T.C.
NECMETTİN ERBAKAN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYARLI GÖRÜ İLE 3 BOYUTLU KONUM BELİRLEME VE CNC ROBOT İLE KONUM DOĞRULAMA

Zeynel Emre TAY

ORC-ID: 0009-0001-8432-7793

YÜKSEK LİSANS TEZİ

Mekatronik Mühendisliği Anabilim Dalı

Haziran-2025
KONYA

TEZ KABUL VE ONAYI

Zeynel Emre TAY tarafından hazırlanan Bilgisayarlı Görü İle 3 Boyutlu Konum Belirleme Ve Cnc Robot İle Konum Doğrulama adlı tez çalışması 25.06.2025 tarihinde aşağıdaki jüri tarafından oy birliği ile Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü Mekatronik Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Doç. Dr. Ali YAŞAR	……..
Danışman Doç. Dr. Barış GÖKÇE	……..
Üye Doç. Dr. Ümit ÖNEN	……..

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun …./…/20.. gün ve …….. sayılı kararıyla onaylanmıştır.

Prof. Dr. Havvanur UÇBEYİAY
FBE Müdürü

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

İmza

Zeynel Emre TAY

25.06.2025

ÖZET

YÜKSEK LİSANS TEZİ

BİLGİSAYARLI GÖRÜ İLE 3 BOYUTLU KONUM BELİRLEME VE CNC ROBOT İLE KONUM DOĞRULAMA

Zeynel Emre TAY

Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü
Mekatronik Mühendisliği Anabilim Dalı

Danışman: Doç. Barış Gökçe
2025, 108 Sayfa

Jüri
Doç. Dr. Barış GÖKÇE
Doç. Dr. Ali YAŞAR
Doç. Dr. Ümit ÖNEN

Bu tez çalışmasında, endüstriyel kaynak dikişlerinin taşlama işlemlerini otomatikleştirmek amacıyla, dört eksenli bir CNC sistemi üzerine entegre edilmiş tek bir kamera kullanan, maliyet-etkin ve görüntü tabanlı yeni bir otomasyon mimarisi tasarlanmış ve performansı analiz edilmiştir. Geliştirilen yöntemde, çalışma alanının görüntüsü üzerinde belirlenen bir hedef bölge ilk olarak perspektif düzeltme ile ölçeklendirilmiş, ardından YOLOv8 tabanlı bir yapay zekâ modeli ile kaynak dikişinin merkezi tespit edilmiştir. Nesnenin üç boyutlu konumu ise, kameranın bilinen bir mesafede yanal hareketiyle elde edilen derinlik bilgisi ve görüntü düzlemindeki konumunun birleştirilmesiyle belirlenmiştir. Sistemin performansı, 150x150 mm'lik bir alanda gerçekleştirilen sistematik testlerle değerlendirilmiş; geliştirilen doğrusal paralaks modelinin Z ekseninde (derinlik) konuma bağlı sistematik hatalar sergilediği, ancak bu hatanın yayılımının nihai X ve Y konumlandırmasında ±2.5 mm aralığında kaldığı tespit edilmiştir. Sonuç olarak bu çalışma, geleneksel stereo sistemlere kıyasla daha düşük maliyetli bir donanımla, endüstriyel polisaj görevleri için yüksek tekrarlanabilirliğe sahip, tam otonom bir çözüm sunma potansiyelini ortaya koymuştur.

,

Anahtar Kelimeler: derinlik kestirimi, ROI, YOLOv8, perspektif dönüşümü, GRBL, PID kontrol, endüstriyel kaynak

ABSTRACT

MS THESIS

TEZ BAŞLIĞININ İNGİLİZCE'SİNİ BURAYA YAZINIZ

Zeynel Emre TAY

Computer Vision-Based 3D Position Determination and Position Verification with a CNC Robot
Advisor: Assoc. Prof. Dr. Barış Gökçe

2025, 108 Pages

Jury
Assoc. Prof. Dr. Barış GÖKÇE (Advisor)
Assoc. Prof. Dr. Ali YAŞAR
Assoc. Prof. Dr. Ümit ÖNEN

In this thesis, a novel, cost-effective, and vision-based automation architecture was designed and its performance analyzed for the purpose of automating the grinding processes of industrial weld seams, utilizing a single camera integrated into a four-axis CNC system. In the developed method, a target region of interest on the workspace image was first scaled via perspective correction, after which the center of the weld seam was detected using a YOLOv8-based artificial intelligence model. The three-dimensional position of the object was then determined by combining the depth information, obtained through the lateral movement of the camera over a known distance, with its position on the image plane. The system's performance was evaluated through systematic tests conducted over a 150x150 mm area; it was determined that the developed linear parallax model exhibited position-dependent systematic errors in the Z-axis (depth), though the propagation of this error remained within a ±2.5 mm range in the final X and Y positioning. In conclusion, this study has demonstrated the potential of the proposed approach to offer a highly repeatable and fully autonomous solution for industrial polishing tasks with lower-cost hardware compared to traditional stereo systems.

Keywords: depth estimation, ROI, YOLOv8, perspective transformation, GRBL, PID control, industrial weld polishing

ÖNSÖZ

Bu çalışma için bana yeterli imkanları sağlayıp hem bu çalışmada hem de bana çok büyük tecrübeler kattığını düşündüğüm diğer çalışmalarımda destek olarak rehberlik eden çok değerli danışmanım sayın Doç. Dr. Barış Gökçe'ye, her türlü kararda beni destekleyen ve hem maddi hem de manevi olarak yanımda olan aileme sonsuz saygı ve teşekkürlerimi sunarım.

Zeynel Emre TAY
KONYA-2025

İÇİNDEKİLER

SİMGELER VE KISALTMALAR

Simgeler

B: Kameranın X eksenindeki bilinen öteleme (mm)

Δx: İki görüntü arasındaki piksel kayması (piksel)

f: Odak uzaklığı (piksel)

H_{cam}: Kamera–merkez noktası yüksekliği (mm)

W_{cam}: Kamera–merkez noktası yatay uzaklığı (mm)

D_{cam}: Kamera-merkez noktası dikey uzaklığı (mm)

Z: Hesaplanan derinlik (mm)

θ: Aparat eğim açısı (°)

Kısaltmalar

ADC: Analog-Dijital Dönüştürücü (Analog-to-Digital Converter)

AI: Yapay Zekâ (Artificial Intelligence)

CNC: Bilgisayarlı Sayısal Kontrol (Computer Numerical Control)

DOF: Serbestlik Derecesi (Degrees of Freedom)

FOV: Görüş Alanı (Field of View)

GEV: GigE Vision

GigE: Gigabit Ethernet

GRBL: Açık kaynak G-kod yorumlayıcı (Open Source G-code firmware)

PID: Oransal-İntegral-Türevsel Denetleyici (Proportional–Integral–Derivative)

POE: Power over Ethernet (Ethernet Üzerinden Güç)

RMS: Root Mean Square (Karekök Ortalama Kare)

ROI: Region of Interest (İlgi Alanı)

SDK: Yazılım Geliştirme Kiti (Software Development Kit)

SFM: Hareketten Yapı Çıkarımı (Structure from Motion)

YOLOv8: "You Only Look Once", 8. sürüm gerçek-zamanlı algı modeli

1. GİRİŞ

Derinlik algısı, insan retinasında oluşan iki boyutlu görüntülerin merkezi sinir sistemi tarafından işlenerek üç boyutlu uzaysal bilgiye dönüştürülmesi olgusudur (Marr, 1982). Bu algı, aynı nesnenin iki retina üzerinde oluşturduğu konum farklılığından kaynaklanan geometrik ilişkiye dayanır (Taşel, 2008). Benzer bir geometrik düzenin kameralarla sağlanması yani sahnenin farklı noktalardan görüntülenmesi işlenen kareler üzerinde derinliğin belirlenmesine ve dolayısıyla odaklanılan bir noktanın referans sistemine göre uzaydaki konumunun hesaplanmasına imkân verir.

Günümüzde yüksek hassasiyetli konum saptama için yapılandırılmış ışık, lazer triangülasyon ve stereo görüş gibi yöntemler kullanılmakla birlikte, ışık yansımaları, gölgelenme ve karmaşık kalibrasyon gereksinimleri bu yaklaşımların endüstriyel üretim hatlarında kesintisiz biçimde uygulanmasını güçleştirmektedir (Sansoni, Trebeschi, & Docchio, 2009). Derin öğrenme tabanlı modeller, uygun donanımla saniyede 48 kareyi aşan hızlara ulaşsa da ışık koşullarındaki değişimlere duyarlı olmaları ve hedeflenen < 1 mm hata payını geniş çalışma alanlarında sağlayabilmek için ek öğrenme adımları gerektirmeleri, üretim hattı entegrasyonunda ilave zaman ve maliyet yükü yaratmaktadır (Laga ve ark., 2022 ; Dong ve ark., 2022).

Bu tezde, söz konusu kısıtları aşmak üzere tek endüstriyel kameraya dayalı, gerçek zamanlı ve düşük maliyetli bir görüntü-yönelimli otomasyon mimarisi geliştirilmiştir. Önerilen mimari, dört eksenli GRBL tabanlı CNC robotunu ± 0,5 mm konum hassasiyetiyle yönlendirebilen bir polisaj sistemi üzerine inşa edilmiştir. İlk aşamada, çalışma uzayının fiziksel boyutları referans alınarak görüntüde seçilen bölge (ROI), homografi-temelli perspektif dönüşüme tabi tutulmuş; böylece doğrultulmuş karede 1 piksel ≈ 1 mm metrik ölçek elde edilmiştir. Bu adımı, ROI içinde kaynak noktasının merkezini tespit eden YOLOv8 temelli bir yapay zekâ modelinin kullanımı izlemiştir (Jocher ve ark., 2023).

Ardından, kameranın X ekseninde 100 mm'lik bilinen bir öteleme hareketi gerçekleştirilmiştir. Aynı noktanın iki karedeki piksel kayma miktarı (Δx) ölçülerek benzer üçgen ilişkisiyle gerçeğe yakın derinlik (Z) hesaplanmıştır. Elde edilen (X, Y, Z) koordinatları, PID denetimli JOG komutları aracılığıyla CNC'nin X, Y, Z ve A (torç eğim açısı) eksenlerine gerçek zamanlı olarak iletilmiştir (McGinnis, 2019; Hartley & Zisserman, 2004). Bu geri besleme sayesinde, taşlama aparatının yüzey eğimlerini 0,01 mm RMS hatanın altında takip edebilmesi sağlanmıştır.

Önerilen yöntem, yapay zekâyı yalnızca tek aşamalı nokta tespitinde kullanarak ölçek belirsizliğini ortadan kaldırmış, stereoskopik duyarlılığı tek kamera ile sağlamıştır. Kamera kalibrasyonu, perspektif düzeltme, piksel-milimetre dönüşümü, derinlik çıkarımı ve GRBL senkronizasyonundan oluşan eksiksiz veri hattının çevrim-içi çalışması, operatör müdahalesini en aza indirerek endüstriyel polisaj süreçlerinde yüksek tekrarlanabilirlik ve düşük duruş süresi sunmuştur.

Bu araştırmanın özgün katkıları şu şekilde özetlenebilir:

- *Tek kamerayla stereoskopik duyarlılık*: Fiziksel öteleme bağlı paralaks tabanlı derinlik hesabı, ek donanım gerektirmeden milimetrik düzeyde hassasiyet sağlamıştır.
- *Metrik ROI ölçeklemesi:* Homografi ile piksel = mm eşitliği; karmaşık dış kalibrasyon ihtiyacını ortadan kaldırmıştır (Hartley ve & Zisserman, 2004).
- *Gerçek zamanlı CNC geri beslemesi:* PID + GRBL JOG entegrasyonu, 0,01 mm RMS yüzey izleme hatasıyla endüstriyel toleransları karşılamıştır.
- *Hızlı ve pratik veri kümesi oluşturma:* Arttırma (Augmentations) destekli artırma ve iptal edilebilir YOLOv8 eğitimi, hattı durdurmadan model güncellemesine imkân tanımıştır.

2. KAYNAK ARAŞTIRMASI

2.1. Tek Kamera ile Derinlik Tahmini ve 3B Konum Belirleme Yöntemleri

Son yıllarda derin öğrenme tabanlı monoküler derinlik tahmini (tek kameradan derinlik çıkarımı) alanında önemli ilerlemeler kaydedilmiştir. Derin konvolüsyonel sinir ağları, tek bir RGB görüntüden her pikselin göreceli derinliğini tahmin etme görevinde başarılı sonuçlar elde etmektedir (Dong ve ark., 2022). Örneğin, Fu ve arkadaşları derinlik tahminini bir sıralı sınıflandırma problemine dönüştürerek yeni bir çözünürlük ölçekleme yaklaşımı sunmuş ve tek görüntüden daha kesin derinlik haritaları elde etmişlerdir (Fu ve ark., 2018). Benzer şekilde, kendinden denetimli (self-supervised) yöntemler sol-sağ görüntü tutarlılığını kullanarak derinlik öğrenimini mümkün kılmıştır. Godard ve arkadaşları stereoskopik veriye ihtiyaç duymadan, tek bir kamerayla çekilen videolardan derinlik öğrenen bir model geliştirerek tutarlı ve kararlı sonuçlar elde etmişlerdir (Godard ve ark., 2019). Veri çeşitliliğini artırma yoluyla modellerin genelleme yeteneğini yükseltmek de bir diğer yaklaşımdır; Ranftl ve arkadaşları farklı veri kümelerini harmanlayarak tek kamera derinlik ağlarının yeni ortamlara uyumunu iyileştirmiştir (Ranftl ve ark., 2022). Ancak, monoküler derinlik tahmininde en büyük zorluk gerçek metrik ölçeğin belirsiz olmasıdır; ağlar genellikle göreceli (ölçeksiz) derinlik üretir ve çıktıların gerçek dünyadaki metrik karşılığını belirlemek ek bilgi gerektirir (Guizilini ve ark., 2023). Bu sorunu aşmak için bazı çalışmalar ağ mimarisine ölçek ipuçları entegre etmekte veya sahne içerisindeki bilinen boyutlardan faydalanmaktadır. Örneğin, kamera parametrelerini veya sahnedeki bir referans objenin boyutunu kullanarak mutlak ölçekli derinlik haritaları üreten yöntemler mevcuttur (Guizilini ve ark., 2023). Ayrıca, son dönemde vizyon transformatörleri de monoküler derinlik için kullanılmaya başlanmıştır. Ranftl ve arkadaşları, görüntü bölgesel ilişkilerini daha iyi modellemek için transformatör mimarisi ile yoğun derinlik tahmini yaparak ayrıntı kalitesini artırmışlardır (Ranftl ve ark., 2021). Bunun yanı sıra, tek görüntüden derinlik tahmin modellerinin genelleme kabiliyetini ve görece öğrenimini inceleyen tez çalışmaları bulunmaktadır. Örneğin, tek kameradan gözetimsiz derinlik çıkarımının farklı sahnelere uyarlanması konusunda 2020 yılına ait bir çalışma, derinlik ağlarının dinamik nesneler ve yeni ortamlar karşısındaki performansını iyileştirmeye odaklanmıştır (Varma, 2020).

Monoküler derinlik tahmini sonuçları, 3B konum belirleme için kullanılabilir. Tek kamera ile 3B konum belirleme, genellikle bir nesnenin görüntüdeki konum bilgilerini ve tahmin edilen derinliğini birleştirerek gerçekleştirilmektedir. Örneğin, bir nesnenin görüntü koordinatları biliniyor ve o nesneye ait tahmini derinlik haritası elde edilmişse, kamera iç parametreleri yardımıyla nesnenin gerçek dünyadaki konumu hesaplanabilir. Literatürde,

derinlik tahmini ve 3B yeniden yapılandırma için klasik yaklaşımlar da bulunmaktadır. Özellikle Hareketten Yapı Çıkarımı (SfM) gibi yöntemler, hareketli bir kameranın birden fazla görüntüsünden hem kamera pozlarını hem de sahnenin 3B nokta bulutunu çıkarabilmektedir. Fakat SfM yöntemleri çoklu görüntü gerektirdiğinden bu bölümde tek görüntüye dayalı yöntemler üzerinde durulmuştur. Sonuç olarak, tek kameradan 3B bilgi çıkarımı alanında yapılan çalışmalar, derin öğrenmenin katkısıyla hem doğruluk hem de hız bakımından büyük ilerleme göstermiştir (Dong ve ark., 2022)(Ranftl ve ark., 2022)(Ranftl ve ark., 2021).

2.2. Çift Kamera (Stereo Görüş) ile Derinlik Algılama Yöntemleri

İki kamera kullanan stereo görüş sistemleri, aynı sahnenin farklı açılardan alınan görüntülerindeki paralaks farkını kullanarak derinlik hesaplar. Stereo kameraların donanım bazında doğrudan metrik derinlik ölçümü yapabilmesi, monoküler sisteme göre önemli bir avantajdır. Elde edilen disparite (piksel kayması) değerleri, kameralar arası baz mesafesi ve odak uzunluğu yardımıyla gerçek dünyadaki mesafelere dönüştürülebilir. Son beş yılda stereo görüntülerden derinlik çıkarma konusunda derin öğrenme tabanlı yöntemler ön plana çıkmıştır. Laga ve arkadaşlarının derlemesine göre, derin öğrenme teknikleriyle stereo derinlik kestirimi konusundaki çalışmalar doğruluk ve hız bakımından klasik yaklaşımları geride bırakmıştır (Laga ve ark., 2022). Klasik stereo algoritmaları (örn. blok eşleştirme, Görüş geometrisi yöntemleri) yoğun eşleşme optimizasyonu yaparak disparite haritaları üretirken, güncel yaklaşımlar bu problemi uçtan uca öğrenme ile çözmektedir. Örneğin, Chang ve Chen, Pyramid Stereo Matching Network (PSMNet) adını verdikleri derin ağ ile çok ölçekli özellikler kullanarak stereo eşleştirmede yüksek doğruluk elde etmişlerdir (Chang ve & Chen, 2018). Benzer şekilde, Zhang ve arkadaşları derin sinir ağlarında rehberli maliyet birleştirme (guided aggregation) stratejisiyle GA-Net modelini geliştirmiş ve gerçek zamanlıya yakın performansla üst düzey sonuçlar bildirmişlerdir (Zhang ve ark., 2019). Bu modeller, stereo görüntü çiftleri arasındaki eşlenik noktaları öğrenme yoluyla bulup güvenilir disparite (derinlik) haritaları üretebilmektedir. Ayrıca, derin öğrenmeyle tekrarlamalı güncelleme yapan yeni yaklaşımlar da ortaya çıkmıştır. Lipson ve arkadaşları, RAFT-Stereo adını verdikleri modelde yinelenen optimizasyon adımlarıyla farklı ölçeklerde disparite hesaplayarak hem doğruluk hem hız açısından üstün bir yöntem sunmuşlardır (Lipson ve ark., 2021). Stereo derinlik algılama konusunda pratik uygulamaların artmasıyla, gerçek dünya koşullarında bu yöntemlerin performansını değerlendiren çalışmalar da yapılmaktadır. Örneğin, Nordh ve Vikén'in yüksek lisans tezinde stereo kameraların farklı ortamlarda gerçek zamanlı derinlik çıkarımı detaylı

şekilde incelenmiş, çeşitli CNN tabanlı yöntemlerin açık hava ve zorlu koşullardaki başarımları karşılaştırılmıştır (Nordh ve & Vikén, 2021). Benzer biçimde, derin öğrenme ile stereo görüntülerden 3B konum tespiti üzerine odaklanan yakın tarihli bir tez çalışması da farklı mimarileri deneyerek derinlik ve pozisyon kestirimindeki başarımları kıyaslamıştır (Nicholson, 2022). Genel olarak, çift kameralı sistemler doğru kalibre edildiklerinde güvenilir derinlik bilgilerinin elde edilmesini sağlar ve bu sayede 3B konum belirleme problemlerinde yaygın biçimde kullanılırlar.

