda sonucu daha az etkileyen ortalama konum yöntemi alternatif olabilir fakat ortalama konum yöntemi için daha fazla işlem yapılması gerekmektedir (Moeslund, 2012).

Matematiksel terimlerle bağlayıcı kutunun merkezi (x_{bb}, y_{bb}) olarak denklem 28 ve denklem 29 'da gösterildiği gibi hesaplanır.

$$x_{bb} = x_{min} + \frac{x_{max} - x_{min}}{2} = x_{min} + \frac{x_{max}}{2} - \frac{x_{min}}{2} = \frac{x_{min} + x_{max}}{2} \tag{28}$$

$$y_{bb} = y_{min} + \frac{y_{max} - y_{min}}{2} = y_{min} + \frac{y_{max}}{2} - \frac{y_{min}}{2} = \frac{y_{min} + y_{max}}{2} \tag{29}$$

Çizilecek bu kutu, en küçük dikdörtgenin merkezidir. Bu yöntem ile nesnenin konumunu hızlı bir şekilde tahmin etmek mümkündür. Hız kazanmak için yapılan bu yöntem aynı zamanda hesaplama için de kolay bir yöntemdir. Bulmak istediğimiz nesne her seferinde tam orta noktasını bulamayabilir ancak yaklaşım olarak işlev görür. Bu nedenle, hızlı ve ortalama bir tahmin gerektiren durumlarda kullanılır (Moeslund, 2012).

DÖRDÜNCÜ BÖLÜM
GÖRÜNTÜ İŞLEME SİSTEMİNİN YAZILIMI VE UYGULAMA SONUÇLARI

4.1. Genel Yapı

Ürünlerin doğru tespiti, makinelerdeki performansı artırmasının yanı sıra kaliteli bir ürün oluşturmanın ilk adımıdır. Üretim hattındaki her bileşenin, belirlenen üretim şartlarına ve kalite standartlarına uygunluğunu sağlamak, maliyetlerin azaltılması ve müşteri memnuniyetinin sürdürülebilirliği açısından son derece önemli rol oynamaktadır. Bu doğrultuda, somunlar gibi küçük ancak kritik parçaların tespiti, yalnızca montaj süreçlerinin doğruluğu açısından değil, aynı zamanda nihai ürünün güvenilirliği ve dayanıklılığı açısından da büyük bir rol oynamaktadır.

Görüntü işleme teknolojisi, bu tip süreçlerde büyük bir kolaylık sağlamaktadır. Bilgisayarla görme sistemleri, üretim hattındaki her parçayı gerçek zamanlı olarak analiz eder ve belirlenen parametrelere uygunluğunu değerlendirir. Örneğin, bir somunun doğru yerleştirilip yerleştirilmediği, boyutlarının ve şeklinin standartlara uygun olup olmadığı, bu teknolojiler aracılığıyla hızlı ve doğru bir şekilde belirlemek mümkündür. Bu tip işlemleri yaparken kullanılan algoritmalar, görüntüdeki piksel yoğunlukları, renk aralıkları gibi faktörleri analiz ederek, olası sapmaları algılayabilmekte ve anında geri bildirim sağlayabilmektedir.

Somunların doğru tespiti, üretim hattındaki genel kaliteyi korumak adına özellikle önemlidir. Bir somun yanlış yerleştirilmişse veya eksikse, bu durum sonraki montaj aşamalarında zincirleme hatalara neden olabilir. Örneğin, yanlış somun kullanımı, mekanik yapıların yeterince sağlam olmamasına, ürünün performansında düşüşe ve hatta son kullanıcı açısından güvenlik risklerine yol açabilmektedir. Bu nedenle, doğru tespit sistemleri, bu tip hataların önceden tespit edilmesine ve hızlı bir şekilde düzeltilmesine imkân tanımaktadır.

Otomatik tespit sistemlerinin en önemli avantajlarından biri, insan müdahalesini minimuma indirerek insan kaynaklı hataları ortadan kaldırabilmesidir. El ile yapılan kontrollerde, operatörler zamanla yorgunluk yaşayabilir ve bu da hata payının artmasına neden olabilmektedir. Ancak görüntü işleme algoritmaları, sürekli ve tutarlı bir doğrulukla çalışır, böylece kalite kontrol süreçlerinde hatasız bir değerlendirme

sunar. Üretim süreci boyunca yapılan bu tür tespitlerin hızla ve hassasiyetle gerçekleştirilmesi, üretim verimliliğini artırırken aynı zamanda maliyetleri de düşürür.

Bu çalışmada, görüntü işleme teknikleri kullanarak somunların ayrıştırılması ve yabancı maddelerin tespiti üzerine bir sistem geliştirilmiştir. Geliştirilen algoritmanın ana amacı, bir parça üzerinde bulunan somunları doğru şekilde tanımlamak, gerekli sayıda somun olup olmadığını kontrol etmek ve bu süreçte olası yabancı maddeleri tespit etmektir. Bu süreç, üretim hattında somunların doğru bir şekilde monte edilip edilmediğini ve parça üzerinde istenmeyen objelerin bulunup bulunmadığını hızlıca anlamayı sağlamaktadır.

Sistemin temel prensibi, bir kamera aracılığıyla elde edilen görüntünün işlenmesi ve somunların belirlenen bölgelerde doğru şekilde olup olmadığının tespit edilmesidir. Somunların tespit edilmesi, renk aralıkları (HSV değerleri) ve belirlenmiş geometrik koordinatlar yardımıyla gerçekleştirilmektedir. Bu koordinatlarla belirlenen alanlar, görüntü üzerindeki somunların bulunması gereken yerlerdir. Sistem, her bir somunun boyutlarını ve konumlarını kontrol eder ve tanımlanan küçük ve büyük somunlar için ayrı ayrı analiz yapar.

Somunları ayırma işlemi, üretim sürecinde kritik bir adımdır. Yanlış somun montajı, parçanın işlevselliğini olumsuz etkileyebilir, ayrıca yabancı maddelerin varlığı üretim kalitesini düşürebilir. Geliştirilen sistem hem üretim verimliliğini artırmak hem de hatalı üretimlerin önüne geçmek amacıyla bu denetimi otomatik hale getirmektedir. Böylece üretim hattında insan müdahalesine ihtiyaç duyulmadan somunların kontrolü sağlanabilir ve olası hatalar hızlıca tespit edilebilir.

4.2. Kamera Bilgileri

Bu projede görüntüleri almak için kullanılan kamera Hikvision markasının DS-2CD1053G0-IUF modelidir.

Şekil 28. Kamera resmi

Kameranın özelikleri

Görüntü Sensörü: 5 megapiksel, 1/2.7" Progressive Scan CMOS sensör.

Bu sensör, düşük ışık koşullarında bile net ve ayrıntılı görüntüler sağlar.

Çözünürlük: 2560 × 1920 piksel

Yüksek çözünürlük, geniş alanları daha fazla detayla izlenmesine imkan tanır.

Lens: 4mm sabit lens

4mm lens, geniş bir görüş alanı sağlar, özellikle iç mekanlarda geniş alanların izlenmesi için uygundur.

4.3. OpenCV

Çalışmada görüntü işleme ve nesne tanıma işlemlerini gerçekleştirmek için OpenCV (Open Source Computer Vision Library) kullanılmıştır. OpenCV, görüntü ve video işleme, nesne algılama, makine öğrenmesi gibi geniş bir yelpazede kullanıma sahip açık kaynaklı bir kütüphane olarak öne çıkmaktadır. Bu kütüphanenin tercih edilmesinin başlıca nedenleri, performansı, geniş fonksiyon yelpazesi ve esnekliği ile ilgilidir.

Öncelikle, OpenCV'nin sağladığı yüksek verimlilik ve hız, özellikle gerçek zamanlı uygulamalar için idealdir. C++ tabanlı bir kütüphane olmasına rağmen Python programlama diliyle uyumlu olması, hızlı ve optimize edilmiş bir performans sunması bu çalışmada tercih edilme sebebinin başında gelmektedir (Elkıran, 2020).

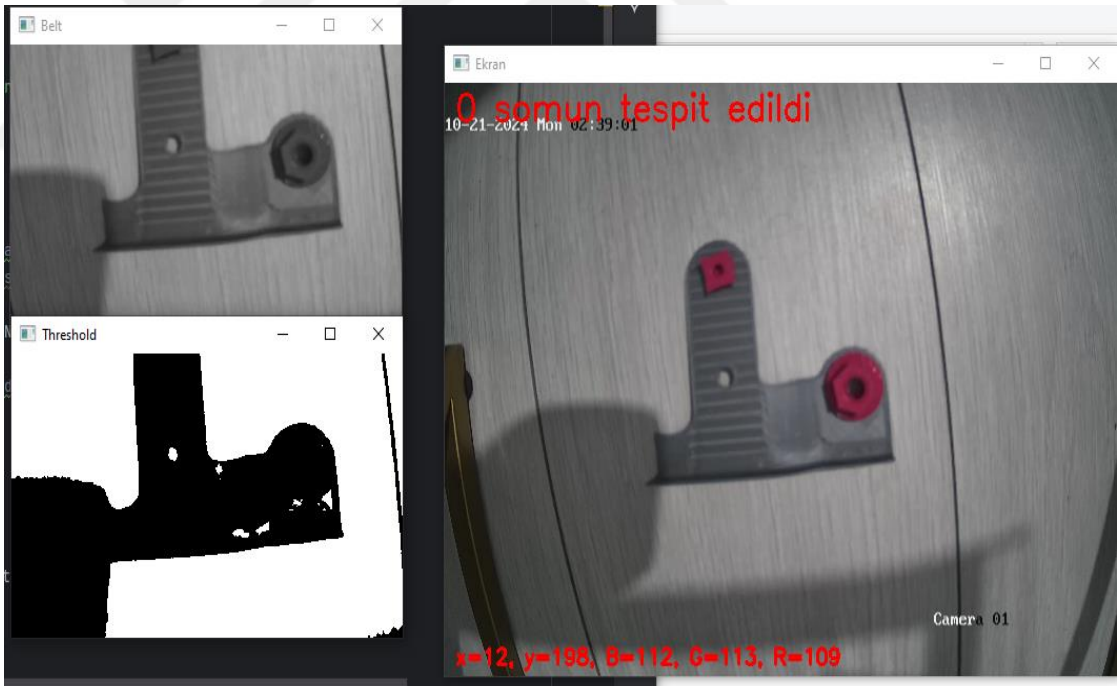
OpenCV'nin sunduğu geniş fonksiyon yelpazesi, görüntü işleme ve nesne algılama işlemlerinde oldukça esneklik sağlamaktadır. Kütüphane, temel görüntü işleme fonksiyonlarından gelişmiş nesne tanıma algoritmalarına kadar pek çok farklı

aracı içinde barındırır. Bu durum, proje aşamasında karşılaşılabilecek farklı sorunlara çözüm bulmaktadır.

Farklı işletim sistemlerinde sorunsuz çalışabilmesi ve geniş bir kullanıcı kitlesine sahip olması, karşılaşılan problemleri hızlıca çözme imkânı sunmaktadır. Aynı zamanda, sürekli güncellenen yapısıyla modern teknolojilere ve algoritmalara uyum sağlamaktadır.