2.3. 3B Nesne Konumlandırma Yöntemleri ve Uygulamaları

Gerçek dünyada bir nesnenin üç boyutlu konumunun (ve gerekirse yöneliminin) belirlenmesi, bilgisayarlı görü ve robotik uygulamalar için kritik bir adımdır. Literatürde, 6 serbestlik dereceli (6DoF) nesne duruş kestirimi olarak da anılan bu problem için çeşitli yaklaşımlar geliştirilmiştir. Son 5 yılda özellikle derin öğrenme temelli yöntemler bu alana hâkim olmaya başlamıştır. Fan ve arkadaşları, tek kameradan nesne tespiti ve takibi ile 6DoF poz tahmini üzerine kapsamlı bir derleme sunarak son gelişmeleri özetlemişlerdir (Fan ve ark., 2022). Bu alandaki yöntemler genellikle iki ana yaklaşıma ayrılmaktadır. Bunlardan ilki olan doğrudan poz regresyonu yönteminde, bir sinir ağı giriş görüntüsünü kullanarak nesnenin üç boyutlu konumunu ve dönüşümünü doğrudan tahmin eder (Fan ve ark., 2022). İkinci yaklaşım olan anahtar nokta tespiti veya 2B/3B eşleştirme yönteminde ise, öncelikle görüntü üzerinde nesneye ait belirgin noktalar veya bir segmentasyon maskesi bulunur ve ardından bu bilgiler kullanılarak kamera modeliyle ilişkili 3B konum hesaplanır (Du ve ark., 2021).Du ve arkadaşlarının çalışmasında, nesnenin önce 2B olarak lokalize edilip ardından PnP (Perspective-n-Point) gibi yöntemlerle poz kestiriminin yapıldığı ve nihayetinde robot kavrama işleminin gerçekleştirildiği bütüncül bir boru hattı anlatılmıştır (Du ve ark., 2021). Bu yaklaşımlar, özellikle endüstriyel robotik kavrama uygulamalarında yaygındır: kamera görüntüsünde nesneyi tanıyıp konumunu bulan sistem, elde edilen 6 serbestlik dereceli poz bilgisini robot kontrolcüsüne ileterek nesnenin tutulmasını sağlar. Nitekim, derin öğrenme tabanlı nesne konumlandırma yöntemleri sayesinde rastgele yığılmış parçaların tanınarak robotlarla alınması (bin-picking) mümkün hale gelmiştir. Sun ve arkadaşları, CAD modeli olmayan nesneler için tek bir örnek görüntüyle öğretim yapıp nesnenin pozunu bulabilen OnePose adlı yöntemi önermiş ve doğrultu kestiriminde yeni bir yaklaşım ortaya koymuştur (Sun ve ark., 2022). Bu yöntem, bir nesnenin yalnızca bir adet referans görüntüsünden yola çıkarak o nesnenin farklı açılardaki pozlarını tahmin edebilmektedir. Öte yandan, birden fazla

kameradan veya çoklu görüşten yararlanan yöntemler de konum belirleme doğruluğunu artırmaktadır. Labbe ve arkadaşlarının CosyPose adlı çalışmasında, birden fazla görüntüde tespit edilen nesnelerin pozları tutarlı bir şekilde optimize edilmiş ve çoklu-görüş birleşimi ile sahnedeki tüm nesnelerin konumları birlikte iyileştirilmiştir (Labbe ve ark., 2020). Bu tür yöntemler, özellikle birden fazla nesnenin bulunduğu ve kısmi gizlenmelerin (occlusion) olduğu ortamlarda, tüm nesnelerin pozlarını tutarlı biçimde bulma avantajı sağlamaktadır.

Nesne konumlandırma teknikleri, farklı uygulama alanlarında başarıyla kullanılmaktadır. Örneğin, artırılmış gerçeklik (AR) uygulamalarında kameradan algılanan referans nesnelerin 3B konumlarının doğru belirlenmesi, sanal içeriklerin gerçek dünya üzerine doğru şekilde bindirilmesi için esastır. Benzer şekilde, otonom araçlar kameraları aracılığıyla trafikteki diğer araç ve yayaların 3B konumlarını algılamak durumundadır; bu amaçla geliştirilmiş derin öğrenme tabanlı 3B nesne algılama yaklaşımlarının kapsamlı bir incelemesi Arnold ve arkadaşlarınca sunulmuştur (Arnold ve ark., 2019). Endüstride ise robotik kollar, montaj hattında kameralar yardımıyla parçaların konumunu saptayıp doğru yerleştirme yapabilmektedir. Bu bağlamda, konum belirleme ve doğrulama tekniklerinin güvenilirliği, tam otonom üretim sistemleri için kritik önemdedir. Sonuç olarak, 3B nesne konumlandırma alanında gerek derin öğrenme temelli gerekse geleneksel pek çok yöntem geliştirilmiş olup her birinin farklı uygulama senaryolarında avantajları bulunmaktadır (Fan ve ark., 2022)(Du ve ark., 2021)(Labbe ve ark., 2020).

2.5. Görüntü İşleme Teknikleri ile Nesne Ayıklama ve Konum Hesaplama

Bir görüntüdeki nesnenin 3B konumunu hesaplayabilmek için öncelikle o nesnenin arka plandan doğru şekilde ayrıştırılması (segmentasyonu) gerekir. Klasik görüntü işleme teknikleri, nesne ayıklama konusunda uzun süredir başarıyla kullanılmaktadır. Örneğin, renk eşikleme, kenar bulma ve morfolojik işlemler yardımıyla hedef nesne pikselleri arka plandan ayrılabilir (Gonzalez ve & Woods, 2018). Basit bir senaryoda, nesnenin rengi arka plandan farklı ise, görüntünün histogram analizi ile uygun bir eşik değeri belirlenip ikili maske oluşturulabilir; ardından erozyon-genişleme gibi morfolojik işlemler uygulanarak gürültüler temizlenip nesnenin şekli belirginleştirilebilir. Gonzalez ve Woods tarafından sunulan dijital görüntü işleme teknikleri literatürde bu tür yöntemlerin teorik temelini ayrıntılı olarak ele almaktadır (Gonzalez ve & Woods, 2018). Nitekim, matematiksel morfoloji tabanlı nesne tespiti üzerine yapılan araştırmalar, belirli şekil ve boyuttaki nesnelerin kalıp eşleştirme yoluyla görüntüden bulunabileceğini göstermiştir. Bu klasik yöntemler hesaplama açısından verimli olup kontrollü

ortamlarda yeterli olabilir; ancak arka planın karmaşık olduğu veya aydınlanmanın değişkenlik gösterdiği durumlarda daha gelişmiş tekniklere ihtiyaç duyulmaktadır.

Günümüzde, derin öğrenme tabanlı nesne tespiti ve segmentasyon algoritmaları, karmaşık sahnelerde nesne ayıklama için standart haline gelmiştir. Tek aşamalı nesne algılama ağları (YOLO gibi) gerçek zamanlı olarak birden fazla nesneyi tespit edip sınıflandırabilmektedir (Bochkovskiy ve ark., 2020). Bochkovskiy ve arkadaşlarının geliştirdiği YOLOv4 modeli, önceki nesil algılayıcılara kıyasla hem hız hem de doğrulukta önemli iyileştirmeler sunmuş ve gerçek zamanlı nesne tespitinde yaygın şekilde kullanılmıştır (Bochkovskiy ve ark., 2020). Bu tür bir dedektör, görüntüdeki nesneleri dikdörtgen kutular ile işaretleyerek konumlarını 2B düzlemde belirler. Nesnenin piksel koordinatları (örneğin kutunun merkez noktası), 3B konum hesaplamada başlangıç noktası olarak kullanılabilir. Nesnenin tam şeklinin çıkarılması gereken durumlarda ise maske tabanlı segmentasyon yöntemleri devreye girer. He ve arkadaşlarının geliştirdiği Mask R-CNN modeli, tespit edilen her nesne için piksel düzeyinde bir maskeyi yüksek doğrulukla üretebilmektedir (He ve ark., 2020). Bu sayede, nesnenin görüntüde kapladığı alan net bir biçimde ayrılarak arka plandan tamamen izole edilir. Derin öğrenme segmentasyon yöntemleri, özellikle birden fazla nesnenin bulunduğu karmaşık sahnelerde, örtüşen nesnelerin ayrıştırılması ve sınırlarının hassas biçimde çizilmesinde başarılıdır. Bunun bir örneği, Chen ve arkadaşlarının DeepLabv3+ modelidir; bu model atrous evrişim ve kodlayıcı-kod çözücü mimarisi ile nesne sınırlarında yüksek duyarlılığa ulaşmıştır (Chen ve ark., 2018). Derin segmentasyon çıktılarını kullanarak bir nesnenin görüntü içindeki tam konumunu (örneğin kütle merkezi ya da belirli bir referans noktası) piksel cinsinden tespit etmek mümkün olur.

Segmentasyon veya tespit adımı tamamlandıktan sonra, elde edilen 2B konum bilgisinin gerçek dünyadaki karşılığının hesaplanmasına geçilir. Eğer ilgili yöntemin çıktısı sadece 2B konum ise, üçüncü boyut bilgisini (derinlik) elde etmek için ya monoküler derinlik tahmini sonuçlarına ya da stereo/derinlik sensör verisine ihtiyaç duyulur. Örneğin, tek kamera durumunda önceki bölümde bahsedilen derinlik tahmin tekniklerinden elde edilen değer, nesnenin piksellerine karşılık gelen mesafe bilgisini verir. Alternatif olarak, görüntü yerine bir derinlik kamerası (örn. Kinect) kullanılıyorsa, her piksele ait derinlik doğrudan ölçülmüş olur. Bu durumda nesnenin segmentasyon maskesi uygulandığında, maskede kalan her piksel için (x, y, derinlik) bilgisi mevcut hale gelir. Bu noktada, kamera kalibrasyon parametreleri kullanılarak piksel koordinatları metrik koordinatlara dönüştürülebilir. Kısacası, görüntü işleme teknikleri nesnenin konumunu görüntü düzleminde belirlerken, ek derinlik bilgisi ile bu konumun 3B uzaydaki karşılığı hesaplanabilmektedir.

2.6. Perspektif Dönüşüm, Kamera Kalibrasyonu ve Konumun Metriğe Dönüştürülmesi

Görüntü koordinatlarından gerçek dünya koordinatlarına geçiş yapabilmek için kamera kalibrasyonu ve perspektif geometrisi konularına ihtiyaç duyulur. Kamera kalibrasyonu, bir kameranın iç parametrelerini (odak uzaklığı, optik merkez, bozulma katsayıları vb.) belirleme işlemidir. Zhang'ın klasik çalışmasıyla yaygınlaşan esnek kalibrasyon yönteminde, farklı açılardan çekilmiş bir dama tahtası desenine ait görüntüler kullanılarak kameranın parametreleri hesaplanabilir (Hartley ve & Zisserman, 2004). Kalibrasyon sonucunda elde edilen parametreler, piksel koordinatları ile kamera ışınları arasındaki ilişkiyi kurar. Szeliski'nin bilgisayarlı görünün temellerini anlatan eserinde, tek kameranın modellemesi ve kalibrasyon teknikleri ayrıntılı şekilde ele alınmaktadır (Szeliski, 2022). Kalibre edilmiş bir kamera modeli sayesinde, görüntüdeki bir pikselin kamera koordinat sisteminde hangi doğrultuya karşılık geldiği bilinir. Eğer o piksele ait bir derinlik veya mesafe bilgisi varsa, ilgili nesne noktası kamera koordinat sisteminde üç boyutlu olarak bulunabilir. Bu süreçte, perspektif projeksiyon modeli devreye girer: Gerçek dünyadaki bir noktanın kameraya göre (X, Y, Z) koordinatları, kalibrasyon parametreleri kullanılarak görüntü düzlemindeki (x, y) piksel konumuna projekte edilir. Bu ilişkinin tersi, yani görüntü noktasından 3B ışın oluşturma ve uygun derinlikte kesişim almak suretiyle gerçek konumu hesaplama, monoküler konum belirlemenin temelini oluşturur [Szeliski, 2022 ; Hartley & Zisserman, 2004].

Kamera kalibrasyonuna ek olarak, gerçek dünya koordinat sisteminin tanımlanması için dış kalibrasyon (kamera konumunun dünyaya göre konumu/orientasyonu) gereklidir. Bir CNC robota entegre kamera söz konusuysa, kameranın robot üzerindeki montaj pozisyonunun kalibrasyonu yapılmalıdır. Bu problem, literatürde el-göz kalibrasyonu (Hand-Eye Calibration) olarak bilinir. Gangal ve arkadaşlarının çalışmasında, OpenCV kütüphanesi kullanılarak bir nesnenin farklı noktalardan görünen konumlarının, perspektif dönüşüm algoritmaları yardımıyla metrik koordinatlara dönüştürüldüğü bir tarama uygulaması anlatılmıştır (Gangal ve ark., 2020). Bu tür kütüphane fonksiyonları, kamera görüntüsündeki bir düzlemin perspektif projeksiyonunu tersine çevirerek (homografi ile) gerçek düzlemdeki konumu bulma olanağı tanır. Kamera ile düzlem arasındaki homografi dönüşümü, ölçü birimi dönüşümünde kullanılabilecek ölçek bilgisi de içerir. Örneğin, kameraya belirli bir açıyla duran düzlemsel bir yüzey (örneğin makine tablası), perspektif dönüşüm ile kuşbakışı görünümüne dönüştürüldüğünde, görüntü piksel koordinatları belirli bir ölçek katsayısı ile gerçek uzunluklara karşılık gelecektir. Bu ölçek, önceden bilinen bir referans uzunluk üzerinden kalibre edilebilir.

Bir nesnenin görüntü koordinatlarından gerçek konumunu bulmak için çoğunlukla referans geometriler kullanılır. Örneğin, endüstriyel bir uygulamada nesne üzerine yerleştirilen birkaç adet işaretleyici (markör), kameradan tespit edilerek Perspective-n-Point (PnP) algoritması ile 3B konum hesaplanabilir. Lee ve arkadaşları, üretim hattında konum hatasını düzeltmek amacıyla nesneler üzerine yapay işaretler yerleştirmiş ve kameradan bu işaretlerin görüntü koordinatlarını alarak PnP yöntemiyle nesnenin hatalı konumunu gerçek zamanlı olarak hesaplamıştır (Lee ve ark., 2021). Bu sayede, bir nesnenin bant üzerindeki konumu belirlenen referansa göre kaymışsa, tespit edilen öteleme ve dönme hatası robot kontrolörüne iletilerek düzeltme yapılabilmektedir. PnP tabanlı yöntemler, bilinen 3B noktaların (örneğin işaretleyici merkezleri veya nesnenin CAD modelindeki belirgin noktalar) görüntüdeki izdüşüm noktalarıyla eşleştirilmesini gerektirir. Yeterli sayıda eşleşik nokta bulunduğunda, kamera denklemlerini tersine çözen algoritmalar (DLS, EPnP gibi yöntemler) sayesinde kameranın nesneye göre pozisyonu elde edilir. Bu yaklaşım hem kamera kalibrasyon parametrelerini hem de görüntü işleme ile bulunan 2B özellikleri kullanarak 3B pozisyonu metrik doğrulukla verir.

Kamera-robot sistemlerinde, kameradan hesaplanan metrik konumun robot koordinat sistemine dönüştürülmesi gerekir. Bu amaçla, robot ile kamera arasındaki dönüşümün bulunması (el-göz kalibrasyonu) için çeşitli teknikler geliştirilmiştir. Craig'in robotik kontrol ve kinematik üzerine çalışması, bir kamera ile robot arasındaki pozisyon ilişkisinin nasıl modellenebileceğine dair temel bilgiler sunmaktadır (Craig, 2018). Yakın dönemde, el-göz kalibrasyonundaki belirsizlik ve ölçüm hatalarını da hesaba katan istatistiksel yöntemler ortaya çıkmıştır. Örneğin, Ulrich ve Hillemann, kameranın robot üzerindeki yerinin belirsizliğini probabilistik olarak modele dahil eden ve kalibrasyon doğruluğunu artıran bir yöntem önermiştir (Ulrich ve & Hillemann, 2023). Bu gibi gelişmeler, kamera tabanlı konum belirleme sistemlerinin endüstriyel ortamlarda daha güvenilir hale gelmesini sağlamaktadır. Son olarak, çoklu kameraların birleşik kullanımı da konum doğruluğunu yükseltebilir. İki veya daha fazla kameradan elde edilen ölçümler, ortak bir koordinat sisteminde birleştirilerek robotun veya nesnenin konumu daha hassas olarak hesaplanabilir. Hartley ve Zisserman'ın çoklu görünüm geometrisi alanındaki temel eserinde, birden fazla kameranın birlikte kalibrasyonu ve 3B yeniden yapılanım konuları derinlemesine incelenmiştir (Hartley ve & Zisserman, 2004). Bu teorik çerçeve, pratikte stereo kameralar veya kamera-IMU kombinasyonları gibi sensör füzyonu senaryolarında da uygulanmaktadır.