4.4. Hsv Renk Uzayının Tercih Edilme Sebepleri

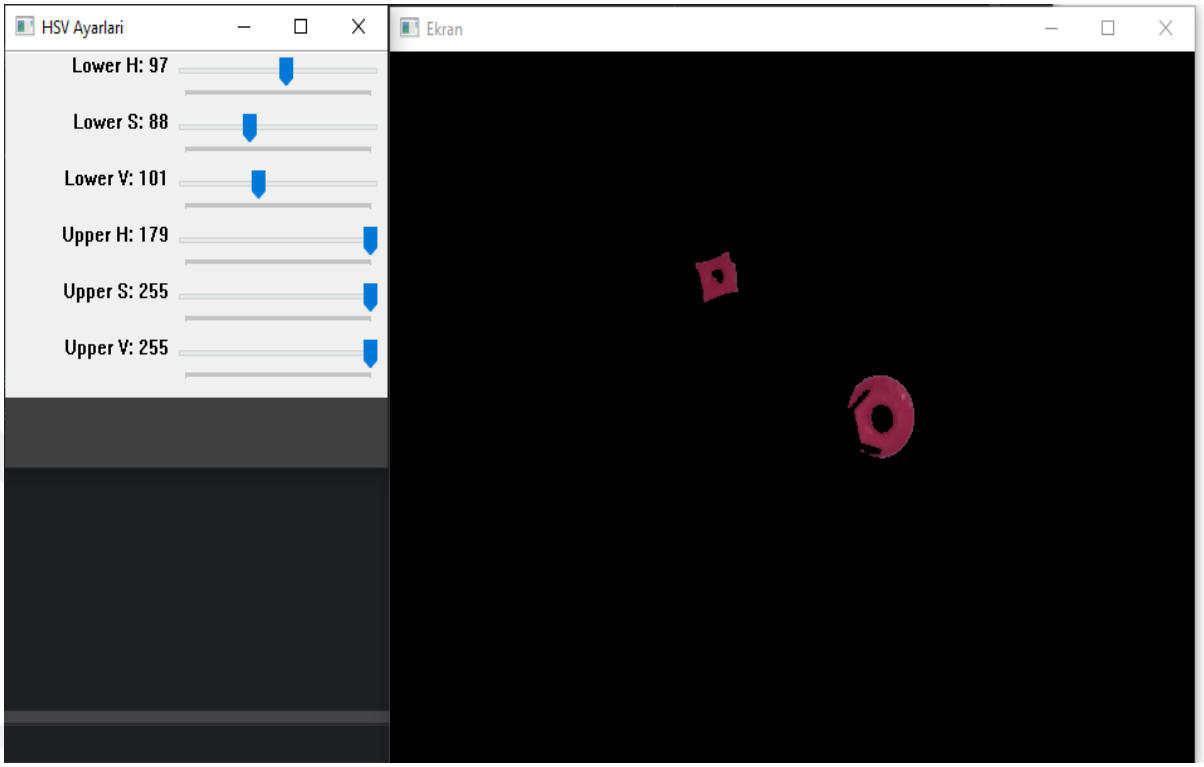
HSV modelinde renkler, ton (Hue), doygunluk (Saturation) ve parlaklık (Value) bileşenleri olarak tanımlanır. Bu tanımlamalar sayesinde, görüntülerdeki renkler daha doğal ve insan algısına daha uygun bir şekilde ortaya konmaktadır. Bu uyumlu çalışma şeklini RGB sisteminde yakalamak mümkün değildir. Çünkü RGB uzayında renkler üç kanalda (kırmızı, yeşil, mavi) olarak temsil edilmektedir ve bu kanallar birbirine bağımlıdır. Bu bağımlılık renk ayrımını zorlaştırmaktadır ve sonuçların doğruluğunu olumsuz etkileyebilmektedir.



Şekil 29. RGB Sisteminde oluşturulmuş bir çalışma

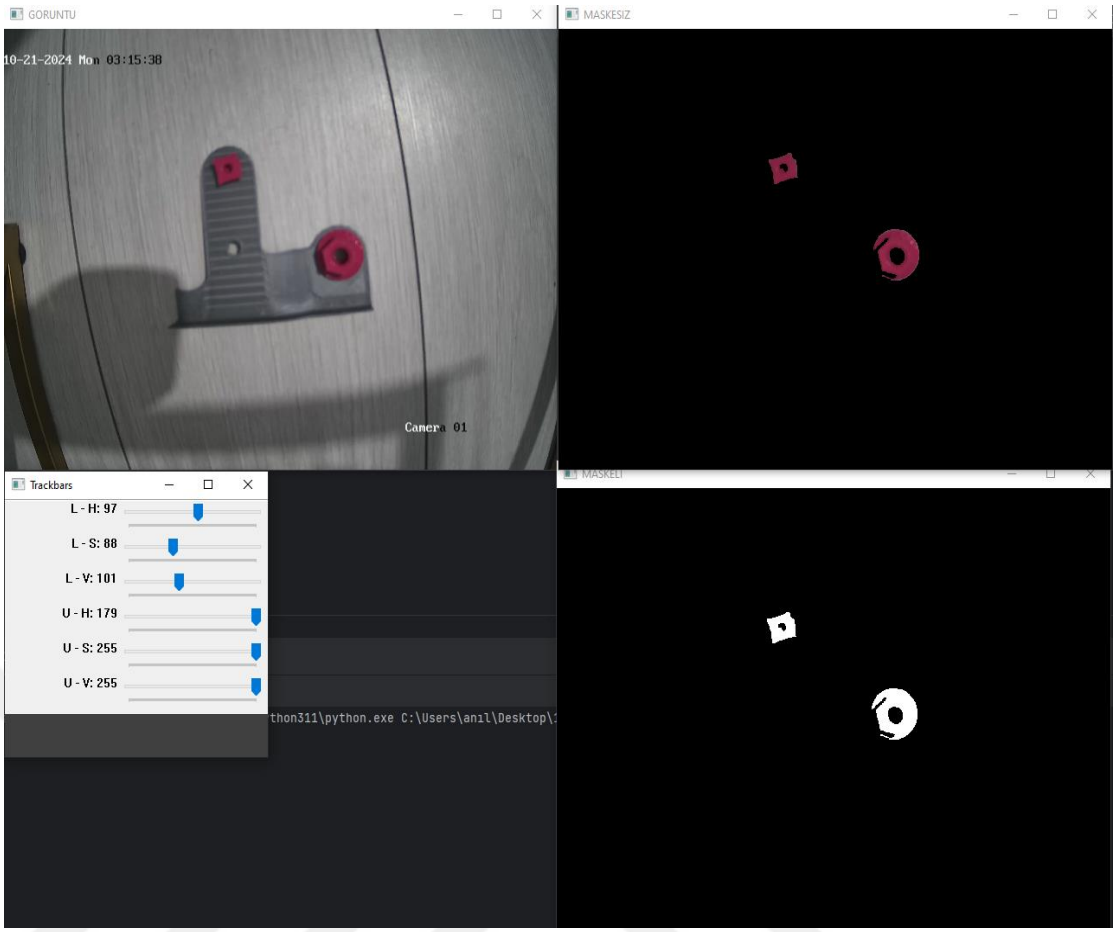
HSV renk uzayı, renk tespiti ve sınıflandırma işlemlerinde büyük avantajlar sağlamaktadır. HSV uzayında bir rengin tespiti genellikle ton bileşeni üzerinden yapılmaktadır. Bu özellik görüntü işleme yaparken büyük bir kolaylık sağlar. RGB modelinde kırmızı, yeşil ve mavi bileşenler arasında belirgin olmayan farklar, renk

ayrımını zorlaştırırken, HSV'de ton bileşeni kullanılarak bir rengin doğrudan tespit edilmesi mümkündür. Örneğin, nesnelerin renklerine göre segmentasyonu RGB uzayında oldukça karmaşık olabilirken, HSV uzayında sadece ton bileşenine bakılarak yapılabilir.



Şekil 30. Somunların maskesiz görüntüsü

HSV modelinin bir diğer önemli avantajı, farklı aydınlatma koşullarına karşı daha dayanıklı olmasıdır. RGB uzayında, aydınlatmadaki değişiklikler renk bileşenlerini ciddi şekilde etkileyebilir ve bu durum renk tespitinde hatalara yol açabilir. Ancak HSV modelinde, parlaklık (value) bileşeni ayrı bir kanal olarak bulunduğundan, ton ve doygunluk bileşenleri sabit kalabilir ve aydınlatmadaki değişikliklere rağmen renk tespiti daha doğru yapılabilir. Bu özellikle endüstriyel görüntü işleme ve otomatik kalite kontrol sistemlerinde önemli bir avantaj sağlar. Örneğin, bir üretim hattında, değişen ışık koşullarına rağmen nesnelerin renklerine göre sınıflandırılması gerektiğinde, HSV uzayının sağladığı bu esneklik, işlem doğruluğunu artırır.



Şekil 31. Maskesiz ve maskeli görüntü

Görüntünün parlaklığını değiştirmek veya bir rengi daha canlı hale getirmek gibi işlemler, HSV modelinde çok daha kolay bir şekilde yapılabilmektedir. RGB'de ise bu tür düzenlemeler karmaşık olabilir ve genellikle istenmeyen sonuçlar doğurabilir.

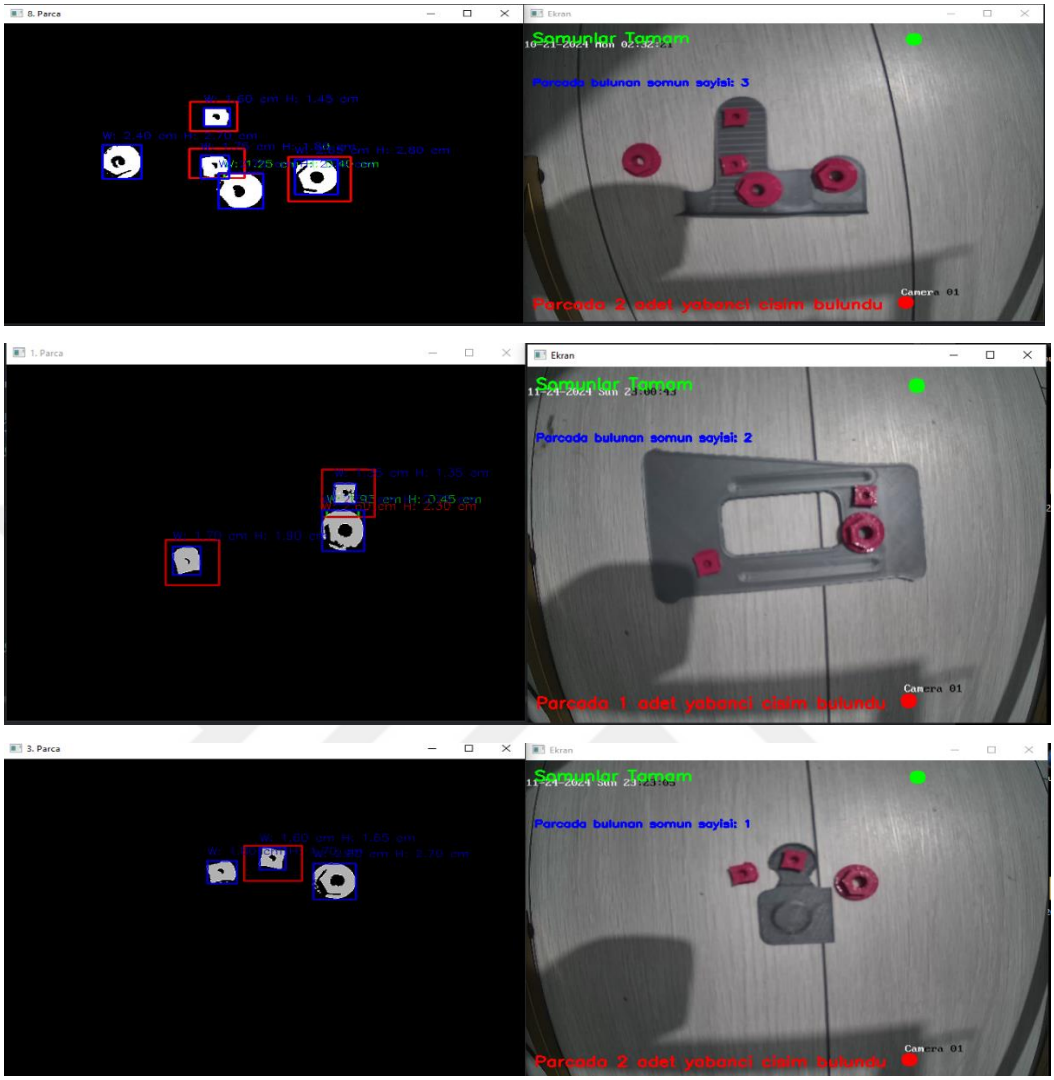
Sonuç olarak, HSV renk uzayı, hem renk tespiti ve sınıflandırma işlemlerinde hem de renk manipülasyonu ve düzenleme işlemlerinde büyük avantajlar sunmaktadır.

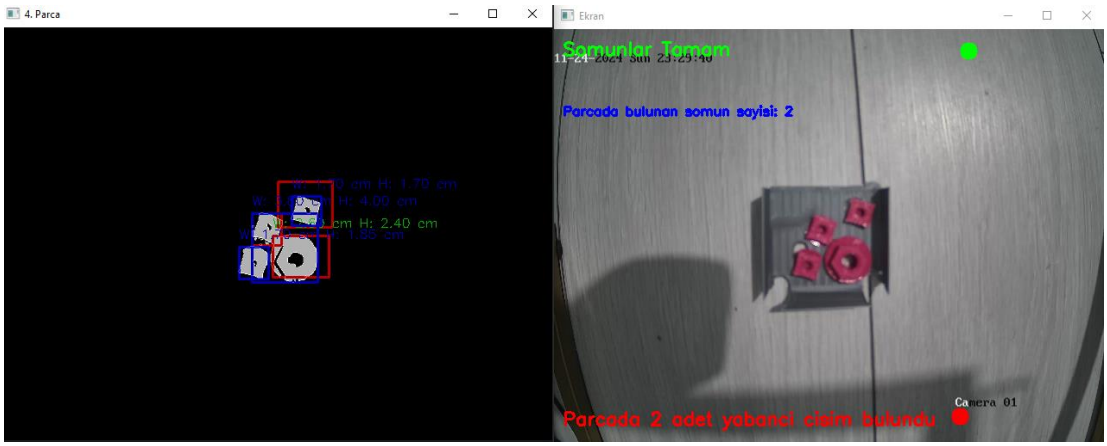
4.5. Yanlış Bölgedeki Somunların Tespiti

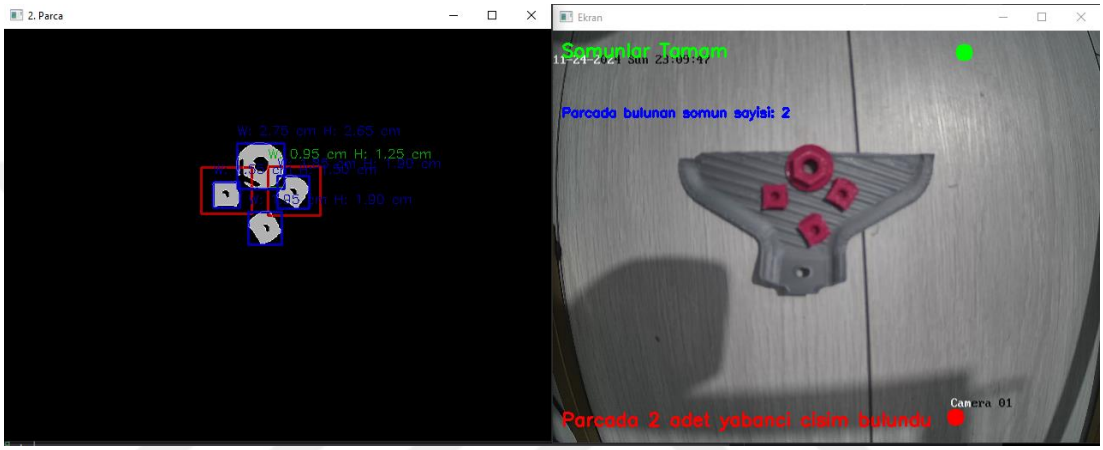
Otomotiv parçalarında bütün parçaların milimetrik hassasiyetle doğru yerlerde olması kritik bir öneme sahiptir. Yanlış bölgede bulunan parçalardan doğabilecek hataları önleyerek kalite standartlarını sağlamak ve hatalı ürünlerin son kullanıcıya ulaşmasını engellemek için önceden tespit edilmelidir. Bu tez çalışmasında üzerinde durulan konulardan bir tanesi de yanlış bölgedeki somunların tespit edilmesidir.

Somunların yerleşim yerlerinin doğruluğunu kontrol etmek için, belirli bir bölge içerisindeki somunların sayısı ve boyutları kod içerisinde belirlenen koordinatlar kullanılarak analiz edilmiştir. Kodun genel akışında, her bir bölge için HSV renk uzayı

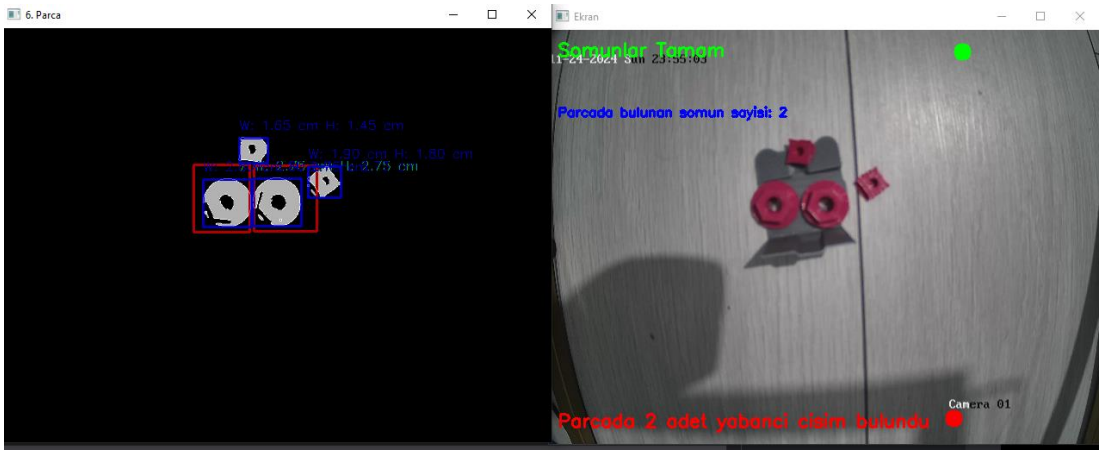
kullanılarak somunların maskelemesi yapılmakta ve bu maskeleme işlemi sonrasında somunların konturları tespit edilerek konumları kontrol edilmektedir.

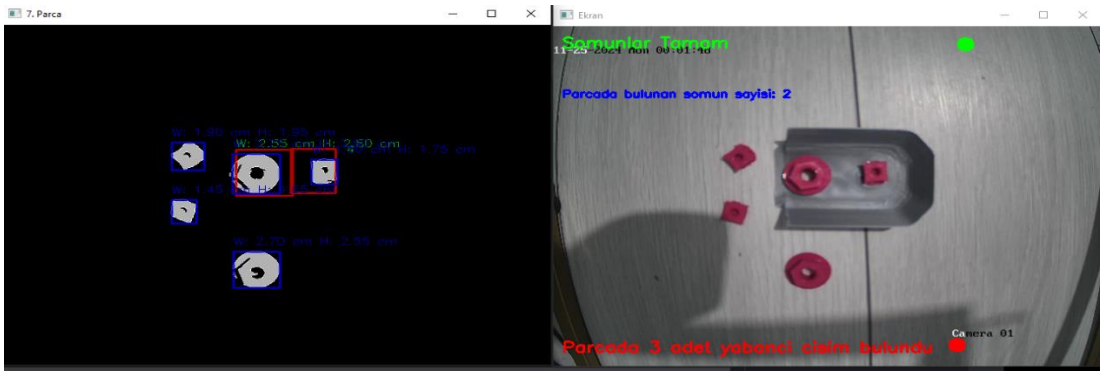



4. Parca
Ekran
Somunlar Tamam
Parcada bulunan somun sayisi: 2
W: 2.40 cm H: 2.40 cm
Camera 01
Parcada 2 adet yabanci cisim bulundu


2. Parca
Ekran
Somunlar Tamam
Parcada bulunan somun sayisi: 2
W: 0.95 cm H: 1.25 cm
Camera 01
Parcada 2 adet yabanci cisim bulundu


5. Parca
Ekran
Somunlar Tamam
Parcada bulunan somun sayisi: 2
Camera 01
Parcada 2 adet yabanci cisim bulundu


6. Parca
Ekran
Somunlar Tamam
Parcada bulunan somun sayisi: 2
Camera 01
Parcada 2 adet yabanci cisim bulundu



Şekil 32. Yabancı cisim bulunan parçanın görüntüsü

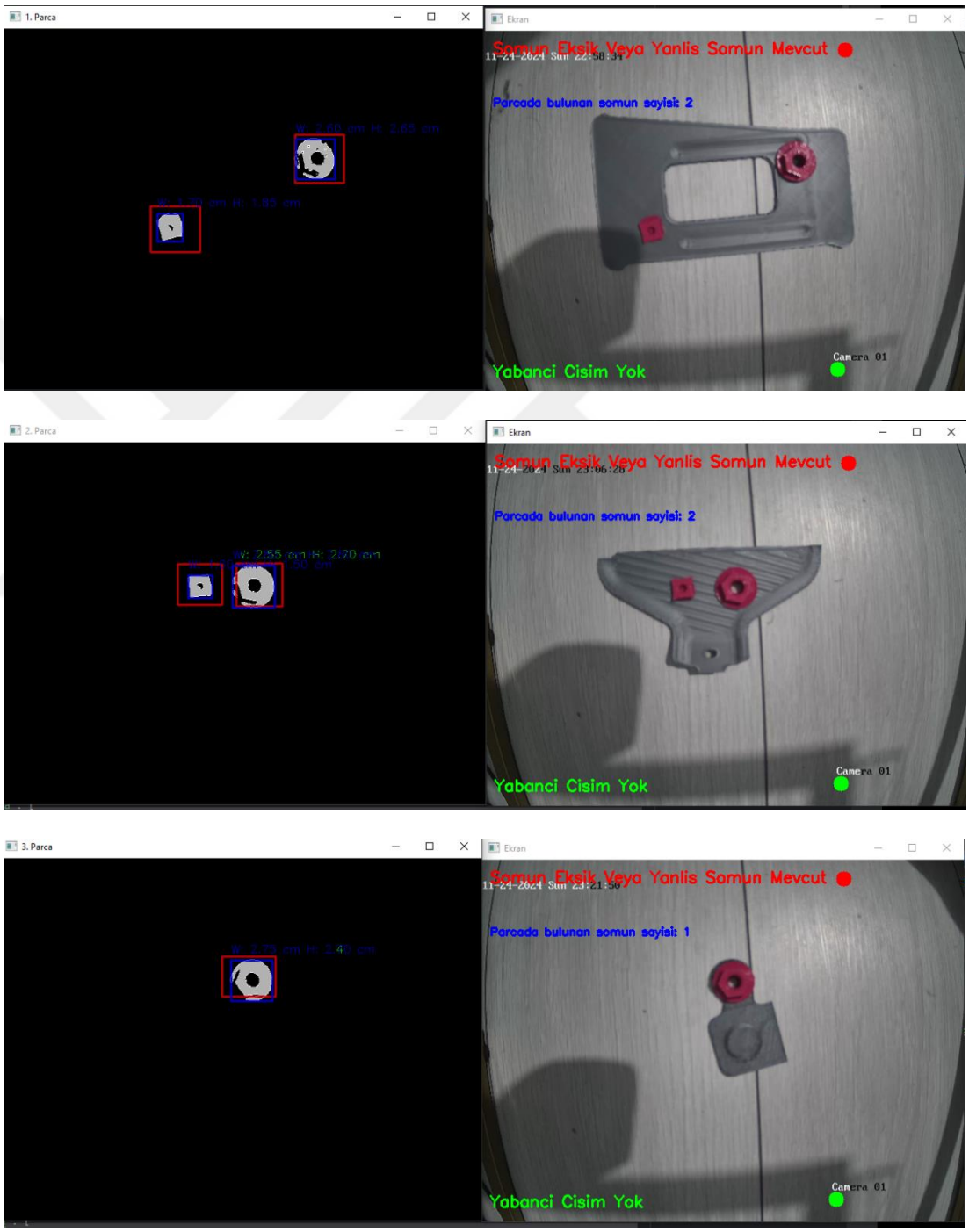
Kodda yer alan check_for_foreign_objects() fonksiyonu, her bir bölgeye özel olarak tanımlanan koordinatların dışında kalan ve tanımlanan boyut aralıklarına uymayan somunları belirlemek amacıyla geliştirilmiştir. Bu sayede, her bir parça için gereksinim duyulan somun sayısına ve türüne uyulup uyulmadığı kontrol edilebilmektedir. Tanımlı bölgeler dışında kalan somunlar yanlış bölgeye yerleştirilmiş olarak kabul edilmekte ve kullanıcıya bu hatalar hakkında bilgi verilmektedir.

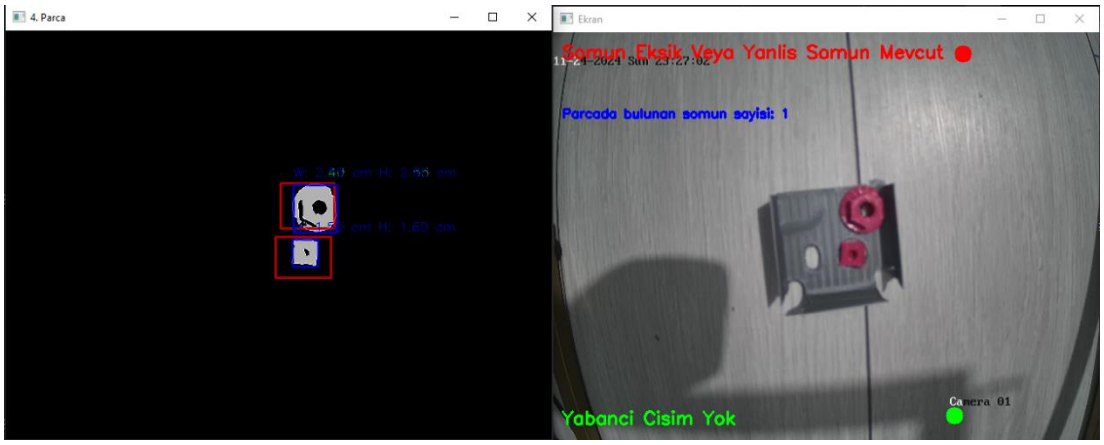
4.6. Farklı Boydaki Ürünlerin Tespiti

Kalitesel hataları dikkate alındığında oluşabilecek problemlerden bir tanesi de farklı boydaki somunların parçada olabilme ihtimalidir. Bu yüzden görüntü işleme tekniklerin boyut bazlı kalite kontrolde yapılabileceğini gösterebilmek için, farklı boyutlardaki ürünlerin otomatik olarak tespit edebileceği şekilde tasarlanmıştır. Kodun işleyişinde, öncelikle analiz edilecek ürünlerin yer aldığı görüntü üzerinde belirli bir renk uzayı (HSV) aralığına göre maskeleme işlemi uygulanır. HSV aralıkları, ürünlerin renk ve ton bilgilerini içerdiğinden, bu maskeleme işlemi ürünleri arka plandan ayırarak yalnızca ilgili nesnelerin öne çıkarılmasını sağlar. Bu şekilde, görüntüdeki istenmeyen gürültüler ve nesneler filtrelenir, yalnızca ürünler odak noktası haline getirilir.