2.7. Görsel Veriye Dayalı Robot Yönlendirme Uygulamaları

Görsel veriye dayalı robot yönlendirme, robotların kamera veya benzeri algılayıcılarla çevrelerini algılayıp hareket veya iş eylemlerini buna göre ayarlaması anlamına gelir. Endüstride, makine görüşü destekli robot sistemleri esneklik ve hassasiyet kazandırdığı için yaygınlaşmıştır. Shahria ve arkadaşları, endüstriyel ve otonom sistemlerde kullanılan görsel robot uygulamalarının mevcut durumunu kapsamlı bir şekilde incelemiş, bu sistemlerin bileşenlerini ve karşılaşılan zorlukları özetlemiştir (Gonzalez ve & Woods, 2018). Görüntü işlemeye dayalı robot kılavuzlama teknikleri, sabit robotik kollar için de mobil robotlar için de uygulanmaktadır. Örneğin, montaj hatlarında bir robot kolun kamerayla bir nesneyi tanıyıp kavraması tipik bir senaryodur. Robotik kavrama konusunda yapılan bir derlemede, nesnenin görüntüden tespit edilmesi, pozunun hesaplanması ve uygun kavrama noktasının belirlenmesi ardışık adımlar olarak ele alınmış ve makine öğrenmesi ile bu adımların her birinde elde edilen gelişmeler özetlenmiştir (Du ve ark., 2021). Bu sayede, önceden pozisyonu sabit olmayan parçaların bile robotlar tarafından otomatik olarak yakalanması ve istenen yere yerleştirilmesi mümkün hale gelmiştir. Nitekim, elleçleme (pick-and-place) uygulamalarında başarılı örnekler mevcuttur; örneğin, Munoz'un tezinde kamera görüntülerine dayanarak nesnelerin stabil kavrama noktalarını tespit etmek için derin öğrenme stratejileri geliştirilmiş ve farklı şekillerdeki nesneler için güvenli tutuş pozisyonları otomatik olarak saptanmıştır (Muñoz, 2019)

. Görsel bilginin robot kontrol döngüsüne gerçek zamanlı entegre edildiği durumlar, görsel servo (visual servoing) olarak adlandırılır. Bu alanda, kamera görüntüsündeki hataya göre robot hareketini sürekli düzeltmeye yarayan geribeslemeli kontrol teknikleri geliştirilmiştir. Özellikle esnek imalat ve montaj sistemlerinde, görüntü tabanlı rehberlik sayesinde robot, konumdaki küçük hataları telafi edebilir ve insan operatöre gerek kalmadan hassas görevler gerçekleştirebilir.

Görsel robot yönlendirme, mobil robotlar ve otonom araçlar için de büyük öneme sahiptir. Örneğin, bir insansız kara aracının sırf kamera görüntülerine dayanarak gezinmesi, yol şeritlerini ve engelleri algılaması mümkün olabilmektedir. Sun ve arkadaşları, görsel rehberlik sistemlerini yapay öğrenme teknikleriyle birleştirerek robotların dinamik ortamlarda genel geçer bir şekilde hareket edebilmesine dair bir derleme sunmuştur (Singh ve ark., 2022). Bu çalışmada, otonom tarım araçlarından insansız hava araçlarına kadar farklı platformlarda görsel navigasyon ve rehberlik yaklaşımları incelenmiştir (Singh ve ark., 2022). Otonom sürüş uygulamalarında da kameralar temel sensörlerden biridir; trafikteki nesnelerin (araç, yaya,

trafik işareti) tanınması ve mesafe tayini için gelişmiş yöntemler kullanılmaktadır. Bu alandaki 3B nesne algılama tekniklerine yönelik kapsamlı bir derleme Arnold ve ekibi tarafından yapılmıştır (Arnold ve ark., 2019). Görsel odometre ve eşzamanlı konumlandırma ve haritalama (SLAM) yöntemleri ise mobil robotların çevreyi tanıyarak kendi konumlarını sürekli güncellemesini sağlar. Campos ve arkadaşları tarafından geliştirilen ORB-SLAM3 sistemi, tek veya çoklu kamera ve atalet birimleri ile harita çıkarma ve konum takibinde yüksek doğruluk elde ederek, görsel veriye dayalı robot navigasyonunda önemli bir adım atmıştır (Campos ve ark., 2021). Bu sayede, GPS olmayan ortamlarda dahi robotlar kameraları vasıtasıyla harita oluşturup kendi konumlarını hesaplayabilmektedir.

Birleşik görüş ve kontrol problemlerine yönelik literatürde bazı temel kaynaklar da mevcuttur. Özellikle robotik ve görsel geri besleme konularını bir arada ele alan Corke'un kitabı, robot kinematiği, kamera modellemesi ve görsel servo kontrolü konularında hem teorik arkaplan hem de pratik algoritmalar sunmaktadır (Corke, 2017). Görsel veriye dayalı yönlendirme sistemlerinin tasarımında, robot dinamikleri ve kontrol teorisi bilgilerinin, görüntü işleme ve bilgisayarlı görü teknikleriyle entegre edilmesi gerekir. Bu çok disiplinli alanda, Springer Handbook of Robotics gibi kapsamlı kaynaklar da görüntü tabanlı kontrol ve algılamanın prensiplerini anlatmakta ve endüstriyel uygulama örneklerine yer vermektedir (Siciliano ve & Khatib, 2016). Sonuç olarak, görsel bilginin etkin kullanımı sayesinde robotlar, değişen ortamlara uyum sağlayarak nesneleri tanıyabilmekte, hassas konumlama yapabilmekte ve otonom bir şekilde görev icra edebilmektedir (Shahria ve ark., 2022)(Campos ve ark., 2021)(Arnold ve ark., 2019).

Robotlu Polisaj ve Endüstriyel Yüzey İşleme Sistemlerinde Konum Doğrulama Yaklaşımları

Robotlu polisaj, kaynak sonrası taşlama, yüzey parlatma gibi endüstriyel yüzey işleme süreçlerinin otomatikleştirilmesini hedefler. Bu alanda en büyük zorluklardan biri, robotun takımının (polisaj başlığının) iş parçası üzerinde doğru konumda ve uygun basınçta tutulmasını sağlamaktır. Görüntü işleme teknikleri ve bilgisayarlı görü algılayıcıları, polisaj yapılacak parçanın konumunu ve şeklini algılayarak robotun yörüngesini dinamik olarak düzeltmek için kullanılabilir. Son yıllarda robot destekli taşlama ve polisaj üzerine yapılan çalışmalar, görsel geri besleme ve kuvvet kontrolünün birleşik kullanımına odaklanmıştır. Ke ve arkadaşlarının 2022 tarihli derlemesi, robot destekli polisaj sistemlerinin mevcut durumunu ve gelecekteki eğilimlerini kapsamlı biçimde incelemektedir (Ke ve ark., 2022). Bu derlemeye göre, robotlu polisajda konum ve kuvvet doğruluğunu artırmak için bir yandan gelişmiş kontrol algoritmaları

geliştirilirken, diğer yandan görsel algılamadan faydalanarak iş parçasının durumu izlenmektedir (Ke ve ark., 2022).

Polisaj veya zımpara yapılacak parçanın üretim toleransları dahilinde doğru yerde olduğundan emin olmak için, bazı sistemler görsel konum doğrulama modülleri kullanır. Örneğin, Xing ve ark. küçük ölçekli gemi parçalarının otomatik polisajı için derinlik kamerası tabanlı bir algılama algoritması geliştirmiştir. Bu sistemde, lazer tarayıcı benzeri bir kamera ile iş parçasının üç boyutlu noktalarını toplayarak parçanın şekli ve konumu çıkarılmakta, ardından bu bilgi robota aktarılıp polisaj yörüngesi buna göre ayarlanmaktadır (Xing ve ark., 2022). Bu sayede, her parça için elle programlama yapmak yerine, kameranın algıladığı konum bilgisine dayanarak robotun hareketi uyarlanabilmektedir. Özellikle her iş parçasının biçiminin hafif değişiklikler gösterdiği döküm, dövme gibi işlemler sonrası, bu tür derinlik algılamalı çözümler robotlu yüzey işleme prosesine esneklik kazandırmaktadır. Diğer bir yaklaşım, iş parçası üzerinde tanınabilir görsel işaretler kullanarak robotun pozisyon hatasını düzeltmektir. Üretim hattı ortamında yapılan bir çalışmada, belirli referans markörler aracılığıyla parçanın bant üzerindeki konumu sürekli takip edilmiş ve robotun polisaj takımının hedef noktaya hizalanması sağlanmıştır (Lee ve ark., 2021). Görüntü tabanlı bu düzeltme modülü sayesinde, parça konumundaki milimetre düzeyindeki sapmalar dahi tespit edilip robota geri besleme olarak iletilmiş ve polisaj işlemi istenen bölgede yoğunlaşacak şekilde ayarlanmıştır.

Robotlu zımpara/polisaj süreçlerinde akademik literatürde bazı prototip uygulamalar da rapor edilmiştir. Örneğin, Kubáček'in 2024 tarihli yüksek lisans tezinde, rastgele şekilli bir nesneyi otomatik olarak zımparalamak için bir tarama ve yol planlama sistemi geliştirilmiştir (Kubáček, 2024). Bu sistemde bir derinlik kamerası ile iş parçasının nokta bulutu elde edilmekte, ardından bu nokta bulutundan yüzey geometrisine uygun bir zımparalama yolu hesaplanmaktadır. Robot, harekete geçmeden önce operatöre artırılmış gerçeklik gözlüğü ile bu yolun bir projeksiyonunu göstermekte ve onay almaktadır. Devamında, polisaj robotu hesaplanan yolu takip ederken, entegre kuvvet sensörü sayesinde sürekli basınç kontrolü yapılmıştır. Bu örnek uygulama, görsel algılamanın ve ileri kontrol tekniklerinin birleşimiyle karmaşık şekilli yüzeylerin insan müdahalesi olmadan işlenebileceğini ortaya koymaktadır (Kubáček, 2024).

Genel olarak, robotlu yüzey işleme alanında konum doğrulama ihtiyacı hem kaliteyi hem de süreci etkileyen bir faktördür. Görsel izleme sistemleri, iş parçasının başlangıç konumunu doğrulamada, işlem esnasında sapmaları tespit etmede ve işlem sonrasında kalite kontrolünde kullanılabilmektedir. Örneğin, birden fazla geçiş gerektiren polisaj işlemlerinde, ilk geçiş sonrası yüzey görüntüsü analiz edilerek kalan pürüzlü bölgeler tespit edilebilir ve

ikinci geçişin rotası buna göre planlanabilir. Bu amaçla geliştirilen bazı görüntü işleme teknikleri, yüzey dokusundaki değişimleri sayısal olarak değerlendirerek, polisajın homojenliğini ölçmektedir (Ke ve ark., 2022)(Xing ve ark., 2022). Ayrıca, işlemin sonunda kamera ile elde edilen yüzey görsellerini, istenen ideal yüzey ile karşılaştırarak otomatik denetim yapan sistemler de önerilmektedir. Sonuç itibariyle, bilgisayarlı görü teknolojileri, robotlu polisaj ve benzeri yüzey işleme uygulamalarında konum ve işlem doğruluğunu artıran vazgeçilmez bir bileşen haline gelmiştir. Bu alandaki literatür incelendiğinde, görsel algılama ile kuvvet/hareket kontrolünü bir arada değerlendiren pek çok ileri yöntem bulunduğu görülür (Ke ve ark., 2022). Çalışmamız kapsamında, doğrudan konu ile ilgili olmayan medikal endoskopik cerrahide 3B konum çıkarımı, otonom insansız hava araçlarında görsel konumlama gibi alt başlıklara girilmemiştir. Ancak alan literatürü son derece zengindir; örneğin robotik cerrahide X-ışını görüntülerinden kateter pozisyonu takibi (Schell, 2020) veya sürücüsüz araçlar için kamera-LiDAR füzyonu ile haritalama (López ve & Schomer, 2020) gibi konular, görsel konum belirleme tekniklerinin farklı disiplinlerdeki uygulamalarına işaret etmektedir. Bu çalışma, kapsamı gereği yalnızca endüstriyel robotik ve üretim süreçlerine odaklanan konum belirleme/doğrulama literatürünü ele almış, diğer alanlardaki ilgili çalışmalar kapsam dışı bırakılmıştır.

3. MATERYAL VE YÖNTEM

Bu bölümde, geliştirilen bilgisayarlı görü tabanlı robotik yüzey işleme sisteminin donanım ve yazılım bileşenleri, sistemin kurulum ve kalibrasyon adımları, yapay zekâ modelinin geliştirilme süreci ve otonom işlem için kullanılan algoritmik yöntemler ayrıntılı olarak açıklanmaktadır.

3.1. Sistem Mimarisi ve Donanım Bileşenleri

Geliştirilen sistem, bir mekanik hareket platformu ile bu platformu akıllı hale getiren algılama, işleme ve kontrol birimlerinin entegrasyonuna dayanan hibrit bir mimariye sahiptir. Sistemin fiziksel altyapısını oluşturan temel donanım bileşenleri ve bu bileşenlerin rolleri diğer başlıklarda detaylandırılmıştır.

3.1.1. 4 Eksenli CNC Robot ve Kontrol Altyapısı

Çalışmanın temelini, üç doğrusal (X, Y, Z) ve bir açısal (A) eksene sahip, kartezyen koordinat sisteminde çalışan bir CNC (Computer Numerical Control) platformu oluşturmaktadır. Bu yapı, takım ucunun iş parçası üzerinde hem konumsal hem de açısal olarak hassas bir şekilde yönlendirilmesine olanak tanır. X, Z ve A eksenleri tek bir step motor ile sürülürken, Y eksenindeki hareket, mekanik stabiliteyi sağlamak amacıyla tek bir sürücüye paralel bağlanmış iki adet step motor tarafından sağlanmaktadır. Sistemin hareket kontrolü, G-kodu komutlarını yorumlayarak TB6600 model step motor sürücülerine gerekli sinyalleri gönderen, açık kaynaklı GRBL gömülü yazılımı (firmware) (McGinnis, 2019) ile donatılmış STM F407 tabanlı bir kontrol kartı tarafından yönetilmektedir. Bu tür sayısal kontrollü sistemler, endüstriyel otomasyonda yüksek tekrarlanabilirlik ve hassasiyet sunmaları nedeniyle temel bir bileşen olarak kabul edilir (Groover ve ark., 1986). Şekil 3.1'de kontrol kartının fiziksel yapısı gösterilmektedir.

Şekil 3.1 GRBL tabanlı 4 eksen CNC kontrol kartı

3.1.2. Görüntü Algılama Donanımı

S Sistemin görsel algı yeteneği, deney düzeneğinde hareketli olan X ekseni üzerine monte edilmiş, 6 MP çözünürlüğe ve 110° görüş açısına (FOV) sahip endüstriyel bir Hikrobot kameraya dayandırılmıştır. Bu yapısı itibari ile 3K (3072 x 1620) piksel çözünürlüğüne sahip görüntüler üretebilmiştir. Veri aktarımı, Power over Ethernet (PoE) teknolojisi ile tek bir Ethernet kablosu üzerinden sağlanmıştır. Bu bağlantı, sisteme dahil edilen Tenda TEG1105P-4-63W model bir PoE switch aracılığıyla gerçekleştirilmiştir. Bu yapı, sistemin güvenilirliğini ve kurulum esnekliğini artırmıştır (Nof, 1999). Tanımlanan kamera ve ilgili donanımların, hareketli X ekseni üzerine entegre edildiği deney düzeneğinin genel görünümü Şekil 3.2'de sunulmuştur.

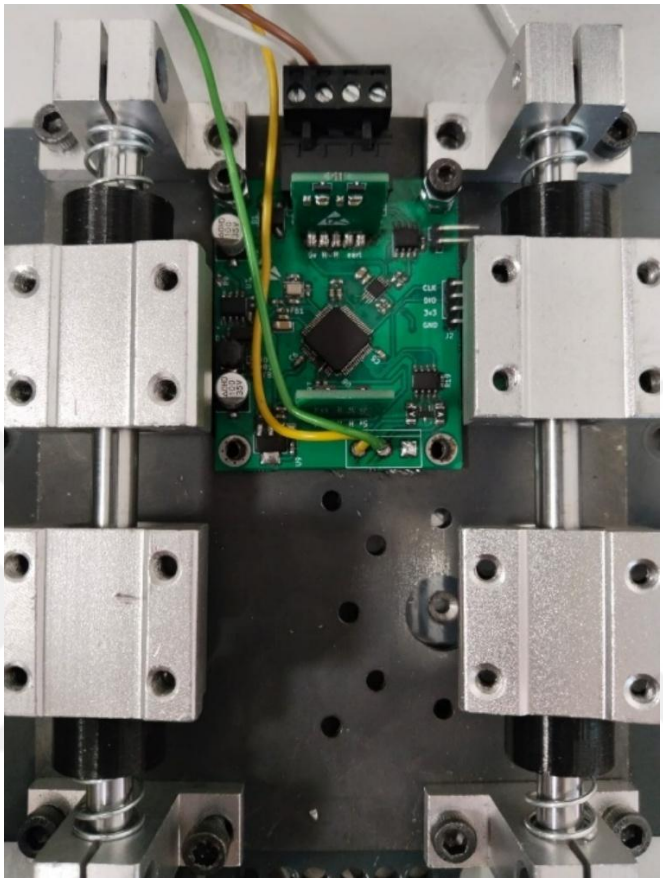
Şekil 3.2 Endüstriyel kamera ve X ekseni bağlantı aparatı

3.1.3. Yüzey Temas Algılama Mekanizması

Yüzey işleme kalitesini doğrudan etkileyen en kritik faktörlerden biri, taşlama ucu ile iş parçası arasında uygulanan baskı kuvvetinin işlem boyunca sabit tutulmasıdır. Özellikle homojen olmayan yüzey topolojilerinde bu kuvveti korumak ve yüzey eğimlerini hassas bir şekilde takip edebilmek amacıyla, bu tez kapsamında özgün bir pasif-uyumlu (passive compliant) kuvvet geri besleme mekanizması tasarlanmış ve imal edilmiştir. Bu sistem, mekanik bir yaylı yapı ile temassız bir manyetik algılama sisteminin bütünleşik çalışması prensibine dayandırılmıştır.