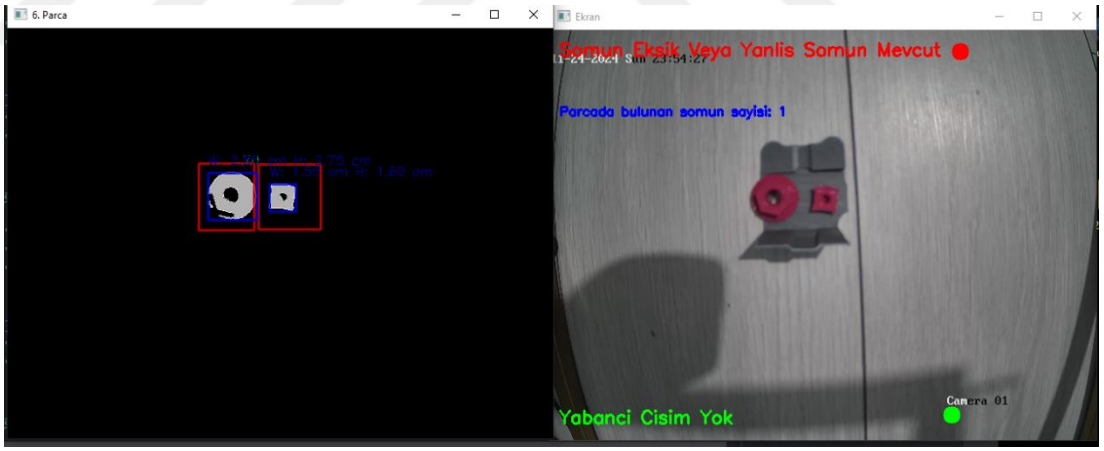
Maskeleme işleminin ardından, cv2.findContours() fonksiyonu ile maskelenmiş görüntü üzerinde kontur tespiti gerçekleştirilir. Kontur tespiti, her bir ürünün sınırlarını ve şekillerini tanımlamak amacıyla yapılır. Elde edilen kontur bilgileri sayesinde ürünlerin koordinatları (x, y konumları) ve boyut bilgileri (genişlik ve yükseklik) hesaplanır. Bu boyut bilgileri daha sonra, kod içerisinde önceden tanımlanan small_nut_range ve large_nut_range gibi boyut aralıkları ile karşılaştırılır. Tanımlanan

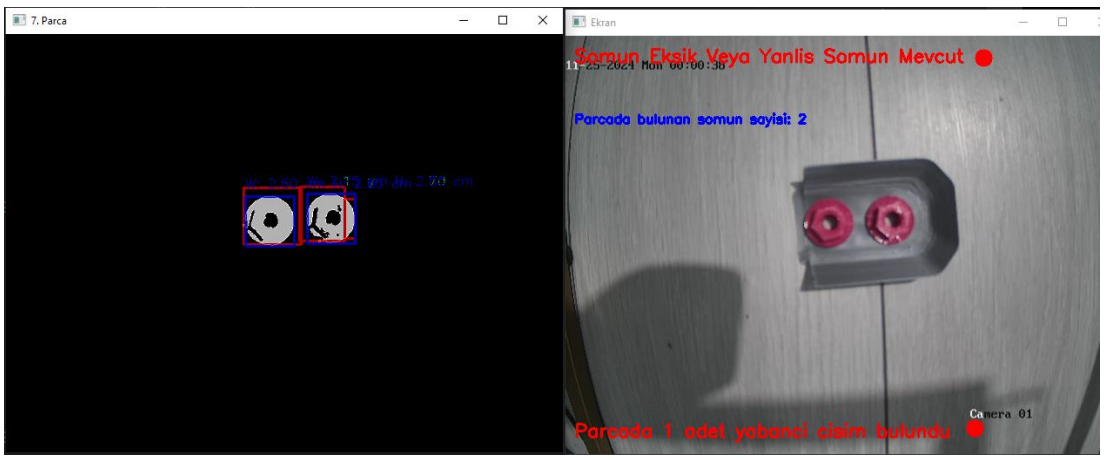
bu boyut aralıkları, küçük ve büyük ürünlerin hangi fiziksel boyutlara sahip olması gerektiğini belirtir. Örneğin; küçük bir ürünün boyut aralığı 1 < 2 cm arasında olabilirken, büyük bir ürünün boyut aralığı 2 < 3 cm arasında olacak şekilde tanımlanabilir.

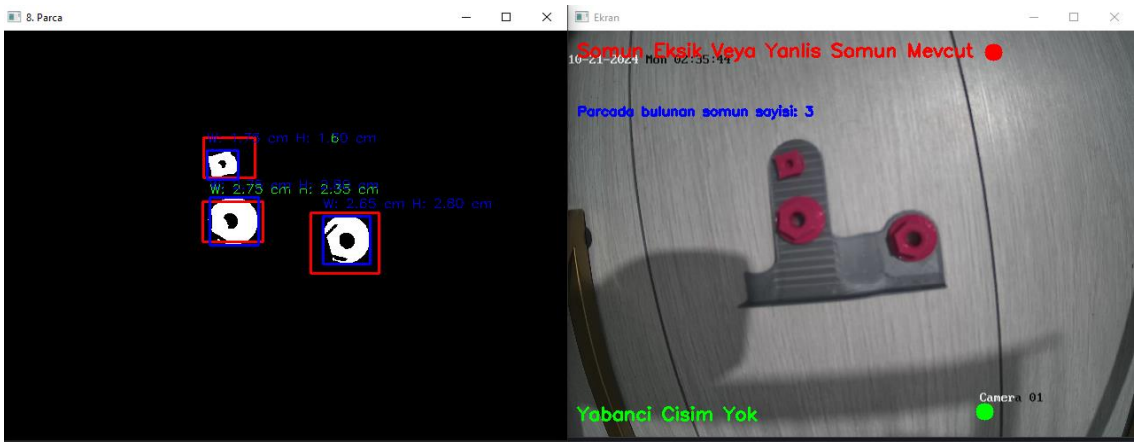



4. Parca
Ekran
Somun Eksik Veya Yanlis Somun Mevcut
Parcada bulunan somun sayisi: 1
Yabanci Cisim Yok
Camera 01


5. Parca
Ekran
Somun Eksik Veya Yanlis Somun Mevcut
Parcada bulunan somun sayisi: 2
Yabanci Cisim Yok
Camera 01


6. Parca
Ekran
Somun Eksik Veya Yanlis Somun Mevcut
Parcada bulunan somun sayisi: 1
Yabanci Cisim Yok
Camera 01


7. Parca
Ekran
Somun Eksik Veya Yanlis Somun Mevcut
Parcada bulunan somun sayisi: 2
Parcada 1 adet yabanci cisim bulundu
Camera 01



Şekil 33. Yanlış somun bulunan parçanın görüntüsü

Boyut tespitinden sonra, ürünlerin bu tanımlı aralıklarla uyumlu olup olmadığı değerlendirilir. Eğer bir ürün tanımlanan boyutların dışında kalıyorsa veya belirlenen boyut sınırlarının çok üzerinde ya da altında bir değere sahipse, bu ürün farklı boyutta bir ürün olarak kabul edilir. Ayrıca, kod içerisinde check_for_foreign_objects() fonksiyonu ile ürünlerin yalnızca tanımlanan bölgelerde olup olmadığı kontrol edilir. Bu işlev, bir ürünün önceden belirlenen alanın dışında yer alıp almadığını anlamak ve hatalı bir yerleşim olup olmadığını belirlemek amacıyla kullanılır.

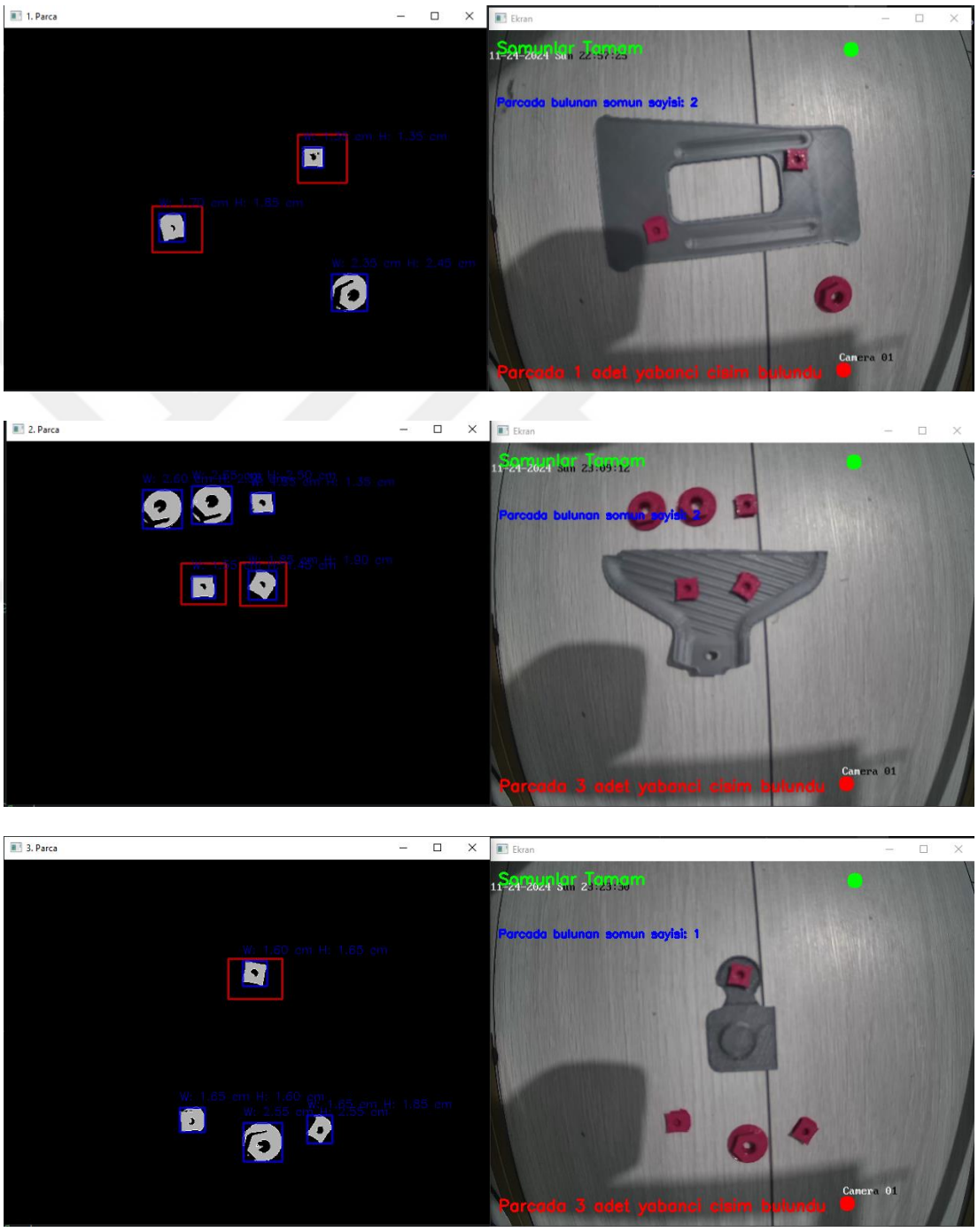
Tespit edilen hatalar ve ürünlerin boyut bilgileri, ekranda görsel olarak işaretlenir ve kullanıcıya geri bildirim olarak sunulur. Yanlış boyutta veya yanlış bölgede bulunan ürünler farklı renklerle işaretlenir; bu sayede kullanıcı hatalı ürünleri kolayca ayırt edebilir. Örneğin, istenilen boyut sınırlarının dışında kalan her ürün kırmızı renk ile işaretlenmektedir. Kontrol edilen ürünün boyut bilgisi ekranda gösterilir.

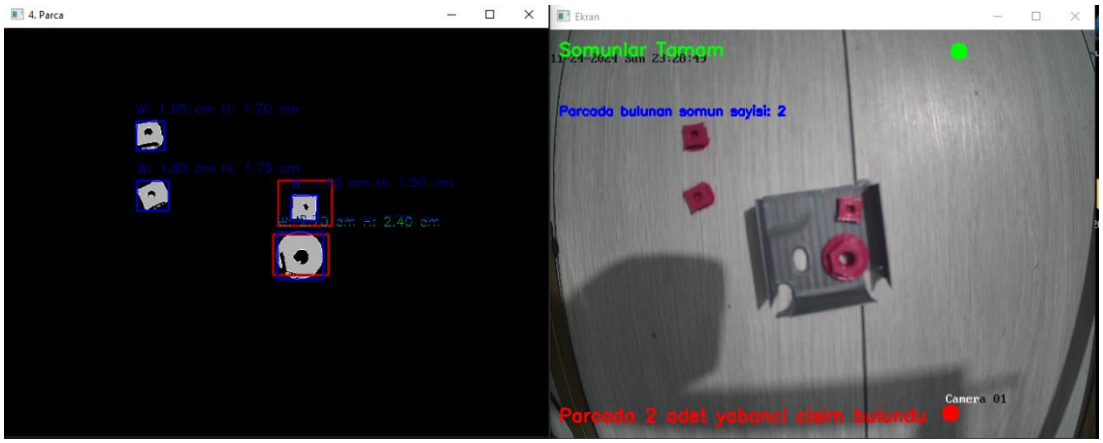
4.7. Yabancı Madde Tespiti

Algoritmanın bir diğer önemli fonksiyonu, tanımlanmış bölgelerin dışında yer alan yabancı maddeleri algılamaktır. Bu yabancı maddelerin boyutları hesaplanarak, görüntü üzerinde işaretlenir ve potansiyel bir sorun olup olmadığı hakkında bilgi vermektedir. Yabancı madde kontrolü, özellikle hatasız bir üretim ortamının korunması açısından büyük önem taşımaktadır. Yabancı maddelerin erkenden tespit edilmesi, üretim hattındaki potansiyel sorunları önceden saptanması zaman ve maliyet kayıplarının önüne geçebilir.

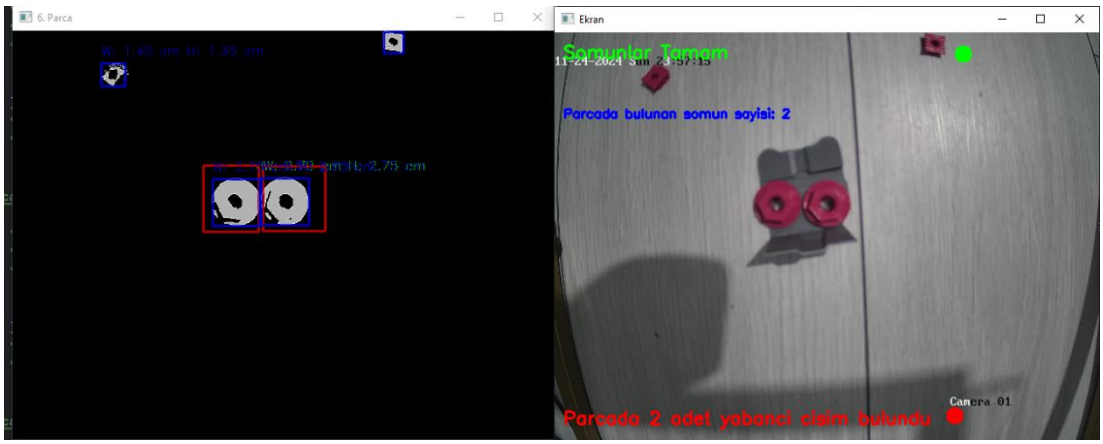
Bu otomatik kontrol mekanizmasının en önemli avantajlarından biri, sürekli izleme yapabilmesidir. Görüntü işleme teknoloji sayesinde oluşturulan bu sistem,

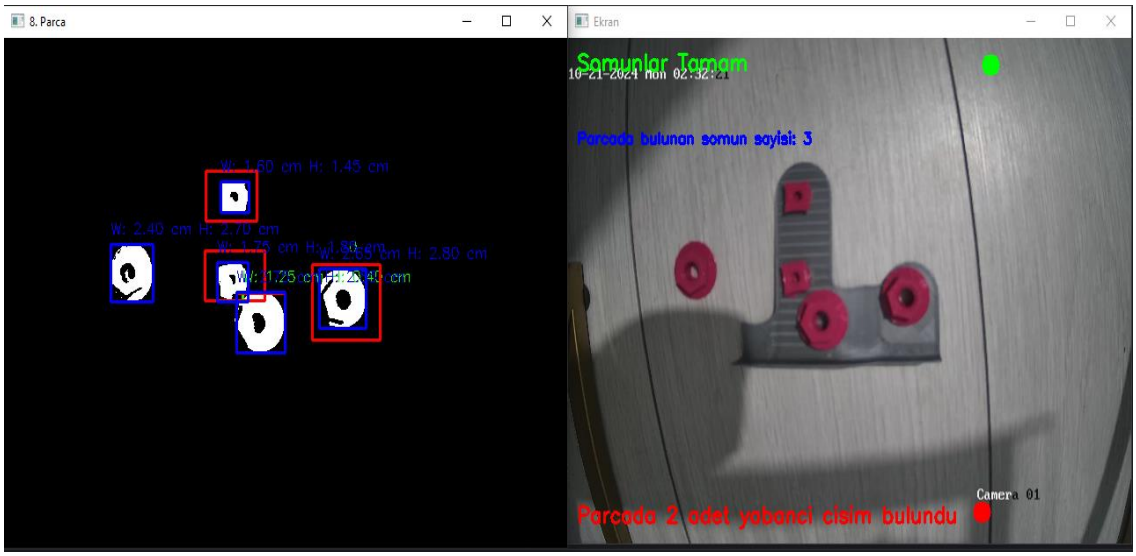
insan hatasından kaynaklanabilecek eksiklikleri ortadan kaldırır ve hızla sonuç elde eder. Ayrıca, sistemin her bir somunu ayrı ayrı analiz etmesi, yanlış somunların tespit edilmesini kolaylaştırır ve üretim sırasında yapılabilecek montaj hatalarını önceden saptayabilmektedir.




4. Parca
Ekran
Somunlar Tamam
Parcada bulunan somun sayisi: 2
Parcada 2 adet yabanci cisim bulundu
Camera 01


5. Parca
Ekran
Somunlar Tamam
Parcada bulunan somun sayisi: 2
Parcada 2 adet yabanci cisim bulundu
Camera 01


6. Parca
Ekran
Somunlar Tamam
Parcada bulunan somun sayisi: 2
Parcada 2 adet yabanci cisim bulundu
Camera 01



Şekil 34. Yabancı cisim bulunan parçanın görüntüsü

Bu sistemin dezavantajları arasında, görüntü kalitesine bağımlılık yer almaktadır. Kötü aydınlatma veya kamera açısındaki sorunlar, somunların doğru şekilde tespit edilememesine neden olabilir. Ayrıca, somunların renk ve boyut varyasyonlarının çok fazla olduğu durumlarda, sistemin doğruluğu azalabilir. Ancak, bu tür sorunlar doğru kamera ve aydınlatma koşulları ile minimuma indirilebilir.

4.8. Canny Algoritması Kullanılma Sebepleri

Canny kenar algılama algoritması, metal parçalar üzerindeki somunların kalite kontrolünde tercih edilen en etkin yöntemlerden biridir. Bu tezde Canny algoritmasının kullanılmasının sebeplerinden bir tanesi Canny algoritmasının görüntü işleme alanında kenar tespitinde en yaygın ve güvenilir yöntemlerden biri olduğunun bilinmesidir. Algoritmanın temel amacı, bir nesnenin sınırlarını belirginleştirerek, bu sınırların net bir şekilde tanımlanmasını sağlamaktır. Özellikle yüksek hassasiyet gerektiren endüstriyel uygulamalarda, otomotiv sektöründe kullanılan somun ve bağlantı elemanlarının denetiminde, hatasız tespit yapılması son derece önemlidir. Çünkü metal yüzeylerde meydana gelebilecek herhangi bir kusur, parça montajı veya genel işleyişi olumsuz etkileyebilmektedir.

Bu tez kapsamında, farklı parçalar üzerinde yer alan somunların doğru konumlandırılıp konumlandırılmadığının yanı sıra, yabancı cisimlerin varlığı da tespit edilmektedir. Bu durum, görüntüde yalnızca belirli alanların dikkate alınmasını ve ilgili alanların dışında kalan bölgelerdeki olası yabancı cisimlerin de hassas bir şekilde belirlenmesini gerektirir. Canny algoritması burada devreye girerek, ilgili alanlarda

yer alan somunların net bir biçimde sınırlarını çizmeyi ve yabancı cisimleri ayırt etmeyi mümkün kılar. Bu işlem, görüntünün önce bir maske ile işlenmesi ve ardından belirlenen HSV renk aralıklarına göre bir alanın tanımlanması şeklinde gerçekleştirilir. Ancak somunların veya yabancı cisimlerin kenarlarının belirginleşmesi, yalnızca renk ayrımıyla değil, aynı zamanda kenarların belirgin hale getirilmesi ile sağlanır. Canny algoritması da tam olarak bu noktada devreye girer; maskelenmiş alanlarda kenarları tespit ederek nesnelerin geometrik yapısını net bir şekilde ortaya koyar.

Canny algoritmasının metal yüzeylerdeki somunlar gibi küçük nesnelerin sınırlarını daha keskin bir şekilde belirleyebilmesi, görüntüdeki her bir detayın ayrıntılı olarak değerlendirilmesine olanak tanır. Algoritma, görüntüdeki gürültüyü azaltarak yalnızca ilgili kenarların tespit edilmesini sağlar. Bu, endüstriyel görüntü işlemelerinde yaygın bir sorun olan "yanlış pozitif" tespitlerin azaltılması anlamına gelir. Kenar belirleme aşamasında gerçekleştirilen Gauss filtresi uygulaması, gürültüyü baskılayarak daha temiz bir görüntü sunar ve ardından ikili eşikleme yöntemiyle görüntüdeki belirgin kenarların kalıcı olarak korunmasını sağlar. Bu sayede, parçaların üzerinde gereksiz detayların işleme dahil edilmesi engellenmiş olur ve yalnızca somun gibi önemli nesneler vurgulanır.

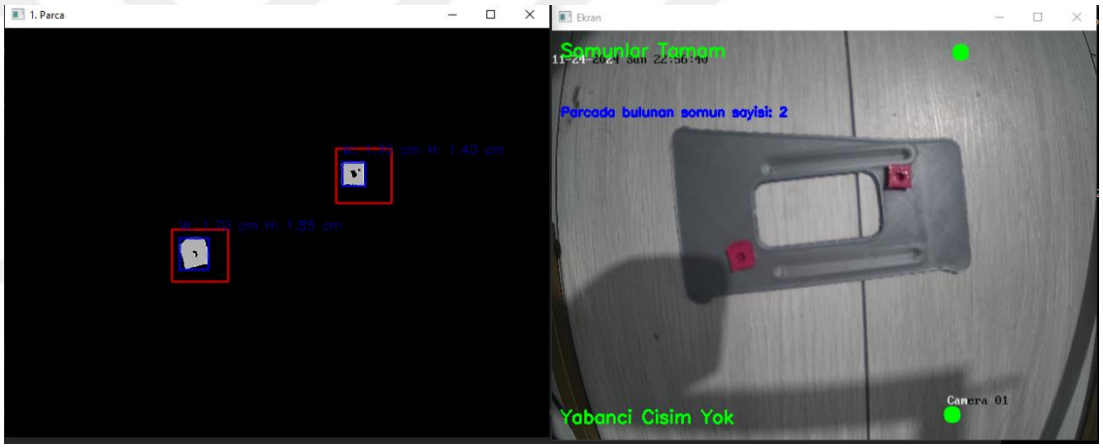
Tezde ayrıca, parça üzerinde bulunması gereken somunların sayısı ve türü de dikkate alınarak kontrol sağlanmaktadır. Somunların sayısı ve boyutları, Canny algoritmasının ortaya çıkardığı kenar hatları sayesinde hesaplanabilmektedir. Bu kenarlar sayesinde somunların alanları belirlenir ve her somunun boyutu doğrulanarak, ilgili parçada olması gereken somun sayısı ile kıyaslanır. Eğer somun sayısında veya boyutunda bir eksiklik ya da fazlalık varsa, bu durum tespit edilerek bir uyarı mesajı ile belirtilir. Bu tür bir kontrol mekanizması, otomotiv sektöründe kalite güvence süreçlerinin temel taşlarından biridir; çünkü her bir somunun doğru konumda ve boyutta olması, parçanın güvenliğini doğrudan etkiler.