Mekanizmanın mekanik altyapısı, birbirine geçmiş iki ana bileşenden oluşturulmuştur: CNC sisteminin ana Z eksenine sabitlenen bir dış taşıyıcı ve bu taşıyıcının içinde lineer rulmanlar aracılığıyla dikey olarak serbestçe hareket edebilen bir iç taşıyıcı. Taşlama aparatı,

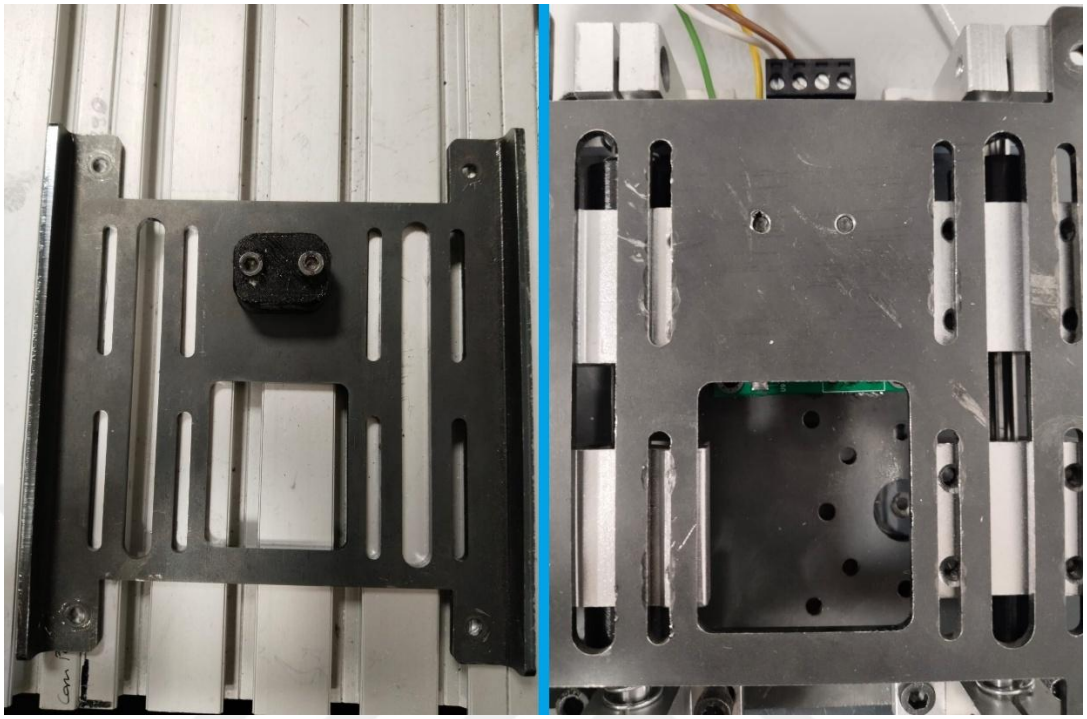
bu hareketli iç taşıyıcıya monte edilmiştir. İki taşıyıcı arasına yerleştirilen yaylar, taşlama ucuna sürekli olarak önceden belirlenmiş bir ön yükleme kuvveti uygulamıştır. Bu tasarım sayesinde, CNC'nin ana Z ekseni sabitken bile taşlama ucu, yüzeydeki mikroskobik engebelere göre yukarı ve aşağı salınım yaparak baskı kuvvetindeki ani değişimleri sönümleyebilmiştir. Mekanizmayı oluşturan bu temel taşıyıcı bileşenlerin tasarımı Şekil 3.3'te sunulmuştur.



Şekil 3.3 Hall Effect sensörünü taşıyan ve yaylı mekanizmanın montajlandığı yapı

Sistemin elektronik algılama katmanı, bu mekanik salınım hareketini yüksek hassasiyetle ölçmek üzere tasarlanmıştır. Bunun için, hareketli iç taşıyıcıya küçük bir neodim mıknatıs entegre edilmiştir. Sabit dış taşıyıcı üzerine ise, bu mıknatısın hareketini manyetik alan değişiminden faydalanarak algılayan bir Hall etki sensörü konumlandırılmıştır. Taşlama ucu yüzeydeki bir engebeye denk geldiğinde, hareketli taşıyıcı dikey olarak yer değiştirmiş ve bu hareket, mıknatıs ile Hall sensörü arasındaki mesafeyi değiştirerek sensörün çıkış voltajında orantılı bir değişikliğe neden olmuştur. Bu analog voltaj sinyali, anlık yay sıkışma miktarının ve dolayısıyla Hooke Yasası gereği anlık baskı kuvvetinin bir göstergesi olarak kullanılmıştır. Elde edilen bu kuvvet verisi, PID kontrol döngüsüne bir geri besleme olarak gönderilmiş ve CNC'nin Z ekseninin anlık olarak mikro düzeltmeler yapması sağlanarak, tüm yüzey boyunca sabit bir baskı kuvvetinin korunması başarılmıştır. Şekil 3.4'te manyetik tetikleyici ve taşlama

aparatını taşıyan hareketli parça yer almaktadır. Bu parçalar özel olarak CNC'nin Z eksenine göre tasarlanmış ve ürettirilmiştir.



Şekil 3.4 Manyetik tetikleyici ve taşlama aparatını taşıyan hareketli parça

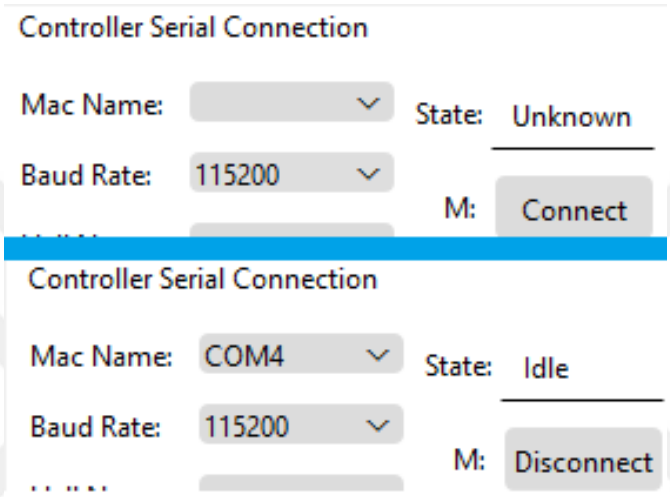
Aparat çalışma yüzeyine temas ettiğinde, yay sisteminin sıkışmasıyla mıknatıs, sabit konumda duran radyometrik Hall effect sensörlerine yaklaşır. Mıknatıs ile sensörler arasındaki mesafenin değişimi, manyetik alanda eksponansiyel bir farklılık yaratarak sensörlerin elektriksel çıktısını değiştirir. Sensörlerden gelen bu analog sinyal, yüksek hassasiyetli bir 16-bit ADC (Analog-to-Digital Converter) ile sayısallaştırılır. Elde edilen bu dijital veri, gürültü ve anlık dalgalanmaları en aza indirmek amacıyla sensör kartı üzerinde bulunan, ARM tabanlı STMF105 işlemcisi tarafından bir Kalman filtresinden geçirilir (Kalman, 1960). Filtrelenmiş ve kararlı hale getirilmiş bu nihai veri, sistemin PID kontrol döngüsü için temel geri besleme sinyali olarak kullanılır ve takımın yüzeye uyguladığı basıncın dolaylı olarak ölçülmesini sağlar.

3.1.4. Sistem Bileşenlerinin Haberleşme Mimarisi

Sistemin merkezi kontrol birimi olan bilgisayar ile çevre donanımlar arasındaki veri akışı, her bir bileşenin özelliğine uygun olarak farklı haberleşme kanalları üzerinden yönetilmiştir. Bu merkezi mimari, tüm donanım bileşenlerinin tek bir yazılım üzerinden senkronize bir şekilde yönetilmesine olanak tanır.

3.1.4.1 CNC Kontrolcüsü-Bilgisayar

Haberleşme, USB üzerinden sanal bir seri port (COM) aracılığıyla gerçekleştirilir. Geliştirilen ana kontrol yazılımı, Qt kütüphanesinin QSerialPort sınıfını kullanarak G-kodu komutlarını bu kanal üzerinden GRBL kontrolcüsüne gönderir ve kontrolcüden gelen anlık durum ve pozisyon raporlarını alır. Şekil 3.5'te bu bağlantının yönetildiği kontrol arayüzü, Şekil 3.6'da Qt IDE'sinin sağladığı QSerialPort kütüphanesi ile oluşturulan kompakt C++ fonksiyon bloğu gösterilmektedir.



Şekil 3.5 CNC kontrolcüsü bağlantı arayüzü

```
void MainWindow::on_btn_ConnectMac_clicked()
{
    if(!core->MacConnected)
    {
        if(core->Port_Connect(core->serialMac,ui->cbox_MacPortN->currentText(),ui->cbox_MacPortBR->currentIndex(),QIODeviceBase::ReadWrite))
        {
            core->MacConnected=true;
            connect(core->serialMac, &QSerialPort::readyRead,this,&MainWindow::grblReading);
            grblStatusThread->start();
            ui->btn_ConnectMac->setText("Disconnect");
        }
        else
            core->writeToConsole("Machine connection failed!");
    }
    else
    {
        core->MacConnected=false;
        core->Port_Disconnect(core->serialMac);
        grblStatusThread->Stop();
        QThread::msleep(110);
        core->serialMac->close();
        ui->btn_ConnectMac->setText("Connect");
    }
}
```

Şekil 3.6 Kontrolcü bağlantısını sağlayan ve seri port bağlantısını yöneten buton fonksiyonu

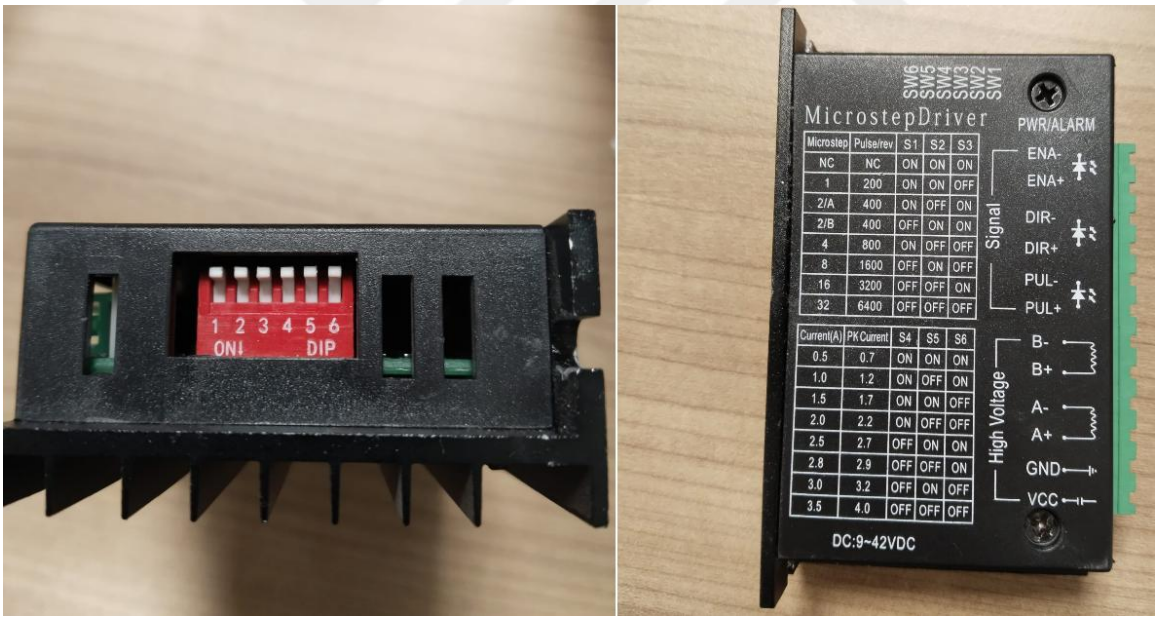
3.1.4.2 GRBL ile Motor Kontrolü

GRBL firmware'i, PC'den gelen metin tabanlı G-kodu komutlarını aldığında bir hareket planlayıcısı (motion planner) olarak görev yapar. Hedef konuma ve hıza göre ivmelenme ve yavaşlama profillerini de içeren bir yörünge hesaplar. Ardından, bu yörüngeyi her bir eksenin step motor sürücüsüne yönelik düşük seviyeli elektriksel sinyallere dönüştürür. Bu sinyaller

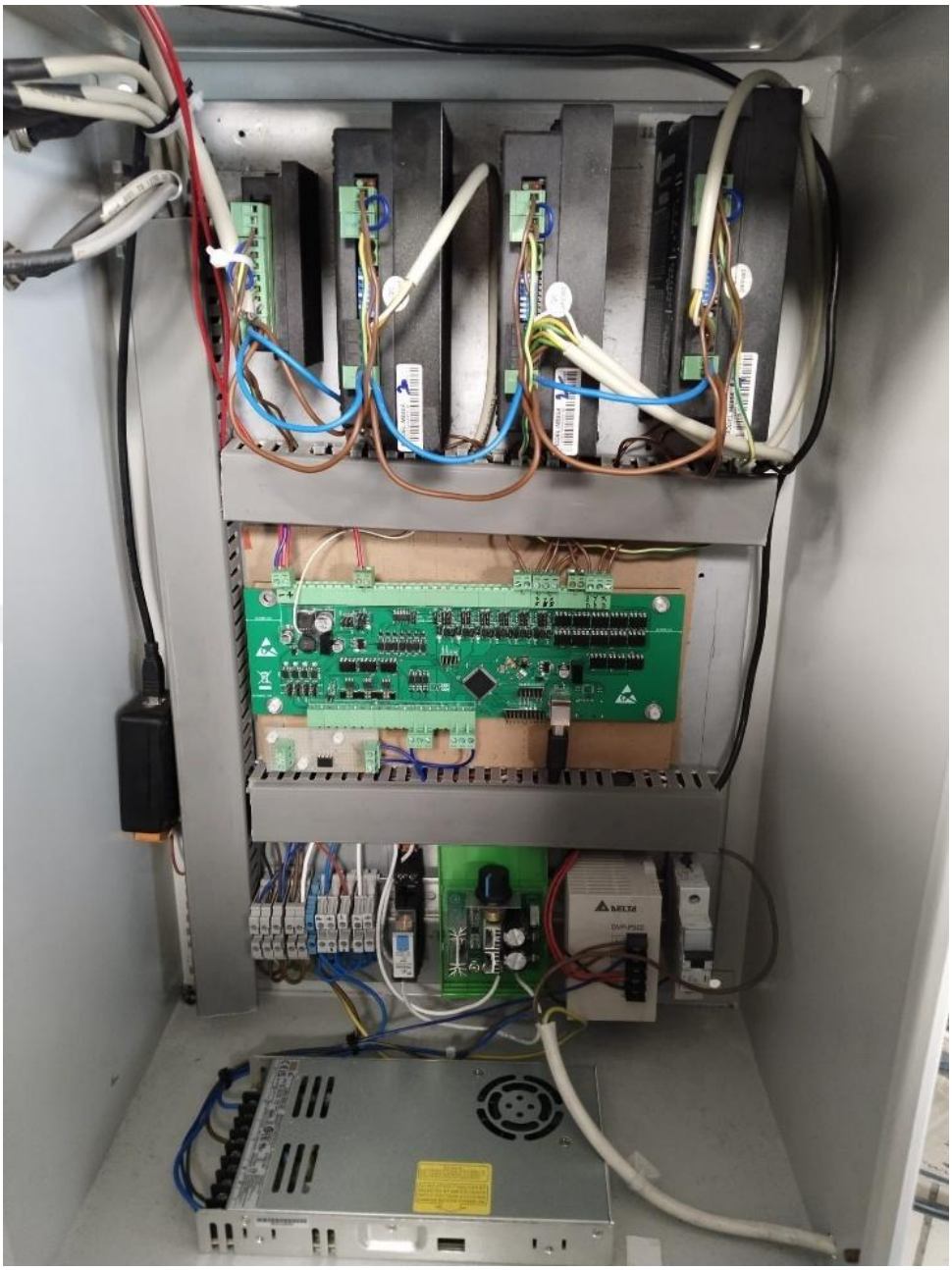
temel olarak iki türdür: Motorun ne kadar döneceğini belirleyen Adım (Step) sinyali ve hangi yöne döneceğini belirleyen Yön (Direction) sinyali. Bu mantıksal sinyaller, TB6600 step sürücüler tarafından güçlendirilerek motor sargılarını enerjilendiren ve fiziksel hareketi oluşturan yüksek akımlı sinyallere çevrilir. Şekil 3.7'de TB6600 step sürücüsü gösterilmektedir.

Tüm sürecin idaresi, kontrol kartı, sürücüler, güç kaynağı ve sigortaların bir araya getirildiği bir elektrik panosu üzerinden sağlanmıştır. Pano içerisinde motor kabloları, sürücüler üzerinden geçirilerek kontrolcü kartının ilgili pinlerine bağlanmıştır. Güç beslemesi ise tüm sürücülere paralel olarak, klemensler aracılığıyla dağıtılmıştır. Klemensler de ana sigortadan geçirilerek ana güç kaynağına bağlanmıştır.

Motorlardaki değişken akım değişimlerinin kontrol kartı üzerinde bozucu bir etki yarattığının gözlemlenmesi üzerine, kontrol kartının, sürücülerin bağlı olduğu güç kaynağından bağımsız olarak harici bir güç kaynağı ile beslenmesi sağlanmıştır. Şekil 3.8'de, tüm bağlantıları içeren ve sürecin idaresinin yapıldığı sistem panosu gösterilmiştir.



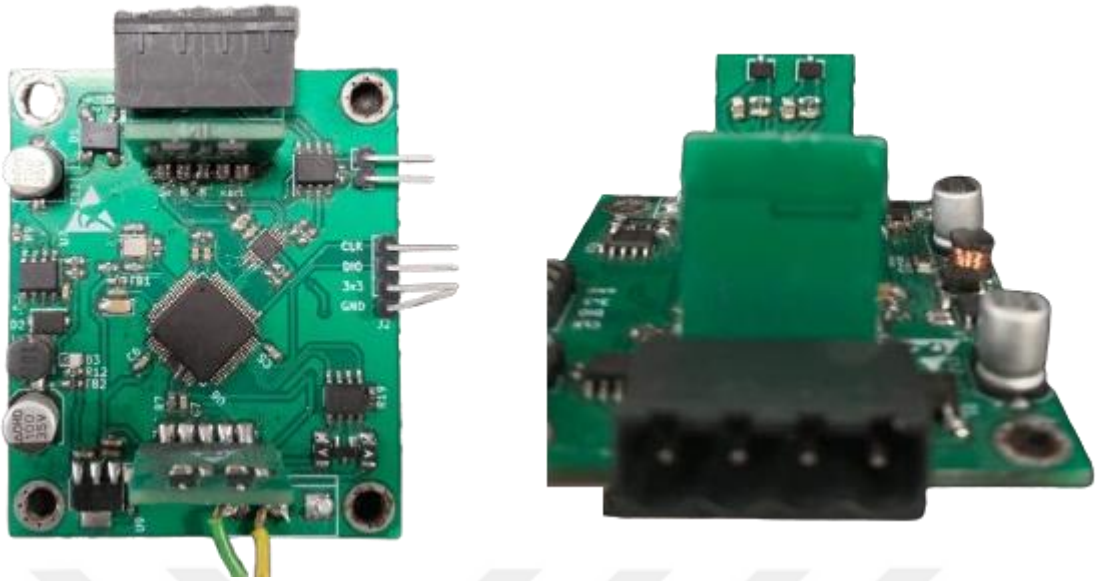
Şekil 3.7 TB6600 step motor sürücüsü ve pin çıkışları



Şekil 3.8 CNC kontrol panosu; kontrolcü kartı ve sürücülerin bağlantıları

3.1.4.1 Hall Effect Sensörü-PC

Sensörden gelen anlık manyetik alan verisi, UART-RS485 dönüştürücü aracılığıyla yine seri haberleşme üzerinden ana bilgisayara aktarılır. Sürekli akan bu veri, PID kontrol döngüsüne girdi olarak beslenir. Şekil 3.9'da radyometrik Hall Effect sensörü ve sensör verilerini filtreleyerek çıkışa gönderen kontrol entegresinin görüntüsü yer almaktadır. Şekil 3.10'da ise bu veriyi RS485'ten UART iletişim protokolüne dönüştüren, dönüştürücü entegresi yer almaktadır.