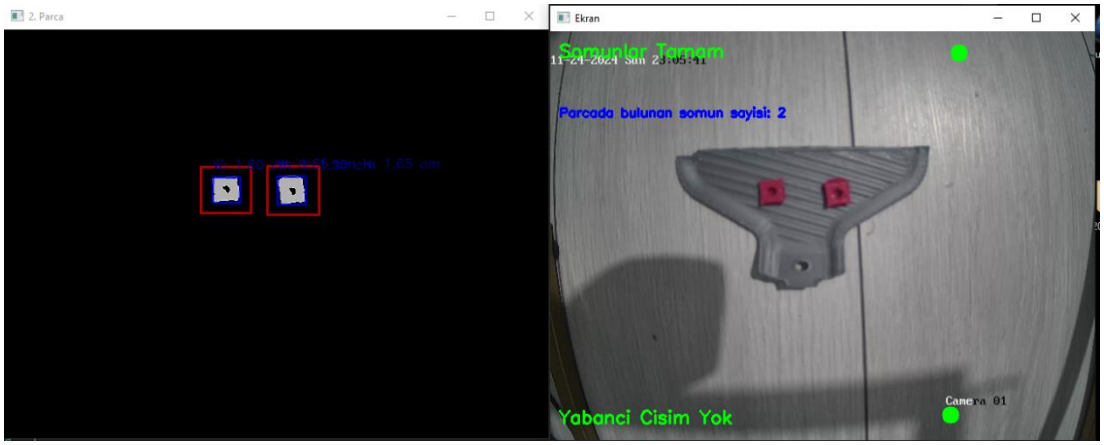
Canny algoritmasının bu tezde öne çıkmasının diğer bir nedeni de maskelenmiş alanların dışında yer alan yabancı cisimleri de tespit edebilmesidir. Görüntü üzerinde istenmeyen herhangi bir cisim bu sistemle kolayca fark edilebilir. Canny algoritması sayesinde, yabancı cisimlerin kenarları belirginleştirilir ve bu kenar bilgileri ile cismin büyüklüğü ve konumu saptanır.

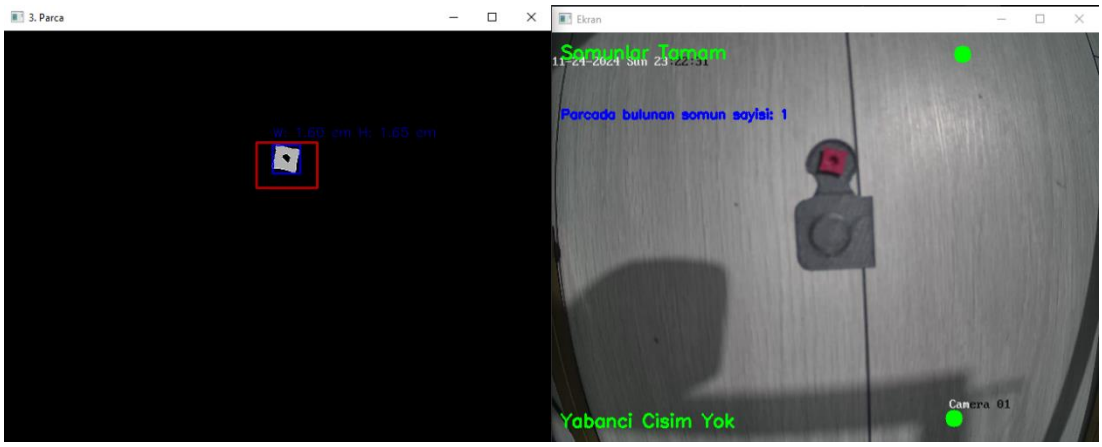
Tezde mevcut sistemin işleyişine baktığımızda, tanımlanan alanların dışında kalan bölgelerde yabancı cisimlerin tespiti için de özel bir kontrol mekanizması bulunmaktadır. Canny algoritması bu mekanizma ile uyumlu çalışarak, her bir parçanın kenar tespitini başarıyla tamamlar ve parçanın dışında yer alan yabancı cisimlerin belirlenmesini sağlar. Kenarları hassas bir şekilde belirleyebilme özelliği, yüzeylerdeki ince detayların dahi gözden kaçmamasını sağlar. Böylece üretim aşamasında meydana gelebilecek potansiyel hatalar önceden tespit edilerek kalite güvence sürecine katkıda bulunur.

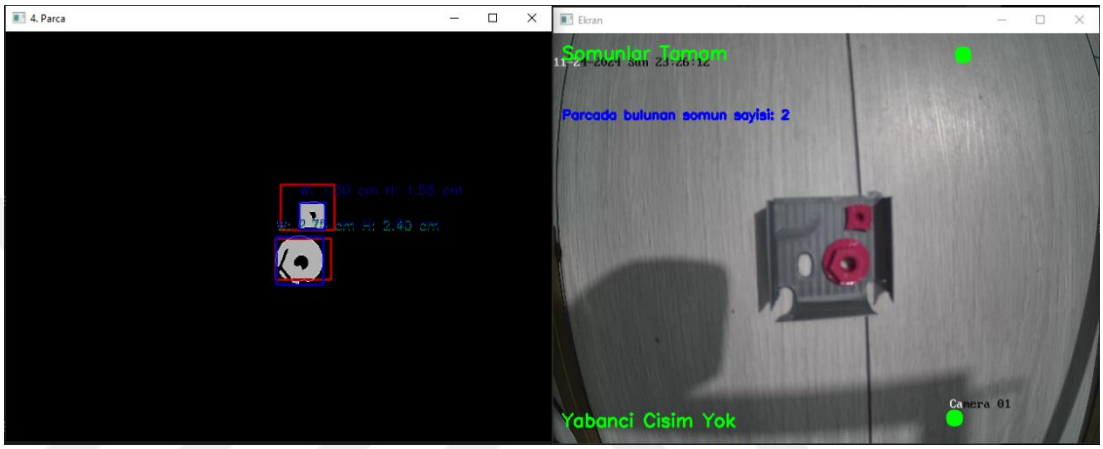
SONUÇ VE ANALİZ

Bu çalışmada, görüntü işleme teknikleri kullanılarak 8 farklı parça üzerinde somunların tespiti ve yabancı cisim kontrolü yapılmış, %100 başarı sağlanmıştır. Her bir parçaya ait somunların boyutları (küçük ve büyük somunlar) doğru bir şekilde ayrılmış, istenen sayıda somun tespiti eksiksiz olarak gerçekleştirilmiştir. Ayrıca, belirlenen bölgeler dışındaki yabancı cisimler de hassasiyetle algılanmıştır. Bu başarı, somun ve yabancı cisimlerin tespiti için kullanılan bu yöntemin etkinliğini göstermiştir. Yabancı cisimlerin tespiti sırasında, belirlenen boyut limitleri aşılmadığı sürece, tanımlı olmayan bölgelerde herhangi bir nesneye rastlanmamış ve yanlış pozitif sonuç elde edilmemiştir. Sistemin doğruluğu, ölçeklendirme faktörü ve maske tabanlı analizlerle desteklenmiş olup, somunların boyutları piksel bazında hassas ölçümlerle cm cinsine çevrilmiştir.

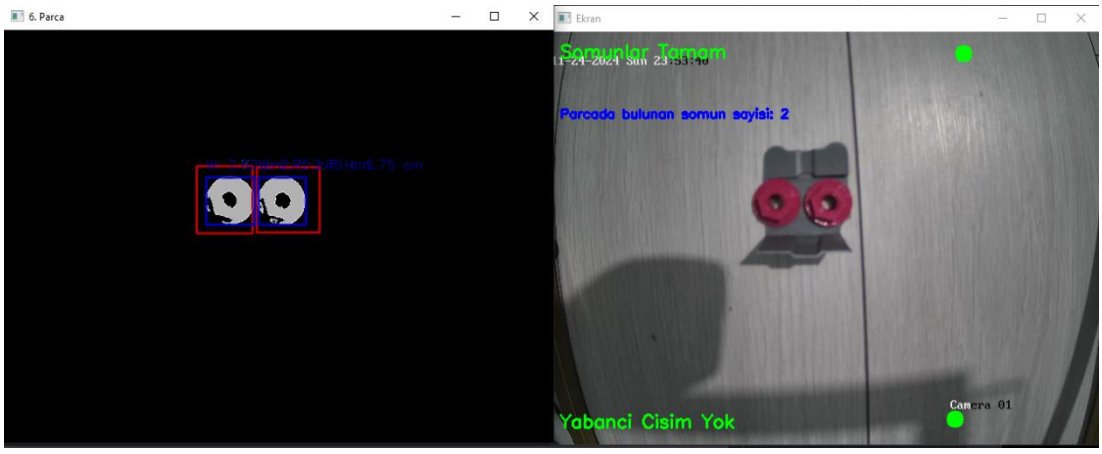





3. Parca
Ekran
Somunlar Tamam
Parcada bulunan somun sayisi: 1
Yabanci Cisim Yok
Camera 01


4. Parca
Ekran
Somunlar Tamam
Parcada bulunan somun sayisi: 2
Yabanci Cisim Yok
Camera 01


5. Parca
Ekran
Somunlar Tamam
Parcada bulunan somun sayisi: 2
Yabanci Cisim Yok
Camera 01


6. Parca
Ekran
Somunlar Tamam
Parcada bulunan somun sayisi: 2
Yabanci Cisim Yok
Camera 01

Şekil 35. Sistemde herhangi bir hata bulunmaması

Ancak, daha da iyileştirilebilecek noktalar mevcuttur;

İlk olarak, daha yüksek çözünürlüklü kameralarla görüntü kalitesi artırılarak somunların ve yabancı cisimlerin daha hassas tespiti sağlanabilir. İkinci olarak, ortamın ışıklandırması optimize edilerek renk algılamada daha yüksek doğruluk elde edilebilir. Üçüncü olarak, kontur algılama ve segmentasyon algoritmaları daha fazla optimize edilerek, özellikle küçük somunların tespitinde hassasiyet artırılabilir. Ayrıca, algoritmanın işlem hızı ve verimliliği artırılarak, gerçek zamanlı uygulamalarda daha hızlı sonuçlar alınabilir. Bu sistem, mevcut haliyle %100 doğrulukla çalışmış olsa da, bu önerilerle performans ve hassasiyetin daha da iyileştirilebileceği düşünülmektedir.

Sonuç olarak, bu tür bir görüntü işleme tabanlı kontrol sistemi, özellikle üretim hatlarında hızlı ve etkili bir kalite kontrol aracı olarak kullanıma uygundur. Somunların doğru ve eksiksiz monte edilip edilmediğini kontrol eden bu sistem, aynı zamanda yabancı maddelerin tespitini sağlayarak üretim kalitesini artırır. Sistemin üretim

süreçlerine entegre edilmesi hem maliyetleri düşürebilir hem de daha yüksek kaliteli ürünler elde edilmesine katkı sağlayabilir.

KAYNAKÇA

Ahsani, A. F., Sari, Y. A. ve Adikara, P. P. (2019, Eylül). Food image retrieval with gray level co-occurrence matrix texture feature and CIE L* a* b* color moments feature. In 2019 International Conference on Sustainable Information Engineering and Technology (SIET). 130-134 doi:10.1109/SIET48054.2019.8985990.

Akgün, M. (2005). Kalite maliyetlerinin faaliyet tabanlı maliyetleme sistemine entegrasyonu. *Muhasebe ve Denetime Bakış* (15), 31-48.

Amaricai, A., Gavriliu, C. E., ve Boncalo, O. (2014, Eylül). An FPGA sliding window-based architecture harris corner detector. 2014 24th International Conference on Field Programmable Logic and Applications (FPL). doi:10.1109/fpl.2014.6927402

Anand, A. (2011). BMP To JPEG-the conversion process.*Journal of Global Research in Computer Science*, 2(6), 145-150.

Arslan, E. (2011). Hücresel sinir ağı sistemleri kullanarak hareketli nesnelerin görüntü işleme uygulamaları (Yayınlanmamış doktora tezi). İstanbul Üniversitesi, İstanbul.

Asmaz, K. (2006). Görüntü işleme ile iki boyutlu cisimlerden grafik modelleri için veri eldesi (Yayımlanmamış yüksek lisans tezi). Yıldız Teknik Üniversitesi, İstanbul.

Bayram, B. R. (2019). Metal sektörü için görüntü işleme tabanlı bir kusurlu ürün tespit sistemi (Yayımlanmamış yüksek lisans tezi). Bursa Uludağ Üniversitesi, Bursa.

Bourke, P. (1998, Temmuz). BMP ımage format. BMP Files.

Boyacıgil, M. (2022). Görüntü işleme teknikleriyle bir endüstriyel kalite kontrol uygulaması (Yayımlanmamış yüksek lisans tezi). Necmettin Erbakan Üniversitesi, Konya.

Canny, J. F. (1986) A computational approach to edge detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 8(6) , 679-698. doi: 10.1109/tpami.1986.4767851

Chen, J., Zou, L. H., Zhang, J. ve Dou, L. H. (2009). The comparison and application of corner detection algorithms. *Journal of multimedia*, 4(6),435-441.