Şekil 3.9 Hall Effect Sensörü ve sensör verilerini filtreleyen işlemci entegresi

Şekil 3.10 RS485-UART dönüştürücü devresi

3.1.4.2 Endüstriyel Kamera-PC

Yüksek çözünürlüklü görüntü verisinin aktarımı, PoE switch üzerinden Ethernet tabanlı olarak yapılmıştır. Veri akışı, kameraya ait özel SDK (Software Development Kit) fonksiyonları aracılığıyla yönetilerek anlık görüntülerin ana uygulamaya iletilmesi sağlanmıştır. Şekil 3.11'de kamera beslemesini yapan ve PoE portlarından gelen verileri Ethernet çıkışına yönlendiren Tenda PoE çoklayıcısının görseli yer almaktadır.

Şekil 3.11 Kamera ile bilgisayar arasındaki haberleşme hattını bağlayan PoE çoklayıcı

3.2. Yazılım Altyapısı ve Geliştirme Ortamı

Bu tez çalışması, donanım bileşenlerini uyum içinde çalıştıran, gerçek zamanlı veri işleyen ve akıllı kararlar alan bütünleşik bir yazılım mimarisine dayanmaktadır. Bu mimari, farklı görevler için en uygun teknolojilerin seçildiği hibrit bir yaklaşımla tasarlanmıştır.

Sistemin ana kontrolü, kullanıcı arayüzü ve donanım haberleşmesi gibi gerçek zamanlılık gerektiren görevler C++/Qt çatısı altında geliştirilmiş; yapay zekâ modelinin eğitimi ve çıkarım işlemleri için ise Python programlama dilinden yararlanılmıştır. Burada Python dilinin seçimindeki esas neden, bu dilin yoğun hesaplama gerektiren işlemlerde sunduğu gelişmiş kütüphane desteği ile hem kolay entegrasyon hem de optimize kullanım özelliği sağlamasından kaynaklanmıştır.

3.2.1. C++/Qt Tabanlı Ana Kontrol Uygulaması

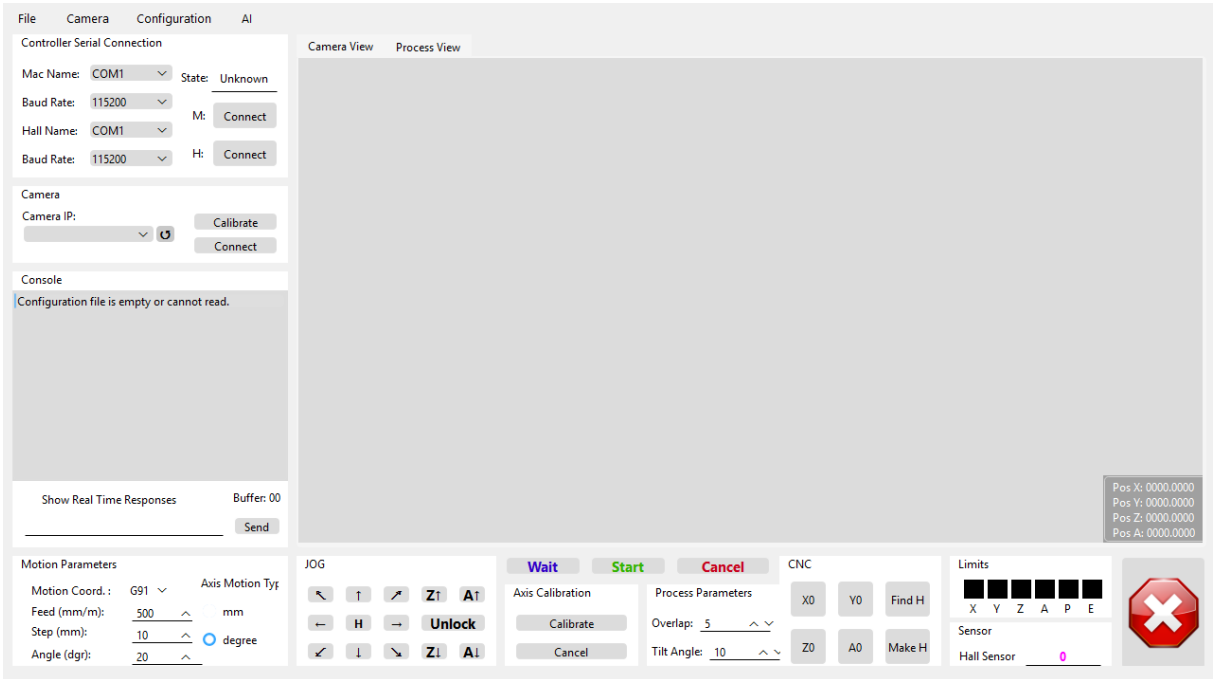
Sistemin merkezi sinir sistemini, C++ programlama dili ve Qt 6 uygulama çatısı kullanılarak geliştirilen bir masaüstü uygulaması oluşturmaktadır. Qt'nin sinyal-slot mekanizmasına dayalı olay güdümlü yapısı, kullanıcı arayüzü ile donanım kontrolü gibi eş zamansız işlemlerin etkin bir şekilde yönetilmesine olanak tanır. Uygulamanın mimarisi, kararlılığı ve yanıt verebilirliği en üst düzeye çıkarmak için şu temel prensipler üzerine kurulmuştur:

- *Çoklu İş Parçacığı (Multi-threading) Mimarisi:* Kullanıcı arayüzünün, donanım yoklama veya görüntü işleme gibi uzun süren işlemler sırasında donmasını engellemek

amacıyla QThread tabanlı bir yapı benimsenmiştir. Kamera görüntülerinin sürekli alınması, GRBL kontrolcüsünden anlık pozisyon verilerinin çekilmesi, ana yüzey işleme algoritmasının yürütülmesi ve kamera kalibrasyonu gibi tüm kritik görevler, ana arayüz iş parçacığından bağımsız olarak arka planda çalışır.,

- *Merkezi Veri Yönetimi:* Tez çalışması kapsamında geliştirilen sistemdeki farklı pencereler, diyaloglar ve iş parçacıkları arasında veri tutarlılığını sağlamak amacıyla merkezi bir sınıf tasarlanmıştır. Bu sınıf, uygulamanın genel durumunu tutan bir veri havuzu görevi görür. Makinenin anlık pozisyonu, sensör verileri, kamera görüntüleri, uygulama konfigürasyonları ve GRBL parametreleri gibi tüm paylaşılan değişkenler bu sınıf içinde yönetilir. Bu sayede, verinin kendisi kullanıcı arayüzü bileşenlerinden ayrıştırılarak modüler ve yönetimi kolay bir yapı elde edilmiştir.
- *Modüler Arayüz Tasarımı:* Uygulamanın kullanıcı arayüzü, her biri belirli bir göreve odaklanmış QMainWindow ve QWidget tabanlı alt pencerelerden oluşur. Kamera ayarları, uygulama konfigürasyonu, veri etiketleme gibi farklı işlevler, kendi pencereleri içinde kapsüllenerek ana pencerenin karmaşıklığı azaltılmıştır.

Şekil 3.12'de ana kontrol arayüzünün son görüntüsü yer almaktadır. Bu arayüz üzerinden genel kontroller ile süreç takibi yapılmaktadır ve GRBL durum bilgileri denetlenmektedir.



Şekil 3.12 Ana kontrol uygulamasının merkez kullanıcı paneli

3.2.2. Python Tabanlı Yapay Zekâ ve Görüntü İşleme Betikleri

Yapay zekâ modelinin eğitimi ve nesne tespiti gibi yoğun matematiksel işlem gerektiren görevler, bu alanda zengin kütüphane desteği sunan Python dilinde geliştirilmiş iki ana betik tarafından yürütülmüştür. C++ ve Python süreçleri arasındaki haberleşme, paylaşılan bir JSON dosyası üzerinden sağlanmıştır. Bu yöntem, iki farklı programlama dilinin güçlü yönlerini tek bir sistemde birleştirmeyi mümkün kılmıştır:

- *Haberleşme Protokolü:* C++ uygulamasında, eğitimin veya çıkarımın başlaması için gerekli olan hiper parametreler, veri yolları ve komutlar bir JSON dosyasına yazılmış ve her bilgisayar için standartta var olan "Dokümanlar" klasörünün içinde açılan lokal klasörün altında kaydedilmiştir.
- *Süreç Başlatma:* Ardından, QProcess sınıfı kullanılarak ilgili Python betiği ayrı bir süreç olarak başlatılmıştır.
- *İşlem ve Geri Bildirim:* Python betiği, başlangıçta bu JSON dosyasını okuyarak görevini (eğitim veya tespit) yerine getirmiş ve süreç boyunca ilerleme durumunu (örn. epoch sayısı) veya işlem sonucunu (örn. tespit edilen poligon koordinatları) yine aynı JSON dosyasına yazarak güncellemiştir.
- *Sonuçların Alınması:* C++ uygulaması, bu JSON dosyasını periyodik olarak okuyarak Python sürecinin durumunu izlemiş ve sonuçları alarak kullanıcı arayüzünü güncellemiş veya bir sonraki adıma geçmiştir.

Bu dosya tabanlı haberleşme mimarisi, iki süreç arasında esnek bir entegrasyon sağlamıştır. Şekil 3.12'de Qt üzerinde yapılan, 3.13'de ise Python tarafında yapılan dosya denetleme blokları yer almaktadır.

```
while (true)
{
    if (jsonFile.open(QIODevice::ReadOnly))
    {
        QByteArray jsonData = jsonFile.readAll();
        jsonFile.close();

        QJsonDocument doc = QJsonDocument::fromJson(jsonData);
        if (!doc.isObject()) {...}

        QJsonObject jsonObj = doc.object();
        if(jsonObj.contains("error")) {...}
        if (jsonObj.contains("progress")) {...}
    }
    QThread::sleep(250);
}
```

Şekil 3.12 Eğitim prosesinin JSON tabanlı kontrolünü gerçekleştiren C++ bloğu

```python
def get_documents_path():
    return str(Path.home() / "Documents")

# JSON dosyasının tam yolu
json_path = os.path.join(get_documents_path(), "AIPolishing", "parameters.json")

def update_json_status(key, value):
    """JSON dosyasına bilgi yaz."""
    try:
        with open(json_path, "r+") as f:
            config = json.load(f)
            config[key] = value
            f.seek(0)
            json.dump(config, f, indent=4)
            f.truncate()
    except (OSError, IOError) as e:
        print(f"JSON dosyası yazılamıyor: {e}")
```

Şekil 3.13 Eğitim prosesinin JSON tabanlı kontrolünü gerçekleştiren Python bloğu

3.2.3. Kullanılan Kütüphaneler ve Harici Araçlar

Bu tez çalışmasının geliştirilmesinde, belirli işlevleri yerine getirmek ve geliştirme sürecini hızlandırmak amacıyla alanında standartlaşmış çeşitli harici kütüphanelerden ve araçlardan faydalanılmıştır:

OpenCV (Open Source Computer Vision Library): Bilgisayarlı görü ve görüntü işleme algoritmaları için temel kütüphane olarak kullanılmıştır (Bradski, 2000). Görüntü formatları arası dönüşümler, kamera kalibrasyonu, perspektif dönüşüm ve diğer temel görüntü manipülasyonları gibi görevler OpenCV fonksiyonları ile gerçekleştirilmiştir.

Hikvision MVS SDK: Sistemin kullandığı endüstriyel Hikrobot kameranın donanım seviyesinde kontrolü için üretici tarafından sağlanan yazılım geliştirme kitidir (SDK). Kameraya bağlanma, yapılandırma (kazanç, pozlama süresi vb.) ve anlık görüntü çerçevelerini (frame) yakalama işlemleri, MVCameraThread sınıfı içinde bu SDK'nın fonksiyonları aracılığıyla yönetilmiştir.

Ultralytics YOLOv8: Kaynak noktalarının tespiti için kullanılan derin öğrenme modelinin eğitimi ve çıkarım işlemleri, YOLOv8 mimarisini ve onun yüksek seviyeli Python API'sini sağlayan Ultralytics çatısı kullanılarak yapılmıştır (Jocher ve ark., 2023). YOLOv8 mimarisinin bu tez çalışması için tercih edilmesindeki temel etkenler; yüksek çıkarım hızı (FPS), isabetlilik (mAP) ve donanım gereksinimleri arasında sunduğu optimize dengedir. Daha yeni ve büyük modeller potansiyel olarak daha yüksek isabetlilik sunabilse de bu tezdeki gibi gerçek zamanlı CNC kontrolü gerektiren uygulamalarda çıkarım hızı kritik bir öneme sahiptir. Bu bağlamda YOLOv8, mevcut donanım üzerinde yeterli performansı sağlarken, Ultralytics çatısının

sunduğu kullanım kolaylığı sayesinde hızlı prototipleme ve model iterasyonlarına olanak tanıması nedeniyle en uygun seçenek olarak değerlendirilmiştir. Bu çatı, modelin hızlı bir şekilde eğitilmesini ve doğrulanmasını sağlamıştır.

3.3. Sistemin Kurulumu ve Hazırlık Aşamaları

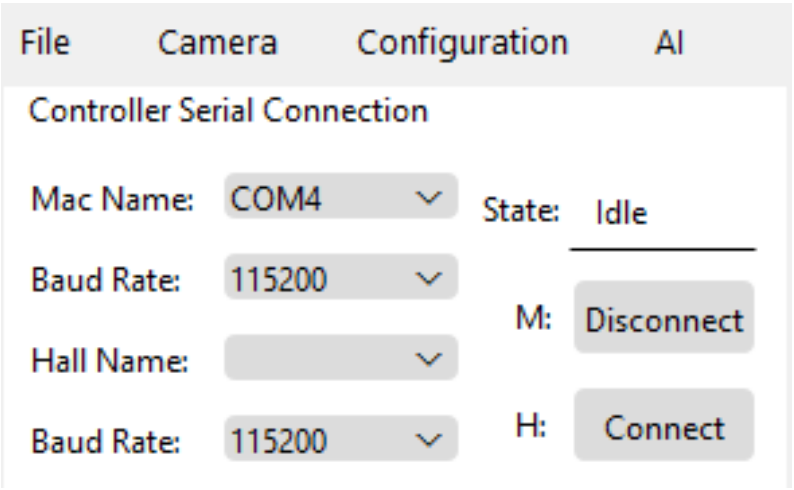
Geliştirilen sistemin otonom olarak görevini icra edebilmesi, bir dizi ardışık kurulum ve hazırlık aşamasının doğru bir şekilde tamamlanmasına bağlıdır. Bu bölüm, sistemin fiziksel donanımının enerjilendirilmesinden, tüm bileşenlerin haberleşme kanallarının yapılandırılmasına ve makinenin otonom çalışmaya hazır hale getirilmesi için gerekli olan manuel operatör işlemlerine kadar olan metodolojiyi ayrıntılı olarak ele almaktadır. Bu aşamalar, Bölüm 3.1 ve 3.2'de tanıtılan donanım ve yazılım altyapısı arasında bir köprü kurarak, sistemin kararlı ve güvenilir bir başlangıç durumuna getirilmesini hedefler.

3.3.1. Donanım Altyapısının Kurulumu ve Haberleşme Kanallarının Yapılandırılması

Otonom işlem öncesindeki ilk ve en temel adım, merkezi kontrol birimi ile tüm çevre donanımlar (CNC kontrolcü, Hall effect sensörü) arasındaki haberleşme kanallarının kurulması olmuştur. Bu süreç, geliştirilen C++/Qt tabanlı ana kontrol uygulaması üzerinden operatör tarafından yönetilmesi ve sistemin entegre bir bütün olarak çalışabilmesi adına kritik öneme sahiptir.

Süreç, yazılımın başlatılmasıyla birlikte, bilgisayara bağlı olan tüm seri (COM) portların otomatik olarak taranmasıyla başlamıştır. Bu tarama işlemi, arka planda çalışan bir iş parçacığı tarafından periyodik olarak tekrarlanmıştır. Bu dinamik tarama mekanizmasının amacı, operatörün manuel olarak port ismi girmesine gerek kalmadan, sisteme yeni bir cihaz takıldığında veya çıkarıldığında arayüzdeki port listelerinin otomatik olarak güncellenmesini sağlamaktır. Bu yaklaşım, özellikle endüstriyel ortamlarda donanım değişikliklerine hızlı adaptasyon ve kullanım kolaylığı sunar (Bailey ve & Wright, 2003).

Operatörün, Şekil 3.5’te arayüzde bulunan açılır menüden CNC kontrolcüsü ve Şekil 3.14’de yer alan, aynı grup içerisindeki menüden de Hall Effect sensörü için doğru COM portlarını ve baud hızlarını seçerek bağlantı işlemini başlatması sağlanmıştır.



Şekil 3.14 CNC kontrolcüsü ve Hall Effect sensörü bağlantı arayüzü

3.3.1.1 CNC Kontrolcüsü Bağlantısı

"Bağlan" butonuna tıklandığında, uygulama belirtilen parametrelerle GRBL kontrolcüsüne bir seri bağlantı açmaya çalışır. Bağlantı başarılı olduğunda, G-kodu gönderme ve alma kanalı aktif hale gelir. Eş zamanlı olarak, "GrblStatusThread" adlı iş parçacığı başlatılır. Bu iş parçacığı, periyodik olarak GRBL'e durum sorgulama komutu '?' göndererek makinenin anlık durumu ve eksen pozisyonları hakkında gerçek zamanlı veri akışını sağlar.

GRBL tarafından gönderilen ve '<' ile başlayıp '>' ile biten durum raporu, makinenin fiziksel ve yazılımsal durum bilgilerini içerir. Rapordaki ilk alan, makinenin o anki çalışma durumunu belirtir; örneğin, **Idle** (yeni bir komut için beklemede), **Run** (bir G-kodu programı yürütüyor), **Hold** (duraklatıldı) veya **Alarm** (bir hata oluştu) gibi. Bu durum bilgisine ek olarak rapor; **MPos** (Makine Pozisyonu), **WPos** (İş Parçası Pozisyonu), **WCO** (İş Koordinat Ofseti), **FS** (Anlık ilerleme ve spindle hızları), **Bf** (Komut arabellek durumu) ve **Pn** (Limit anahtarları gibi pinlerin anlık durumu) gibi kritik verileri de barındırır. Bu veriler ana ekranın ilgili yerlerinde, QLabel tabanlı tasarımlar şeklinde, şekil veya yazı formatında operatöre gösterilir. Şekil 3.15'te bu bilgileri arayüz üzerine işleyen ve kullanıcıyı bilgilendiren kod bloğu yer almaktadır.

```
void MainWindow::onGrblStatusUpdated()
{
    ui->txt_CNCState->setText(core->grblStatus.state);
    ui->lbl_Buffer->setText(QString("Buffer: %1").arg(core->grblStatus.rxBuffer));
    ui->lbl_pos_X->setText(QString("Pos X: %1").arg(core->grblStatus.Pos.x()));
    ui->lbl_pos_Y->setText(QString("Pos Y: %1").arg(core->grblStatus.Pos.y()));
    ui->lbl_posZ->setText(QString("Pos Z: %1").arg(core->grblStatus.Pos.z()));
    ui->lbl_posA->setText(QString("Pos A: %1").arg(core->grblStatus.Pos.w()));
}
```

Şekil 3.15 Makine durum, buffer ve pozisyon bilgisini arayüze işleyen kod bloğu

3.3.1.2 Sensör Bağlantısı

Benzer şekilde, Hall Effect sensörünün bağlı olduğu USB-RS485 dönüştürücü için de ayrı bir seri bağlantı kurulur. Bu bağlantı, yüzey takibi için hayati olan ham manyetik alan verisinin sürekli olarak ana uygulamaya akmasını ve PID kontrol döngüsüne beslenmesini garanti eder.

Uygulama, her iki haberleşme kanalının da başarılı bir şekilde kurulduğunu teyit etmeden, manuel hareket veya otonom işlem gibi ileri seviye fonksiyonların kullanımını kısıtlayarak sistemin kararlı durumda kalmasını sağlar.

Şekil 3.16'da başarıyla gerçekleştirilmiş bir sensör bağlantısı sonrası gelen sensör verisini okuyup, uygulama içerisinde, özellikle asenkron iş parçalarında, kullanılmak üzere çekirdek sınıfına kaydeden kod bloğu yer almaktadır.

```
void MainWindow::ReadSensor(QByteArray read_data)
{
    read_data=read_data.trimmed();
    if(read_data.toFloat()<5)
        core->hallError=true;
    else
        core->hallError=false;
    int ort=read_data.toFloat();
    core->PID.sensor_girdisi=ort;
    if(ort>=core->APPCONFIG.lower && ort<=core->APPCONFIG.upper) {...}
    else {...}
    if(ort<=core->APPCONFIG.lower)
        core->PID.r=core->APPCONFIG.lower;
    else
        core->PID.r=core->APPCONFIG.upper;
    ui->txt_SensorValue->setText(QString::number(ort));
    read_data.clear();
}
```

Şekil 3.16 Sensör verisini ham olarak çekirdek sınıfına kaydeden kod bloğu

3.3.2. GRBL Parametrelerinin Arayüz Üzerinden Yapılandırılması

GRBL standardının davranışı; adım/mm oranları, maksimum hızlar, ivmelenme değerleri ve homing döngüsü gibi temel makine karakteristiklerini tanımlayan, bellekte saklanan bir dizi parametre tarafından yönetilir. Bu parametrelerin geleneksel olarak bir komut satırı arayüzü (CLI) üzerinden tek tek değiştirilmesi hem zaman alıcı hem de hataya açık bir süreçtir. Bu süreci basitleştirmek, verimli hale getirmek ve kullanıcı hatalarını en aza indirmek amacıyla, bu tez kapsamında geliştirilen ana kontrol yazılımına, tüm GRBL parametrelerinin yönetilebildiği bütünleşik bir konfigürasyon arayüzü entegre edilmiştir.

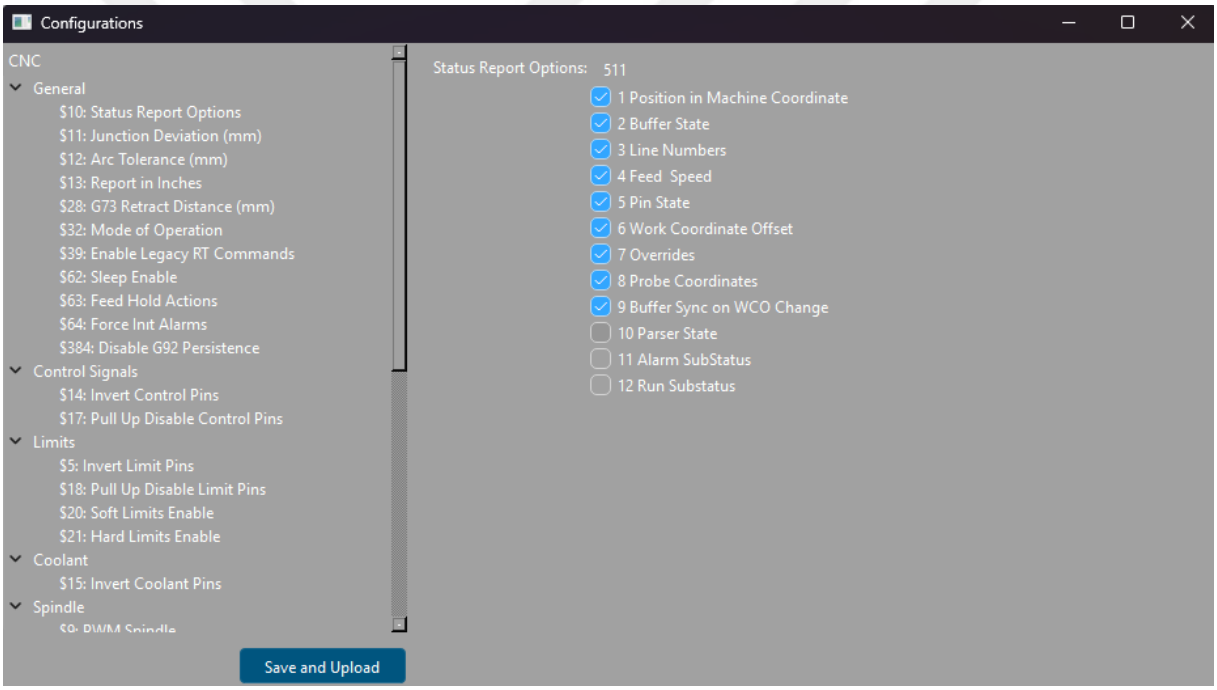
Bu arayüz, operatöre iki panelli bir yapı sunar. Sol panelde, GRBL'e ait 72 farklı parametre, işlevlerine göre (örneğin eksen ayarları, homing ayarları, durum raporu seçenekleri) mantıksal olarak gruplandırılmış bir ağaç yapısı içinde listelenir. Operatör bu listeden bir parametre seçtiğinde, sağ panelde o parametrenin veri tipine özel olarak tasarlanmış düzenleme arayüzü belirir. Bu dinamik yapı, farklı veri türlerinin en anlaşılır şekilde sunulmasını sağlar:

- *Sayısal ve Metinsel Parametreler:* Eksen ivmelenmesi ($120-$123) veya maksimum hareket mesafeleri ($130-$133) gibi doğrudan sayısal değer alan parametreler için standart bir metin kutusu kullanılır. Bu, operatörün değeri doğrudan girmesine olanak tanır.
- *İkili (Boolean) Seçenekler:* "Soft limits" ($20) gibi yalnızca açık veya kapalı (true/false) olabilen ayarlar için, tek bir onay kutusu kullanılır.
- *Bit Maskesi Parametreleri:* Gömülü sistemlerde bellek kullanımını verimli hale getirmek için sıkça kullanılan bir yöntemdir. Bazı GRBL parametreleri, tek bir tamsayı değerinin bitlerini kullanarak birden fazla bağımsız ayarı aynı anda depolar. Örneğin, durum raporu seçeneklerini belirleyen $10 parametresi bu yapıdadır. Geliştirilen arayüz, bu tür parametreleri operatör için soyutlar. Parametrenin sayısal değerini (ör. 511) ikili karşılığına (111111111) çevirir ve her bir bitin temsil ettiği ayarı ("Makine Pozisyonunu Raporla", "Arabellek Durumunu Raporla" vb.) ayrı bir onay kutusu olarak sunar. Operatör bu kutulardan herhangi birini işaretlediğinde veya işaretini kaldırdığında, yazılım arka planda bit düzeyinde "VEYA" (bitwise OR) işlemleriyle yeni tamsayı değerini hesaplar ve ilgili parametreyi günceller. Bu yaklaşım, karmaşık bit maskesi ayarlarının bile son kullanıcı için son derece anlaşılır ve sezgisel bir şekilde yönetilmesini sağlar (Vahid ve & Givargis, 1999).

Yapılandırma işlemi tamamlandığında, operatör "Kaydet ve Yükle" butonunu kullanır. Bu eylem iki aşamalı bir süreci tetikler:

- *Yerel Kayıt:* Tüm parametrelerin güncel durumu, uygulamanın bir sonraki açılışında ayarları hatırlaması için yerel bir metin dosyasına kaydedilir.
- *GRBL'e Yükleme:* Yazılım, 72 parametrenin tamamını döngüye alarak her birini GRBL data formatında ($xx = yy) bir G-kodu komutu olarak seri port üzerinden GRBL kontrolcüsüne gönderir. Bu komutlar, parametrelerin kontrolcünün kalıcı belleğine (EEPROM) yazılmasını ve anında aktif hale gelmesini sağlar.

Şekil 3.17'de konfigürasyon arayüzünün varsayılan arayüzü yer almaktadır. Kullanıcı bir GRBL konfigürasyon parametresi belirlemese de sisteme boş veri gitmemesi için varsayılan konfigürasyon parametreleri oluşturulmuştur.



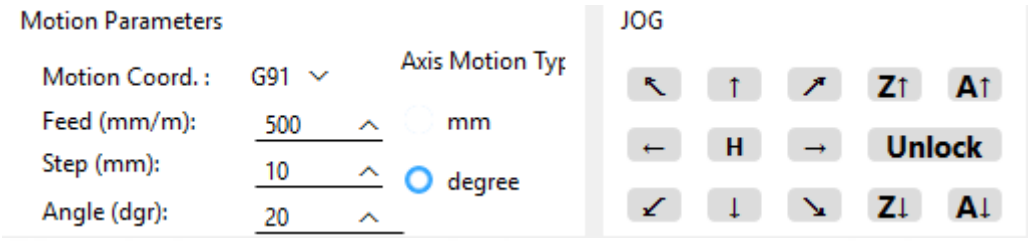
Şekil 3.17 Kullanıcı etkileşimli GRBL konfigürasyon arayüzü

3.3.3. GRBL Kontrolcüsü ile Arayüz Üzerinden Manuel Hareket ve Referanslama

Haberleşme kanalları kurulduktan sonra, makinenin otonom göreve başlamadan önce fiziksel olarak doğru ve bilinen bir konuma getirilmesi gerekmektedir. Bu aşama, arayüzdeki manuel kontrol (jogging) ve referanslama (homing) fonksiyonları ile gerçekleştirilmiştir.

- *Manuel Kontrol (Jogging):* Arayüz üzerinde bulunan yön butonları, operatörün her bir ekseni (X, Y, Z, A) istenen yönde ve hızda hareket ettirmesine olanak tanır. Bu fonksiyonun temel amacı, operatörün iş parçasını tablaya güvenli bir şekilde bağlaması,

kamerayı çalışma alanının üzerine getirmesi ve sistemin mekanik olarak sorunsuz çalıştığını test etmesidir. Arka planda, bu butonlara her basıldığında, uygulama seçili hareket modunda (G91 (artımsal hareket) veya G90 (koordinat tabanlı hareket)) modunda, önceden tanımlanmış bir adım mesafesi ve hız bilgisini içeren G-kodu komutları üretip GRBL'e gönderir. Şekil 3.18'de manuel kontrol arayüzünün ve manuel kontrol parametrelerinin ayarlanabildiği panelin görseli yer almaktadır.



Şekil 3.18 Manuel kontrol arayüzü ve manuel kontrol parametre paneli

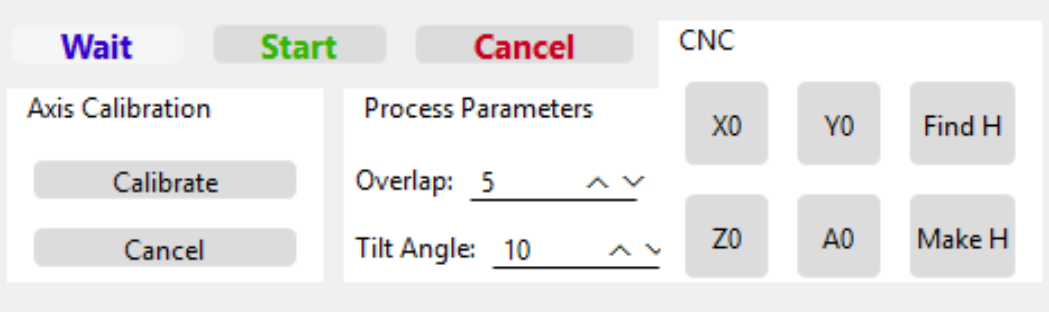
- *Referanslama (Homing ve Sıfırlama):* Hassas ve tekrarlanabilir bir işlem için makine ve iş parçası koordinat sistemlerinin doğru bir şekilde tanımlanması esastır (Smith ve & Chetwynd, 1995). Bu çalışmada iki aşamalı bir referanslama yöntemi benimsenmiştir:
    1. *Makine Referansı (Homing):* Operatör, "Find Home" butonuna tıklayarak GRBL'in dahili referans bulma döngüsünü ($H komutu) tetikler. Bu döngüde, her eksen önceden tanımlanmış bir yönde, kendi limit anahtarına temas edene kadar hareket eder. Tüm eksenler limit anahtarlarını tetiklediğinde, GRBL makinenin mutlak sıfır noktasını (Machine Zero) hassas bir şekilde belirlemiş ve kaydetmiş olur. Bu işlem, makine her kapatılıp açıldığında tutarlı bir başlangıç noktası elde etmek için zorunludur.
    2. *İş Parçası Referansı (Work Zero):* Makine referansı alındıktan sonra, operatör manuel hareket kontrollerini kullanarak takım ucunu iş parçasının başlangıç noktası olarak kabul edilecek bir referans noktasına getirir. Bu noktada, her eksen için "Set 0" butonlarına basılarak hedef ekseni referans alan G92 sıfırlama komutu gönderilir. Bu işlem, mevcut konumu İş Parçası Koordinat Sistemi'nin orijini olarak tanımlar ve makine koordinatlarına olan uzaklığını "WCO" adı altında tutar. Bu parametre doğrudan bir koordinat değeri olmadığı için iş parçasının referans alınan noktaya olan uzaklığı masaüstü uygulaması tarafında Denklem 1.0'da gösterildiği gibi hesaplanmıştır.

$$\overrightarrow{P_{wp}} = \overrightarrow{P_{MPO}} - \overrightarrow{P_{WCO}} \quad (1.1)$$

Bu adımdan sonra gerçekleştirilecek tüm otonom hareketler, makinenin mutlak limitlerine göre değil, doğrudan iş parçasının bu tanımlanmış sıfır noktasına göre yapılmıştır.

Ek olarak, sistemde bir alarm durumu (örneğin limit anahtarına çarpma) oluştuğunda GRBL'in kilitlenmesini çözmek için arayüze bir "Unlock" ($X komutu) fonksiyonu da eklenmiştir. Bu adımların tamamı, otonom sürecin bilinen ve güvenli bir koordinat sisteminde başlamasını garanti altına alır.

Şekil 3.19'da GRBL kontrolleri dahilinde sıfırlama yapan manuel kontrol ekranı yer almaktadır. Ekranın hemen yanındaki panel ise Z eksenini yüzeye teğet hizalamak için kullanılan kalibrasyon ekranıdır. Sensör bağlı ise tetiklendiğinde otomatik çalışan sistem, Z eksenindeki iş takımını güvenli bir yüksekliğe çıkardıktan sonra PID kontrolcü dahilinde kontrollü bir şekilde alçaltır. Yüzeye belli bir kuvvetle temas ettikten sonra Bölüm 3.6.6'da açıklana, boştaki sensör verisi ve temastaki sensör verisinin farkının metrik dönüşümündeki karşılığınca geri çekilir ve yüzeye hizalanmış olur.



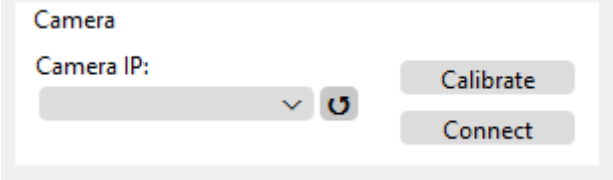
Şekil 3.19 Z ekseni kalibrasyonu paneli ve GRBL referanslama arayüzü

3.4. Görsel Algı Sisteminin Yapılandırılması ve Kalibrasyonu

Sistemin üç boyutlu dünyayı doğru bir şekilde algılayıp metrik olarak ölçebilmesi, ham kamera görüntüsünün bir dizi ardışık işlemden geçirilerek geometrik olarak anlamlı bir veri kaynağına dönüştürülmesine bağlıdır. Bu süreç, kameranın donanım seviyesinde başlatılmasından, lensin neden olduğu optik kusurların matematiksel olarak giderilmesine kadar uzanan temel yapılandırma ve kalibrasyon adımlarını içerir. Bu bölüm, makineye "görme" yeteneği kazandıran ve gördüğü dünyayı doğru yorumlamasını sağlayan bu temel metodolojiyi adım adım açıklamaktadır.

3.4.1. Kamera Bağlantısı ve Görüntü Akışının Başlatılması

Görsel algı sürecinin ilk adımı, fiziksel kamera donanımı ile merkezi kontrol yazılımı arasında kararlı bir iletişim kanalı kurmaktır. Bu işlem, ana arayüz üzerinden operatör tarafından yönetilir ve arka planda, kamera ile ilgili tüm görevlerden sorumlu olan iş parçacığı tarafından yürütülür. Şekil 3.20'de kamera bağlantı arayüzü yer almaktadır.