Christopoulos, C., Skodras, A. ve Ebrahimi, T. (2000). The JPEG2000 still ımage coding system: An overview. *IEEE Transactions on Consumer Electronics*, 46(4), 1103–1127. doi:10.1109/30.920468.

Çankaya, G., Boyacı, A., ve Yarkan, S. (2021). Kızılötesi damar görüntüsü işleme ve damar tespiti. *İstanbul Ticaret Üniversitesi Teknoloji ve Uygulamalı Bilimler Dergisi*, 3(2), 183-188.

Çayır, T. (2010). FBGA üzerinde görüntü iyileştirme uygulaması (Yayımlanmamış yüksek lisans tezi). Yıldız Teknik Üniversitesi, İstanbul.

Çelik, A. ve Tekin, E. (2020). Hough transform görüntü işleme yöntemiyle ekim makineleri için tohum sayma uygulaması. *Avrupa Bilim Ve Teknoloji Dergisi*, (Özel Sayı), 260-267. doi:10.31590/ejosat.araconf33

Çelik, K. (2015). Gradyan uyarlamalı görüntü filtresi tasarımı (Yayımlanmamış yüksek lisans tezi). Gazi Üniversitesi, Ankara

Çelik, S. (2020, Ekim). Görüntü İşleme Nedir? [Blog Yazısı]. Erişim adresi: https://www.simurgai.com/G%C3%B6r%C3%BCnt%C3%BC%C4%B0%C5%9Fleme-Nedir/

Çil, M. M. (2015). Temel görüntü işleme algoritmalarının fpga üzerinde gerçekleşmesi (Yayımlanmamış yüksek lisans tezi). İstanbul Teknik Üniversitesi, İstanbul.

Değirmenci, A., Çankaya, İ., ve Demirci, R. (2018). Gradyan anahtarlamalı gauss görüntü filtresi. *Düzce Üniversitesi Bilim ve Teknoloji Dergisi,* 6(1), 196-215. doi:10.29130/dubited.345116

Ecemiş, E., Güner, K., Kuran, U., Kuran, E. C. (2023). Evrişimli sinir ağlarında transfer öğrenmesi ile gan tarafından üretilen sahte görüntü tespiti. *Mühendislik Bilimleri ve Araştırmaları Dergisi,* 5(1), 98-107. doi: 10.46387/bjesr.1257332

Elkıran, H. (2020). OCC-OPENCV kütüphanesi için blok tabanlı programlama aracı (Yayımlanmamış yüksek lisans tezi). İstanbul Sabahattin Zaim Üniversitesi, İstanbul.

Erkan, U. ve Görkem, L. (2017). Tuz-biber gürültüsünde tekrarsız medyan filtre. *Gaziosmanpaşa Bilimsel Araştırma Dergisi*, 6(2), 11-19.

Frandsen, M. ve Zhang, Y. (2014). Gıf to html5 video: optimizing gıf for modern browsers. 2014 11th Web Information System and Application Conference, 75–78. doi: 10.1109/wisa.2014.22

Gonzalez, R.C., ve Woods, R. E. (2008). *Digital Image Processing.* Erişim adresi:https://dl.ebooksworld.ir/motoman/Digital.Image.Processing.3rd.Edition.www.EBooksWorld.ir.pdf

Güzel, F. ve Kurşunel, F. (2015). Kalite maliyetleri ve veri kalitesi. *Sosyal Ekonomik Araştırmalar Dergisi,* 15 (29), 282-301. doi: 10.30976/susead.302199

Horasan, D. (2016). Mermer yüzeylerin görüntü işleme yöntemi ile değerlendirilmesi (Tezsiz yüksek lisans bitirme projesi). Pamukkale Üniversitesi, Denizli.

Karacan, K., Uyar, T., ve Öztürk, M. K. (2018, Eylül) Görüntü işleme ve gözetimli makine öğrenme teknikleriyle ticari hava aracı sınıflandırma. VII. Ulusal Havacılık ve Uzay Konferansı, Ondokuz Mayıs Üniversitesi, Samsun.

Kefe, İ., ve Naci Tanış, V. (2014). Kalite maliyetleri ve otomotiv sektöründe bir uygulama. *World Of Accounting Science*, 16(1), 45-62.

Kılıç, B., ve Eleren, A. (2009). Turizm sektöründe hizmet kalitesi ölçümü üzerine bir literatür araştırması. *Uluslararası Alanya İşletme Fakültesi Dergisi,* 1(1), 91-118.

Kınalı, İ. (2009). Üretim otomasyon sistemlerinde antigeometrik yayınım ile ürün yüzey hatalarının tespiti (Yayımlanmamış Yüksek Lisans Tezi). Yıldız Teknik Üniversitesi, İstanbul.

Lee, J. P., Wu, Q. Q., Park, M. H., Park, C. K., ve Kim, I. S. (2015). A study on modified Hough algorithm for image processing in weld seam tracking system.*Advanced Materials Research, 1088*, 824–828. doi:10.4028/www.scientific.net/amr.1088.824

Mao, Hu., Hu, Z., Zhu, L. ve Qin, H. (2012). PNG file decoding optimization based embedded system. 2012 8th International Conference on Wireles Communications Networking and Mobile Computing. doi:10.1109/wicom.2012.6478619.

Mıstry, S. ve Patel, A. (2016). Image stitching using harris feature detection. *International Research Journal of Engineering and Technology (IRJET)*, 3(04), 2220-2226.

Moeslund, T. B. (2012*). Introduction to Video and Image Processing. Undergraduate Topics in Computer Science.* Londra:Springer

Nasseri, M. H., Moradi, H., Nasiri, S. M., ve Hosseini, R. (2018). Power line detection and tracking using Hough transform and particle filter. 2018 6th RSI International Conference on Robotics and Mechatronics *(IcRoM).* 130-134 doi:10.1109/icrom.2018.8657568

Onat, E. (2017, Mayıs). FPGA implementation of real-time video signal processing using Sobel, Robert, Prewitt, and Laplacian filters. 2017 25th Signal Processing and Communications Applications Conference (SIU). doi:10.1109/SIU.2017.7960586

Onat, E. (2018). FPGA implementation of target detection algorithm at real time video signal processing using Harris corner detector filter. 2018 26th Signal Processing and Communications Applications Conference (SIU), 1–4. doi:10.1109/SIU.2018.8404520

Öztürk, N. ve Öztürk, S. (2018). Görüntü bölütleme yöntemlerinin karşılaştırılması. 2018 International Conference on Artificial Intelligence and Data Processing (IDAP). doi:10.1109/idap.2018.8620739.

Parsa, Y., Hosseinzadeh, H. ve Effatparvar, M. (2015). Development hough transform to detect straight lines using pre-processing filter. *International Journal of Information, Security and Systems Management*, 4(2), 448-456.

Pehlivan, F. (2022). Görüntü işleme teknikleri ile diş renk skalasının belirlenmesi (Yayımlanmamış yüksek lisans tezi). Konya Teknik Üniversitesi, Konya.

Perihanoğlu, G. M. (2015). Digital görüntü işleme teknikleri kullanılarak görüntülerden detay çıkarımı (Yayımlanmamış yüksek lisans tezi). İstanbul Teknik Üniversitesi, İstanbul.

Qin, X. (2021). A modified Canny edge detector based on weighted least squares. *Computational Statistics*, 36, 641-659. doi:10.1007/s00180-020-01017-8.

Saabith, A. S., Fareez, M. M. M., ve Vinothraj, T. (2019). Python current trend applications-an overview. *International Journal of Advance Engineering and Research Development*, 6(10).

Shen, S., Zhang, X. ve Heng, W. (2010, Eylül). Auto-adaptive Harris corner detection algorithm based on block processing. 2010 International Symposium on Signals, Systems and Electronics(ISSSE). 1–4 doi:10.1109/ISSSE.2010.5638297

Shrivakshan, G. T. ve Chandrasekar, C. (2012, Eylül). A comparison of various edge detection techniques used ın ımage processing. *International Journal Of Computer Science Issues (IJCSI),* 9(5), 269.

Sindhu, M. ve Rajkamal, R. (2009, Kasım). Images and its compression techniques-A review. *International Journal of Recent Trends in Engineering,* 2(4), 71-75.

Şenel, H. G. (2007). Kenar bulma için topolojik gradyan işleçleri. *Eskişehir Osmangazi Üniversitesi Mühendislik ve Mimarlık Fakültesi Dergisi,* 20(2), 135-158.

Talan, T. ve Aktürk, C. (2021). *Bilgisayar Bilimlerinde Teorik Ve Uygulamalı Araştırmalar.* İstanbul: Efe Akademi Yayınları.

Tarı, M. (2020). Görüntü işleme teknikleri ile enjektör üretiminde kalite kontrolü (Yayımlanmamış yüksek lisans tezi). Konya Teknik Üniversitesi, Konya.

Taubman, D. S. ve Marcellin, M. W. (2002, Ağustos). JPEG2000: Standard for interactive imaging. *Proceedings Of The IEEE*, 90(8), 1336–1357. doi:10.1109/jproc.2002.800725.

Toklu, D. (2006). Görsel final kalite kontrol sistemleri (Yayımlanmamış yüksek lisans tezi). İstanbul Teknik Üniversitesi, İstanbul.

Turan, B. (2017). Görüntü işleme algoritmalarının eş zamanlı süreçlere ayrılarak kablosuz ağ üzerinden gerçeklenmesi ve performans analizleri (Yayımlanmamış yüksek lisans tezi). Maltepe Üniversitesi, İstanbul.

Villa, D. (2023). Digital hybridisation and communication of the design process: The case of animated GIFS. *Disegno*, 13, 81-88.

Wiggins, R. H., Davidson, H. C., Harnsberger, H. R., Lauman, J. R., ve Goede, P. A. (2001). Image file formats: Past, present, and future. *RadioGraphics*, 21(3), 789–798. doi:10.1148/radiographics.21.3.g01ma25789

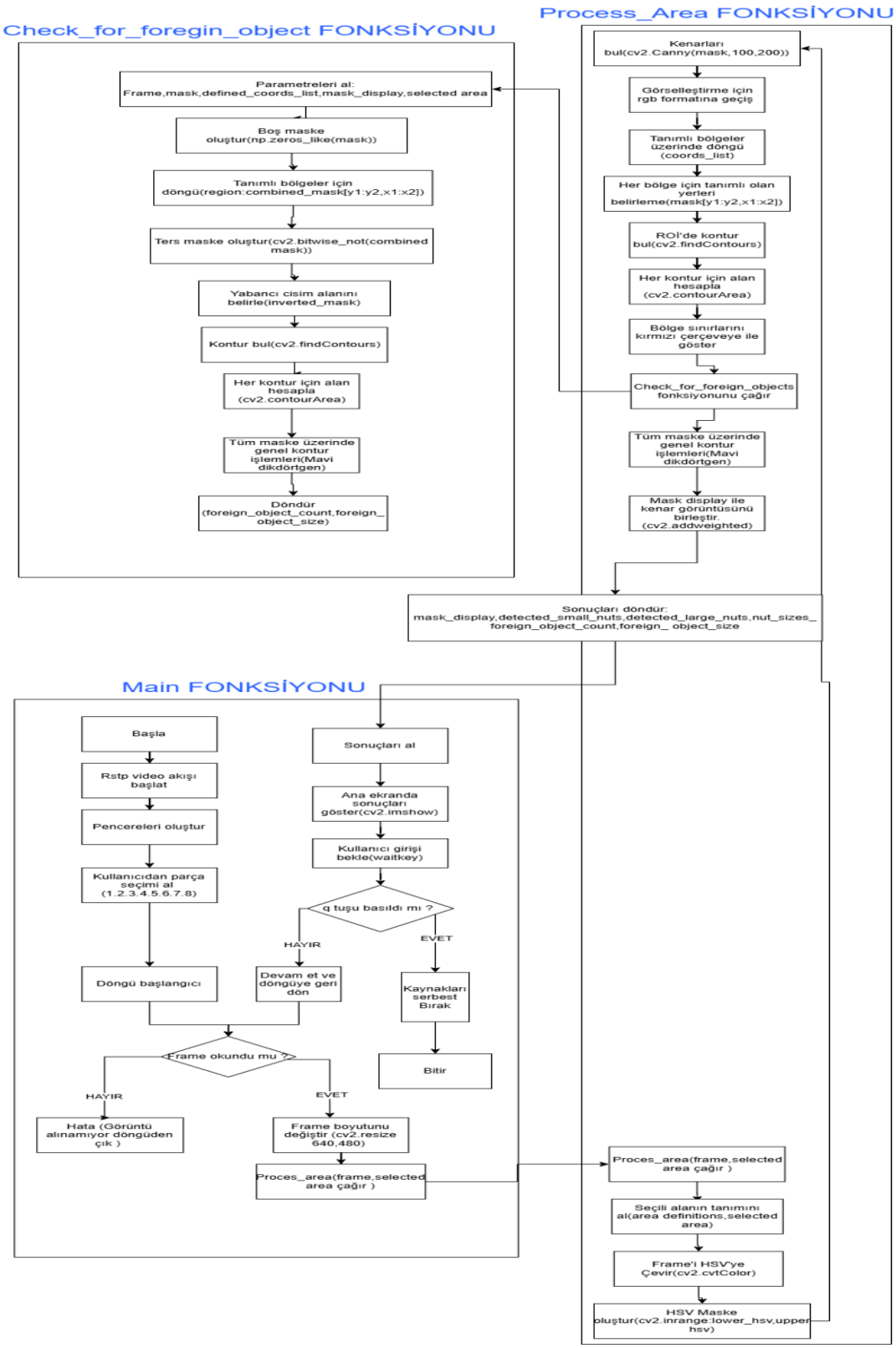
Xu, Z., Baojie, X., ve Guoxin, W. (2017). Canny edge detection based on Open CV. 2017 13th IEEE International Conference on Electronic Measurement and Instruments (ICEMI). 53-56. doi:10.1109/icemi.2017.8265710

Yoldaş, M. (2021). Alüminyum ekstrüzyon profillerinde kesit ölçümlerinin RGB kanallarındaki görüntülerle ölçülmesi (Yayımlanmamış yüksek lisans tezi). Konya Teknik Üniversitesi, Konya.