Şekil 3.20 Endüstriyel kamera sorgulama ve bağlanma arayüzü

Süreç, yazılımın yerel ağ üzerinde GigE Vision protokolü ile uyumlu kameraları taramasıyla başlar. MVCameraThread sınıfı içerisinde bulunan bir fonksiyon, Hikvision SDK'sını kullanarak ağdaki mevcut kameraları listeler ve IP adreslerini arayüzdeki bir açılır menüye doldurur. Bu sayede operatör, manuel bir konfigürasyona gerek duymadan, listeden doğru kamerayı seçebilir. Şekil 3.21'de bu listeleme fonksiyon bloğu yer almaktadır.

```
void MVCameraThread::scanDevices(QComboBox* deviceBox) {
    MV_CC_DEVICE_INFO_LIST stDeviceList;
    memset(&stDeviceList, 0, sizeof(MV_CC_DEVICE_INFO_LIST));

    int nRet = MV_CC_EnumDevices(MV_GIGE_DEVICE | MV_USB_DEVICE, &stDeviceList);
    if (nRet != MV_OK) return;

    deviceBox->clear();
    devices.clear();

    for (unsigned int i = 0; i < stDeviceList.nDeviceNum; ++i) {
        devices.append(*stDeviceList.pDeviceInfo[i]);
        if (stDeviceList.pDeviceInfo[i]->nTLayerType == MV_GIGE_DEVICE) {
            int nIp1 = ((stDeviceList.pDeviceInfo[i]->SpecialInfo.stGigEInfo.nCurrentIp & 0xff000000) >> 24);
            int nIp2 = ((stDeviceList.pDeviceInfo[i]->SpecialInfo.stGigEInfo.nCurrentIp & 0x00ff0000) >> 16);
            int nIp3 = ((stDeviceList.pDeviceInfo[i]->SpecialInfo.stGigEInfo.nCurrentIp & 0x0000ff00) >> 8);
            int nIp4 = (stDeviceList.pDeviceInfo[i]->SpecialInfo.stGigEInfo.nCurrentIp & 0x000000ff);
            QString name = QString("%1.%2.%3.%4").arg(nIp1).arg(nIp2).arg(nIp3).arg(nIp4);
            deviceBox->addItem(name);
        }
    }
}
```

Şekil 3.21 Bilgisayara bağlı GigE Vision uyumlu kameraları tarayan fonksiyon bloğu

Operatör "Bağlan" eylemini tetiklediğinde, yazılım seçilen kamerayla bir oturum başlatmak üzere bir "tanımlayıcı" (handle) oluşturur. SDK aracılığıyla kamera, diğer uygulamaların erişimini engelleyecek şekilde özel erişim modunda açılır. Bu mod özellikle endüstriyel ortamlarda, işlem sırasında başka bir yazılımın kamera ayarlarına müdahale etmesini önlemek için kritik bir adımdır. Bağlantı kurulduktan sonra, kararlı bir görüntü akışı

sağlamak için temel kamera parametreleri yazılım tarafından otomatik olarak ayarlanır. Bunlar arasında, kameranın sürekli görüntü almasını sağlayan tetikleme modunun kapatılması, varsayılan ışık koşullarında net bir görüntü için kazanç (Gain) ve pozlama süresi (Exposure Time) gibi parametrelerin ayarlanması ve Ethernet üzerinden veri paketlerinin kayıpsız iletimi için en uygun GEV paket boyutunun yapılandırılması yer alır. Şekil 3.22'de bu işlemleri yerine getiren kod bloğu yer almaktadır.

```
void MVCameraThread::connectToDevice(int index)
{
    if (index < 0 || index >= devices.size()) return;
    MV_CC_DEVICE_INFO &info = devices[index];
    if (!MV_CC_IsDeviceAccessible(&info, MV_ACCESS_Exclusive)) { ...}
    int nRet = MV_CC_CreateHandle(&handle, &info);
    if (nRet != MV_OK) { ...}
    nRet = MV_CC_OpenDevice(handle);
    if (nRet != MV_OK) { ...}
    if (info.nTLayerType == MV_GIGE_DEVICE) {
        int nPacketSize = MV_CC_GetOptimalPacketSize(handle);
        if (nPacketSize > 0) {
            nRet = MV_CC_SetIntValue(handle, "GevSCPSPacketSize", nPacketSize);
            if (nRet != MV_OK) {
                core->writeToConsole("Warning: Set Packet Size fail!");
            }
        } else {
            core->writeToConsole("Warning: Get Packet Size fail!");
        }
    }
    MV_CC_SetEnumValue(handle, "TriggerMode", 0);
    MV_CC_SetFloatValue(handle, "Gain", 20.0f);
    MV_CC_SetFloatValue(handle, "ExposureTime", 50000.0f);
    MVCC_INTVALUE stParam;
    memset(&stParam, 0, sizeof(MVCC_INTVALUE));
    nRet = MV_CC_GetIntValue(handle, "PayloadSize", &stParam);
    if (nRet != MV_OK || stParam.nCurValue == 0) { ...}
    nPayloadSize = stParam.nCurValue;
    pData = new unsigned char[nPayloadSize];
    if (!pData) { ...}
    nRet = MV_CC_StartGrabbing(handle);
    if (nRet != MV_OK) { ...}
    frameSkipCount = 0;
    connected = true;
    this->start();
}
```

Şekil 3.22 GigE Vision tabanlı cihaz bağlantı bloğu

Tüm bu yapılandırmalar tamamlandıktan sonra, kameranın sürekli görüntü yakalama moduna geçmesi için "MV_CC_StartGrabbing" komutu gönderilir. Bu andan itibaren MVCameraThread iş parçacığı, ana "run()" döngüsü içinde periyodik olarak kameranın arabelleğinden en güncel görüntü çerçevesini çeker. Kameradan gelen ham Bayer formatındaki

görüntü, yine SDK fonksiyonları aracılığıyla standart bir RGB renk uzayına dönüştürülerek hem arayüzde gösterilmeye hem de sonraki adımlarda OpenCV tarafından işlenmeye hazır hale getirilir. Şekil 3.23'de Bayer formatındaki görüntüyü Ethernet portundan okuyup sistemde kullanılan ve RGB renk uzayına sahip temel görüntü formatı olan QImage'ye dönüştüren kod parçası yer almaktadır.

```
while(true)
{
    QElapsedTimer eTimer;
    eTimer.start();
    if (frameSkipCount < 5) {
        MV_FRAME_OUT_INFO_EX dummyInfo = {};
        MV_CC_GetOneFrameTimeout(handle, pData, nPayloadSize, &dummyInfo, 100);
        frameSkipCount++;
        continue;
    }
    MV_FRAME_OUT_INFO_EX stImageInfo = {};
    int nRet = MV_CC_GetOneFrameTimeout(handle, pData, nPayloadSize, &stImageInfo, 1000);
    if (nRet == MV_OK) {
        if (stImageInfo.enPixelType == PixelType_Gvsp_BayerRG8) {
            const qsizetype dstSize = stImageInfo.nWidth * stImageInfo.nHeight * 3;
            std::unique_ptr<uchar[]> pDst(new uchar[dstSize]);

            MV_CC_PIXEL_CONVERT_PARAM conv {};
            conv.nWidth          = stImageInfo.nWidth;
            conv.nHeight         = stImageInfo.nHeight;
            conv.enSrcPixelType  = PixelType_Gvsp_BayerRG8;
            conv.enDstPixelType  = PixelType_Gvsp_RGB8_Packed;
            conv.pSrcData        = pData;
            conv.nSrcDataLen     = nPayloadSize;
            conv.pDstBuffer      = pDst.get();
            conv.nDstBufferSize  = dstSize;

            if (MV_CC_ConvertPixelType(handle, &conv) == MV_OK) {
                QImage Qimg(pDst.get(), conv.nWidth, conv.nHeight,
                            QImage::Format_RGB888);
```

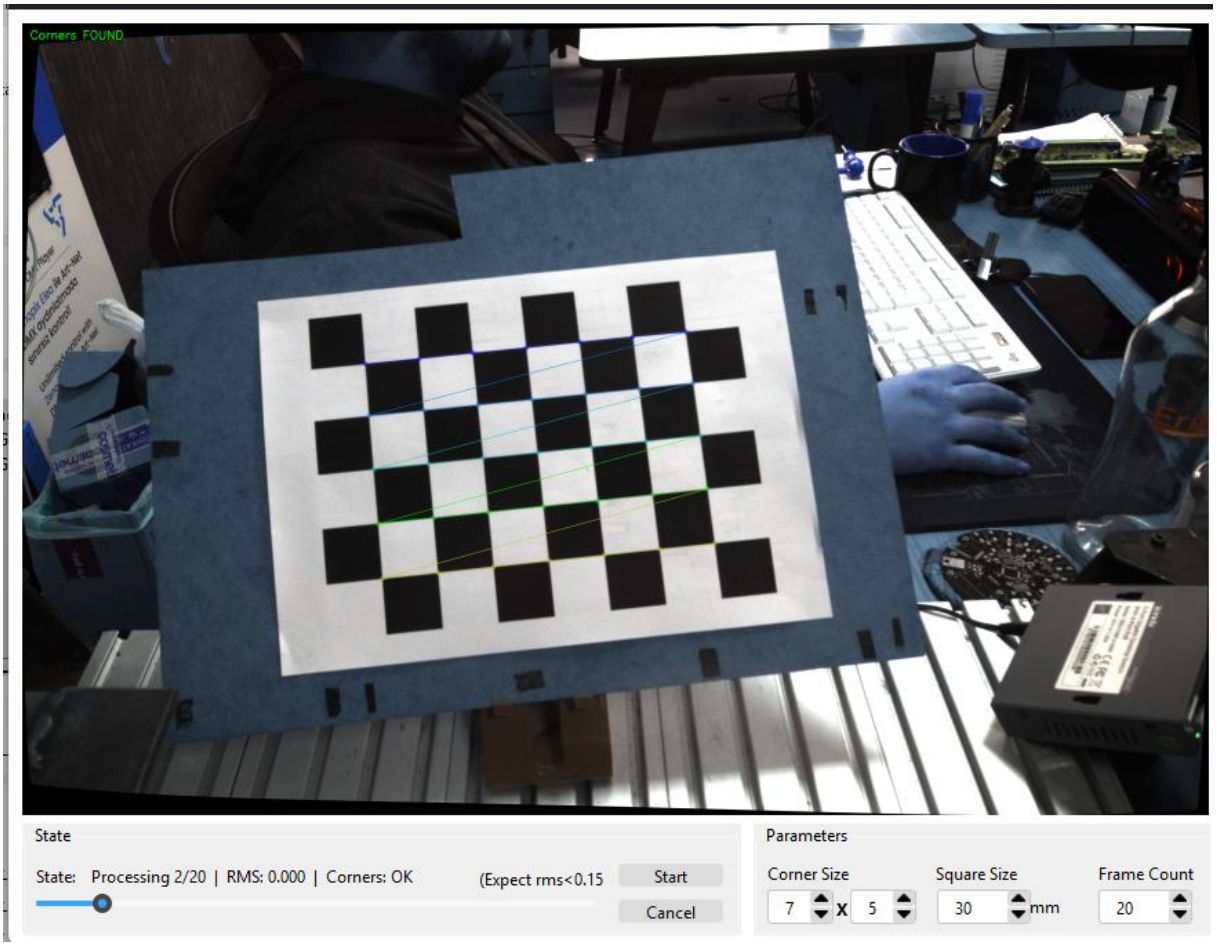
Şekil 3.23 Görüntüyü okuyan ve Bayer-RGB dönüşümü yapan iterasyon parçası

3.4.2. Kamera İç Parametrelerinin Belirlenmesi: Dama Tahtası ile Kalibrasyon

Fiziksel lenslerin doğası gereği, ideal bir iğne deliği modelinden sapmalar gösterirler. Bu sapmalar, görüntüde radyal ve teğetsel bozulmalar olarak ortaya çıkar ve gerçek dünyadaki düz çizgilerin görüntüde eğri görünmesine neden olur. Sistemin milimetrik hassasiyette ölçüm yapabilmesi için, bu lens kaynaklı geometrik hataların matematiksel bir modelle tanımlanması ve her görüntü karesinden arındırılması zorunludur. Bu çalışmada, bu sorunu çözmek için bilgisayarlı görü literatüründe yaygın olarak kabul görmüş, Zhang tarafından önerilen kalibrasyon yöntemi (Zhang, 2002) temel alınmıştır.

Bu metodoloji, sisteme entegre edilen başka bir iş parçacığı tarafından otonom olarak yürütülmüştür.

Süreç, operatörün arayüz üzerinden kalibrasyonu başlatması ve bilinen geometrik özelliklere sahip bir dama tahtası desenini görüş alanı içinde farklı açı ve konumlarda göstermesiyle başlar. Şekil 3.24'de kalibrasyon sürecine ait bir uygulama görseli yer almaktadır. Süreç, uygulama içinde kalibrasyon başlatıldığı zaman ekrana gelen bu arayüzden yönetilir ve takip edilir.



Şekil 3.24 Dama tahtası deseni tespit edildiği anda yakalanmış bir uygulama görüntüsü

1. *Görüntü Toplama:* İş parçacığı belirli zaman aralıklarıyla kullanıcı tarafından tanımlanan sayıda görüntüyü otomatik olarak yakalar.
2. *Köşe Tespiti:* Yakalanan her bir görüntü karesi için, OpenCV kütüphanesinin "findChessboardCorners" fonksiyonu kullanılarak dama tahtası desenindeki iç köşelerin piksel koordinatları yüksek hassasiyetle tespit edilir. Yazılım, köşelerin tespit edilemediği veya net olmadığı kareleri otomatik olarak eler ve sadece başarılı tespitlerin yapıldığı görüntüleri işleme dahil eder.

3. *Parametre Optimizasyonu*: Yeterli sayıda farklı duruşta görüntü toplandıktan sonra, kalibrasyon fonksiyonu çağrılır. Bu fonksiyon, her görüntüdeki 2B köşe noktaları ile bu noktaların dama tahtası düzlemindeki bilinen 3B gerçek dünya koordinatları arasındaki ilişkiyi kullanarak, Levenberg-Marquardt gibi yinelemeli bir optimizasyon algoritması çalıştırır. Bu optimizasyon süreci, kameranın içsel geometrisini tanımlayan iki temel parametre setini çözer:
    - *Kamera Matrisi (K):* Kameranın odak uzaklıklarını (fx,fy) ve optik merkezini (cx,cy) içeren 3x3 bir matristir. Bu matris, ideal bir iğne deliği kamerasının projeksiyon geometrisini tanımlar.
    - *Bozulma Katsayıları (Distortion Coefficients):* Lensin radyal ve teğetsel bozulmalarını matematiksel olarak modelleyen bir katsayılar vektörüdür (k1,k2 ,p1,p2,k3…).

Kalibrasyon süreci tamamlandığında, hesaplanan bu Kamera Matrisi ve Bozulma Katsayıları, ileride kullanılmak üzere yerel bir "yml" dosyasına kaydedilir. Sistem çalışırken, her yeni görüntü çerçevesi yakalandığında bu dosyadan okunan parametreler, OpenCV'nin "remap" fonksiyonu aracılığıyla görüntüdeki bozulmayı gerçek zamanlı olarak düzeltmek için kullanılır. Bu sayede, sonraki tüm ölçüm, tespit ve konumlandırma adımlarının geometrik olarak doğruluğu kanıtlanmış bir görüntü üzerinde yapılması garanti altına alınır (Forsyth & Ponce, 2002). Şekil 3.25'te kalibrasyonun tamamlanması sonrası, görüntü okuma döngüsü içerisinde otomatik dahil edilen ve görüntüyü kalibre eden kod parçası yer almaktadır. Kalibrasyon, OpenCV üzerinde gerçekleştirildiği için mevcut proseste önce QImage formatından OpenCV'nin görüntü formatı olan Mat formatına, ardından bu Mat formatından tekrar QImage formatına dönüşüm gerekmiştir..

```
//Kalibre et
cv::Mat matImg= Core::QImageToMat(Qimg);
if(!isCalibFileReaded)
    ReadCalibrationParams(&matImg);
if(Core::cameraYmlExists())
    remap(matImg, matImg, map1x, map1y, cv::INTER_LINEAR, cv::BORDER_CONSTANT, cv::Scalar());

//QImage dönüşüm
cv::cvtColor(matImg,matImg,cv::COLOR_BGR2RGB);
QImage calibratedQImg=Core::matToQImage(matImg);
core->camImage = calibratedQImg.copy();
```

Şekil 3.25 Gelen kamera görüntüsünü kalibrasyon filtresinden geçiren kod parçası

3.4.3. Kinematik ve Fiziksel Parametrelerin Tanımlanması

Yazılımın, dijital komutları hassas fiziksel hareketlere dönüştürebilmesi ve kamera görüntüsündeki pikselleri gerçek dünya koordinatlarına eşleyebilmesi, sistemin fiziksel geometrisini tanımlayan bir dizi temel parametrenin doğru bir şekilde bilinmesine bağlıdır. Bu parametreler; takımın boyutundan kameranın montaj pozisyonuna kadar, her bir fiziksel kuruluma özgü olan ve sistemin tüm matematiksel modellerinin temelini oluşturan sabitelerdir. Bu kritik verilerin yönetimi için, ana kontrol yazılımına özel bir yapılandırma arayüzü entegre edilmiştir.

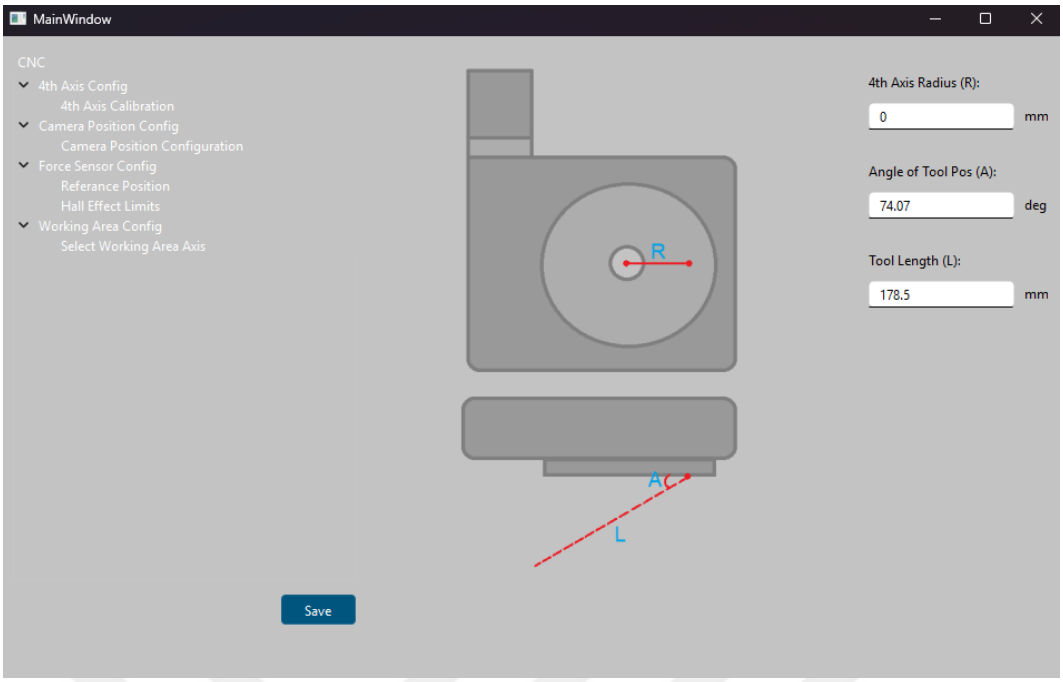
Bu arayüz, operatöre sistemin tüm fiziksel ve kinematik sabitlerini girebileceği merkezi ve kullanıcı dostu bir ortam sunmuştur. Parametreler, arayüzde mantıksal olarak kategorize edilmiştir. Operatörün, bu arayüzü kullanarak temel parametreleri tanımlaması sağlanmıştır.