EKLER

EK-A

PROGRAMIN AKIŞ ŞEMASI



EK-B

PYTHON PROGRAMLAMA DİLİYLE SOMUNLARIN BULUNMASI

```
import cv2
import numpy as np
mouseX = 0
mouseY = 0*
# RTSP URL



rtsp_url = "rtsp://admin:Burakemin1@192.168.0.99:554/Streaming/Channels/101"
scale_factor = 0.05
area_definitions = {
    "1. Parca": [
        {
            "lower_hsv": np.array([97, 88, 101]),
            "upper_hsv": np.array([179, 255, 255]),
            "coords_list": [
                {"coords": (141, 205, 389, 454), "type": "small"},
                {"coords": (236, 297, 196, 262), "type": "small"}
            ],
            "required_small_nuts": 2,
            "required_large_nuts": 0
```

```
    }
],
"2. Parca": [
    {
        "lower_hsv": np.array([97, 88, 101]),
        "upper_hsv": np.array([179, 255, 255]),
        "coords_list": [
            {"coords": (161, 215, 231, 290), "type": "small"},
            {"coords": (160, 217, 309, 370), "type": "small"}
        ],
        "required_small_nuts": 2,
        "required_large_nuts": 0
    }
],
"3. Parca": [
    {
        "lower_hsv": np.array([97, 88, 101]),
        "upper_hsv": np.array([179, 255, 255]),
        "coords_list": [
            {"coords": (130, 183, 295, 366), "type": "small"}
        ],
        "required_small_nuts": 1,
        "required_large_nuts": 0
    }
],
"4. Parca": [
```

```
    {
        "lower_hsv": np.array([97, 88, 101]),
        "upper_hsv": np.array([179, 255, 255]),
        "coords_list": [
            {"coords": (179, 232, 322, 385), "type": "small"},
            {"coords": (242, 290, 316, 381), "type": "large"}
        ],
        "required_small_nuts": 1,
        "required_large_nuts": 1
    }
],
"5. Parca": [
    {
        "lower_hsv": np.array([97, 88, 101]),
        "upper_hsv": np.array([179, 255, 255]),
        "coords_list": [
            {"coords": (170, 251, 225, 293), "type": "small"},
            {"coords": (156, 217, 340, 416), "type": "small"}
        ],
        "required_small_nuts": 2,
        "required_large_nuts": 0
    }
],
"6. Parca": [
    {
        "lower_hsv": np.array([97, 88, 101]),
```

```
        "upper_hsv": np.array([179, 255, 255]),
        "coords_list": [
            {"coords": (158, 236, 224, 289), "type": "large"},
            {"coords": (159, 235, 294, 367), "type": "large"}
        ],
        "required_small_nuts": 0,
        "required_large_nuts": 2
    }
],
"7. Parca": [
    {
        "lower_hsv": np.array([97, 88, 101]),
        "upper_hsv": np.array([179, 255, 255]),
        "coords_list": [
            {"coords": (176, 241, 273, 339), "type": "large"},
            {"coords": (175, 237, 337, 389), "type": "small"}
        ],
        "required_small_nuts": 1,
        "required_large_nuts": 1
    }
],
"8. Parca": [
    {
        "lower_hsv": np.array([97, 88, 101]),
        "upper_hsv": np.array([179, 255, 255]),
        "coords_list": [
```

```
                {"coords": (125, 172, 229, 287), "type": "small"},
                {"coords": (200, 247, 228, 296), "type": "small"},
                {"coords": (213, 284, 350, 427), "type": "large"}
            ],
            "required_small_nuts": 2,
            "required_large_nuts": 1
        }
    ]
}
small_nut_range = (1.0, 2.0)
large_nut_range = (2.0, 3.0)

def check_for_foreign_objects(frame, mask, defined_coords_list, mask_display, selected_area):
    combined_mask = np.zeros_like(mask)
    for region in defined_coords_list:
        y1, y2, x1, x2 = region["coords"]
        combined_mask[y1:y2, x1:x2] = 255
    inverted_mask = cv2.bitwise_not(combined_mask)
    foreign_object_area = cv2.bitwise_and(mask, mask, mask=inverted_mask)
    contours, _ = cv2.findContours(foreign_object_area, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

    foreign_object_count = 0
    foreign_object_sizes = []

    for cnt in contours:
```

```
            area_contour = cv2.contourArea(cnt)
            if area_contour > 100:
                x, y, w, h = cv2.boundingRect(cnt)
                width_cm = w * scale_factor
                height_cm = h * scale_factor
                foreign_object_sizes.append((width_cm, height_cm))
                cv2.rectangle(mask_display, (x, y), (x + w, y + h), (0, 0, 255), 2)
                foreign_object_count += 1


                if selected_area == "1. Parca":
                    cv2.putText(mask_display, f"W: {width_cm:.2f} cm H: {height_cm:.2f} cm", (x, y - 10),
                                cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 1)

        return foreign_object_count, foreign_object_sizes


    def process_area(frame, selected_area):
        if selected_area not in area_definitions:
            print(f"{selected_area} geçerli bir bölge değil.")
            return frame, 0, [], [], 0, []

        area_info_list = area_definitions[selected_area]
        required_small_nuts = area_info_list[0]["required_small_nuts"]
        required_large_nuts = area_info_list[0]["required_large_nuts"]
        lower_hsv = area_info_list[0]["lower_hsv"]
```

```
            upper_hsv = area_info_list[0]["upper_hsv"]

            hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
            mask = cv2.inRange(hsv, lower_hsv, upper_hsv)

            mask_display = cv2.cvtColor(mask, cv2.COLOR_GRAY2BGR)
            detected_small_nuts = 0
            detected_large_nuts = 0
            nut_sizes = []
            foreign_object_count = 0
            foreign_object_sizes = []

            for area_info in area_info_list:
                coords_list = area_info["coords_list"]

                for region in coords_list:
                    y1, y2, x1, x2 = region["coords"]
                    area_type = region["type"]
                    area = mask[y1:y2, x1:x2]
                    contours, _ = cv2.findContours(area, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

                    for cnt in contours:
                        area_contour = cv2.contourArea(cnt)
                        if 100 < area_contour < 10000:
                            x, y, w, h = cv2.boundingRect(cnt)
```

```
                width_cm = w * scale_factor
                height_cm = h * scale_factor
                cv2.rectangle(mask_display, (x + x1, y + y1), (x + w + x1, y + h + y1), (0, 255, 0), 2)
                cv2.putText(mask_display, f"W: {width_cm:.2f} cm H: {height_cm:.2f} cm", (x + x1, y - 10 + y1),
                            cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 1)
                nut_sizes.append((width_cm, height_cm))
                if area_type == "small" and small_nut_range[0] <= width_cm <= small_nut_range[1] and small_nut_range[0] <= height_cm <= small_nut_range[1]:
                    detected_small_nuts += 1
                elif area_type == "large" and large_nut_range[0] <= width_cm <= large_nut_range[1] or large_nut_range[0] <= height_cm <= large_nut_range[1]:
                    detected_large_nuts += 1

            cv2.rectangle(mask_display, (x1, y1), (x2, y2), (0, 0, 255), 2)

        foreign_object_count, foreign_object_sizes = check_for_foreign_objects(frame, mask, coords_list, mask_display, selected_area)

    contours, _ = cv2.findContours(mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    for cnt in contours:
        area_contour = cv2.contourArea(cnt)
        if area_contour > 100:
            x, y, w, h = cv2.boundingRect(cnt)
            width_cm = w * scale_factor
            height_cm = h * scale_factor
```

```
                cv2.rectangle(mask_display, (x, y), (x + w, y + h), (255, 0, 0), 2)
                cv2.putText(mask_display, f"W: {width_cm:.2f} cm H: {height_cm:.2f} cm", (x, y - 10),
                            cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 0, 0), 1)

        return mask_display, detected_small_nuts, detected_large_nuts, nut_sizes, foreign_object_count, foreign_object_sizes


    def show_coordinates_and_pixel_values(event, x, y, flags, param):
        global mouseX, mouseY
        if event == cv2.EVENT_MOUSEMOVE or event == cv2.EVENT_LBUTTONDOWN:
            mouseX, mouseY = x, y


    def main():
        cap = cv2.VideoCapture(rtsp_url)
        cv2.namedWindow('Ekran', cv2.WINDOW_NORMAL)
        cv2.setMouseCallback('Ekran', show_coordinates_and_pixel_values)

        selected_area = input("Görmek istediğiniz parçayı seçin (örneğin: 1, 2, 3, 4, 5, 6, 7, 8): ").strip()
        selected_area = f"{selected_area}. Parca"

        while True:
            ret, frame = cap.read()
            if not ret:
                print("Kamera akışından görüntü alınamıyor.")
```

```
            break

        frame = cv2.resize(frame, (640, 480))
        total_detected_small_nuts = 0
        total_detected_large_nuts = 0
        all_sizes = []
        total_foreign_objects = 0
        all_foreign_sizes = []

        mask_display, detected_small_nuts, detected_large_nuts, nut_sizes, foreign_object_count, foreign_object_sizes = process_area(frame, selected_area)
        total_detected_small_nuts += detected_small_nuts
        total_detected_large_nuts += detected_large_nuts
        all_sizes.extend(nut_sizes)
        total_foreign_objects += foreign_object_count
        all_foreign_sizes.extend(foreign_object_sizes)

        cv2.imshow(selected_area, mask_display)
        cv2.putText(mask_display,
                    f"{selected_area}: {detected_small_nuts} küçük somun, {detected_large_nuts} büyük somun, Boyutlar: {', '.join([f'W: {w:.2f}xH: {h:.2f}' for w, h in nut_sizes])}",
                    (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 1)
```

```
            if selected_area == "1. Parca":
                for i, (w, h) in enumerate(all_foreign_sizes):
                    cv2.putText(mask_display, f"Foreign {i + 1}: W: {w:.2f} cm, H: {h:.2f} cm", (10, 50 + i * 20),
                                cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 1)

            cv2.putText(frame, f"Parcada bulunan somun sayisi: {total_detected_small_nuts + total_detected_large_nuts}", (10, 100), cv2.FONT_HERSHEY_SIMPLEX,
                        0.5, (255, 0, 0), 2)

            if total_detected_small_nuts == area_definitions[selected_area][0]["required_small_nuts"] and total_detected_large_nuts == area_definitions[selected_area][0]["required_large_nuts"]:
                somun_durumu = "Somunlar Tamam"
                color_somun = (0, 255, 0)
            else:
                somun_durumu = "Somun Eksik Veya Yanlis Somun Mevcut"
                color_somun = (0, 0, 255)

            cv2.putText(frame, somun_durumu, (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.7, color_somun, 2)
            cv2.circle(frame, (480, 25), 10, color_somun, -1)

            if total_foreign_objects > 0:
```

```
            yabanci_durum = f"Parcada {total_foreign_objects} adet yabancı cisim bulundu"
            color_yabanci = (0, 0, 255)
        else:
            yabanci_durum = "Yabanci Cisim Yok"
            color_yabanci = (0, 255, 0)

        cv2.putText(frame, yabanci_durum, (10, frame.shape[0] - 20), cv2.FONT_HERSHEY_SIMPLEX, 0.7, color_yabanci, 2)
        cv2.circle(frame, (frame.shape[1] - 240, frame.shape[0] - 30), 10, color_yabanci, -1)

        cv2.putText(frame, f"Koordinatlar: X={mouseX}, Y={mouseY}", (10, 50), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 255, 255), 2)

        cv2.imshow('Ekran', frame)

        key = cv2.waitKey(1)
        if key == ord('q'):
            break

    cap.release()
```

```
    cv2.destroyAllWindows()


if __name__ == "__main__":
    main()
```