3.4.3.1 Dördüncü Eksen Kinematik Parametreleri

Bu grup, A ekseni etrafında gerçekleşen dönme hareketlerinin neden olduğu takım ucu sapmasını telafi etmek için kullanılan kinematik modelin temel girdileridir (Craig, 2018).

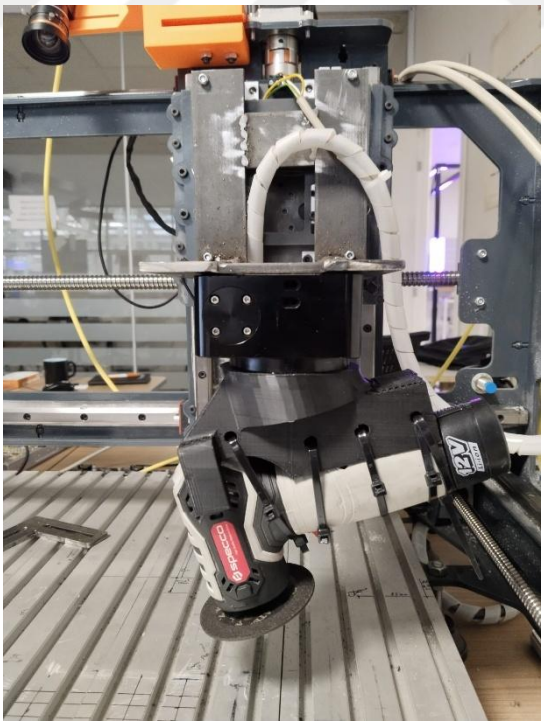
- *Takım Boyu (l):* Taşlama aparatının, A ekseninin dönme merkezinden takım ucuna kadar olan uzunluğudur. Bu değer, kinematik ofset hesaplamalarında bir kaldıraç kolu görevi görür.
- *Takım Açısı (θ):* Takımın, A eksenine göre olan montaj açısıdır.
- *Montaj Yarıçapı (r):* Taşlama aparatının, robotun 4. eksenini oluşturan döner parçasının üzerinde konumlandığı noktanın fiziksel yarıçapıdır.

Şekil 3.26'da bu parametrelerin girildiği uygulama konfigürasyon ekranının bir paneli gösterilmektedir.



Şekil 3.26 Kinematik model için gerekli olan 4. eksen parametrelerinin ayarlandığı pencere

Şekil 3.27'de ise bu parametrelerin ölçüldüğü, test düzeneğinin 4. Eksen takımı ve iş takımı montajının yapılmış, nihai yapısı yer almaktadır.



Şekil 3.27 4. eksen takımı ve şarjlı taşlama aparatı

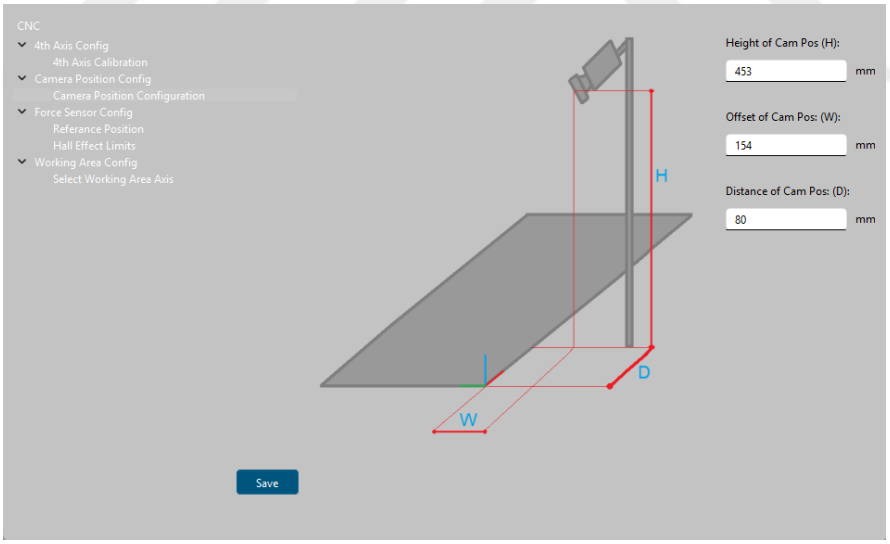
Dördüncü eksen takımı ile iş takımı birbirine ölçüleri belli bir ara parça ile monte edilmiştir. Bu sayede 4. eksen kalibrasyonu için gerekli olan parametreler daha hassas bir şekilde ölçülebilmiştir.

3.4.3.2 Kamera Konum Parametreleri

Bu parametreler, kameranın optik merkezinin, makinenin iş parçası koordinat sisteminin orijinine (0,0,0) göre olan konumunu tanımlar. Bu değerler, Bölüm 3.6.4'te detaylandırılan perspektif ofseti telafisi için hayati öneme sahiptir.

- *Kamera Yüksekliği (Hcam):* Kamera lensinin, Z=0 kabul edilen çalışma düzlemine olan dikey mesafesidir.
- *Kamera X ve Y Mesafeleri (Dcam,Wcam):* Kameranın optik merkezinin, sırasıyla makinenin Y ve X eksenlerine olan yatay uzaklıklarıdır.

Şekil 3.28'de aynı ekran üzerinde yer alan ve kamera konum parametrelerinin girilebildiği pencere yer almaktadır.

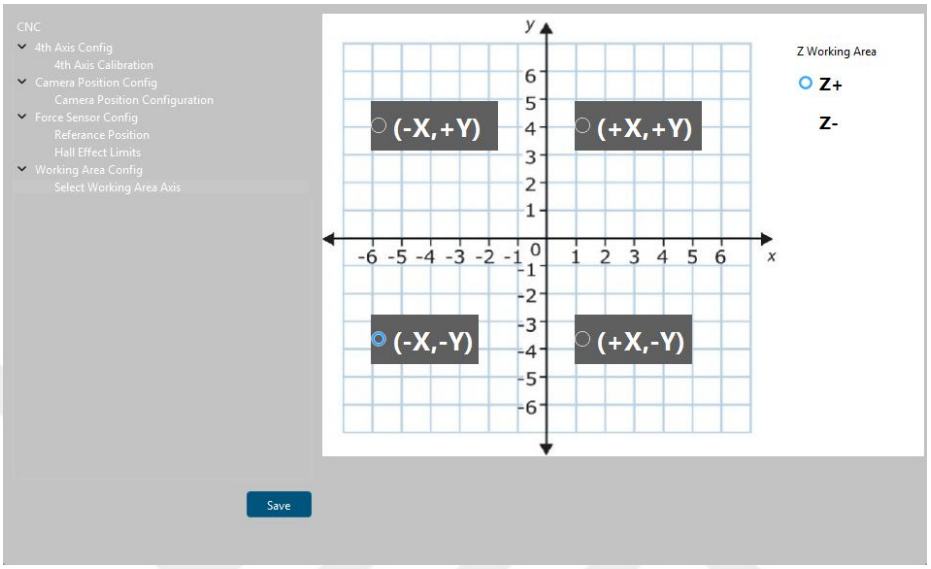


Şekil 3.28 Kamera konum bilgilerinin ayarlanabildiği pencere

3.4.3.3. Referans ve Çalışma Alanı Parametreleri

Bu ayarlar ile makinenin çalışma zarfı ve hareket yönleri tanımlanmıştır. Operatör, makinenin güvenli bir referans noktasına gitmesi için bir Ref_Point tanımlayabilir. Ayrıca,

kameranın bakış yönüne göre hangi kartezyen çeyrekte çalışılacağını belirterek, görüntü koordinatlarının makine G-kodlarına doğru bir şekilde dönüştürülmesini sağlar. Şekil 3.29'da bu ayarları yapabildiği pencere gösterilmektedir. Bu pencere de diğer konfigürasyonlarla birlikte aynı ekranı paylaşmıştır.

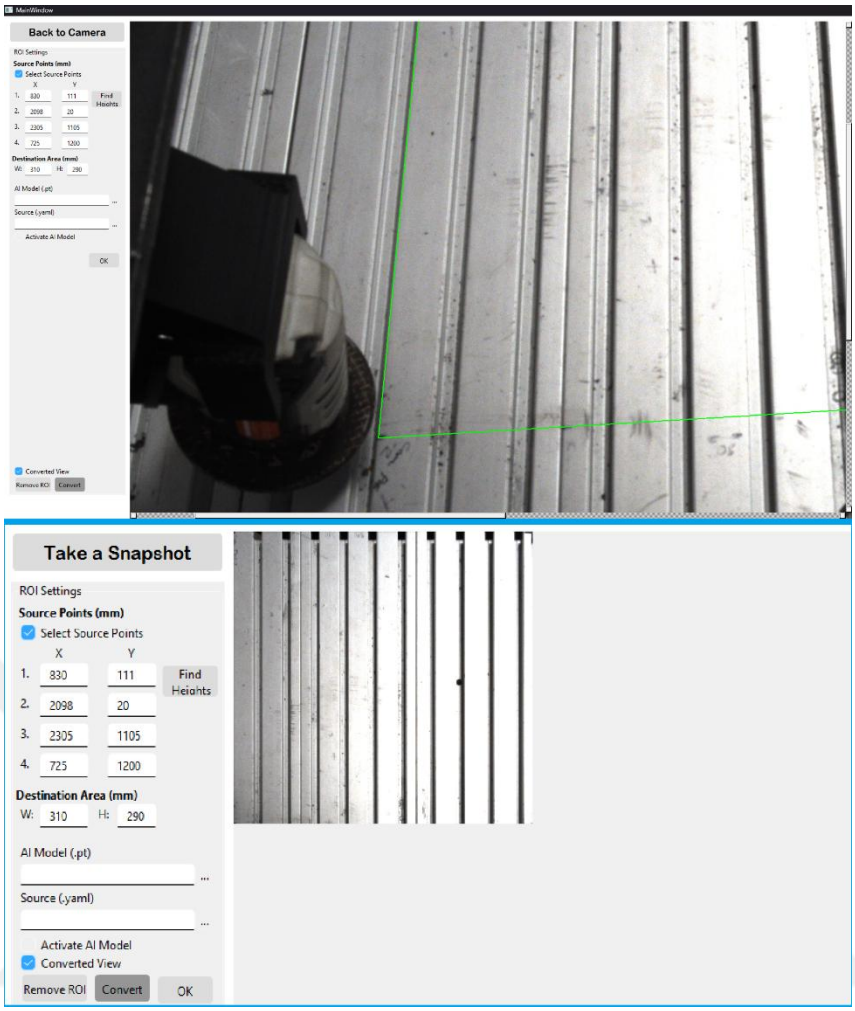


Şekil 3.29 Çalışma uzayının seçildiği ve referans değerlerin atandığı ekran

Operatör tarafından girilen tüm bu parametreler, uygulamanın yeniden başlatılmasında tekrar girilmesine gerek kalmaması için yerel bir yapılandırma dosyasına kaydedilmiştir. Tüm hesaplamaların doğruluğu, bu parametrelerin hassasiyetine doğrudan bağlı olduğundan, bu adım sistemin doğru çalışmasının temelini oluşturur.

3.4.4. Metrik Ölçeklendirme için Perspektif Dönüşüm

Standart bir kamera görüntüsündeki pikseller, metrik bir birime sahip değildir; bir pikselin temsil ettiği gerçek dünya alanı, nesnenin kameraya olan uzaklığına ve görüntü düzlemindeki konumuna bağlı olarak sürekli değişir. Bu ölçek belirsizliği, görüntü üzerinden hassas ölçüm yapmanın önündeki en büyük engeldir. Bu sorunu aşmak ve görüntü koordinatları ile gerçek dünya metrik ölçüleri arasında doğrudan bir ilişki kurmak için bu tez kapsamında metrik ölçekli bir İlgi Bölgesi (Region of Interest - ROI) oluşturma yöntemi geliştirilmiştir. Şekil 3.30'da kamera arayüzü üzerinden yapılan ROI dönüşümünün görseli yer almaktadır.



Şekil 3.30 ROI dönüşümü öncesi ve alan seçimi yapıldıktan sonra ROI dönüşümü sonrası kamera ekranı

Süreç, kamera ayar arayüzü üzerinden operatör tarafından yönetilir ve bir dizi aşama içerir.

3.4.4.1 Alan Seçimi ve Boyut Tanımlama

Operatör, öncelikle kalibrasyonu yapılmış (distorsiyonu giderilmiş) canlı kamera görüntüsü üzerinde, gerçek dünyada boyutlarını bildiği dikdörtgen bir alanın dört köşesini fare ile işaretler. Ardından, bu dikdörtgenin gerçek genişlik ve yükseklik değerlerini milimetre cinsinden arayüze girer. Tüm bu süreçler Şekil 3.30'da her iki görselde de yer alan sol panelden yürütülür.

3.4.4.2 Homografi Matrisinin Hesaplanması

Yazılımın, operatörün seçtiği bu dört kaynak noktasının piksel koordinatları ile bu noktaların karşılık geldiği, milimetre cinsinden tanımlanmış hedef dikdörtgenin köşe koordinatları arasında bir eşleme yapması sağlanmıştır. OpenCV kütüphanesinin

“getPerspectiveTransform” fonksiyonu kullanılarak, bu iki dörtgen arasındaki düzlemsel dönüşümü tanımlayan 3x3 boyutunda bir homografi matrisi (H) hesaplanmıştır. Homografi, bir düzlemin farklı bakış açılarından elde edilen iki görüntüsü arasındaki geometrik ilişkiyi tanımlayan temel bir kavramdır (Hartley ve & Zisserman, 2004). Şekil 3.31’de uygulanan prosese ait kodlar yer almaktadır.

```cpp
cv::Mat openCVROI::transformAndCropImage(const cv::Mat& inputImage, const std::vector<cv::Point2f>& corners, int realWidth, int realHeight) {
    // Doğru sayıda köşe noktası olduğunu kontrol edin
    if (corners.size() != 4) {
        std::cerr << "Hatalı köşe nokta sayısı!\n";
        return cv::Mat();
    }

    // Hedef köşe noktaları
    std::vector<cv::Point2f> targetCorners;
    targetCorners.push_back(cv::Point2f(0, 0));
    targetCorners.push_back(cv::Point2f(realWidth, 0));
    targetCorners.push_back(cv::Point2f(realWidth, realHeight));
    targetCorners.push_back(cv::Point2f(0, realHeight));

    // En-boy oranını hesaplayın
    float aspectRatio = calculateAspectRatio(corners);

    // Hedef köşe noktalarını güncelleme
    //  updateTargetCorners(targetCorners, aspectRatio);

    // Dönüşüm matrisini hesaplayın
    cv::Mat perspectiveMatrix = cv::getPerspectiveTransform(corners, targetCorners);;

    // Düzleştirilmiş görüntüyü oluşturun
    cv::Mat flattenedImage;
    cv::warpPerspective(inputImage, flattenedImage, perspectiveMatrix, cv::Size(realWidth, realHeight));

    // Perspektif dönüşümü sonucu elde edilen görüntüyü çerçevenin sınırları içinde tutmak için boyutları kontrol edin
    /* cv::Rect boundingRect = cv::boundingRect(targetCorners); ...*/
    cout<<flattenedImage.size();
    // Beyaz hatları çıkar ve boyut bilgilerini yazdır
    return flattenedImage;
}

void openCVROI::updateTargetCorners(std::vector<cv::Point2f>& targetCorners, float aspectRatio) {
    // İlk iki noktanın arasındaki mesafeyi hesaplayın
    float distance = cv::norm(targetCorners[0] - targetCorners[1]);

    // Yeni genişlik ve yükseklik değerlerini hesaplayın
    float newWidth, newHeight;
    if (aspectRatio > 1.0f) {
        newWidth = distance;
        newHeight = distance / aspectRatio;
    } else {
        newWidth = distance * aspectRatio;
        newHeight = distance;
    }

    // Hedef köşe noktalarını güncelleyin
    targetCorners[1] = targetCorners[0] + cv::Point2f(newWidth, 0);
    targetCorners[2] = targetCorners[0] + cv::Point2f(newWidth, newHeight);
    targetCorners[3] = targetCorners[0] + cv::Point2f(0, newHeight);
}
```

Şekil 3.31 Homografi matrisi ve perspektif dönüşümünün uygulandığı perspektif fonksiyonu ve alt fonksiyonu

3.4.4.3 Perspektif Dönüşümünün Uygulanması

Hesaplanan homografi matrisi, OpenCV’nin warpPerspective fonksiyonu aracılığıyla tüm görüntü çerçevesine uygulanmıştır. Bu işlem, orijinal görüntüdeki eğik düzlemi "düzleştirerek" sanki kameradan tam tepeden bakılıyormuş gibi bir görüntü oluşturur.

Bu metodun en kritik çıktısı, elde edilen yeni görüntünün (ROI) metrik bir ölçeğe sahip olmasıdır. Hedef dikdörtgenin boyutları milimetre olarak tanımlandığı için, dönüştürülmüş görüntüde 1 piksellik bir mesafe, gerçek dünyadaki 1 milimetrelik bir mesafeye yaklaşık olarak eşitlenmiştir. Bu **1px ≈ 1mm** eşlemesi, sistemin sonraki tüm adımları için temel bir kolaylık sağlamıştır. Yapay zekâ modelinin tespit ettiği nesne koordinatları doğrudan milimetrik konumlara dönüştürülebilmiş ve en önemlisi, tek kamerayla stereo vizyon yönteminde hesaplanan piksel kayması (disparite), doğrudan milimetrik bir değer olarak derinlik formülünde kullanılabilmiştir. Bu sayede, sistemin ölçüm hassasiyeti ve doğruluğu önemli ölçüde artırılmıştır.

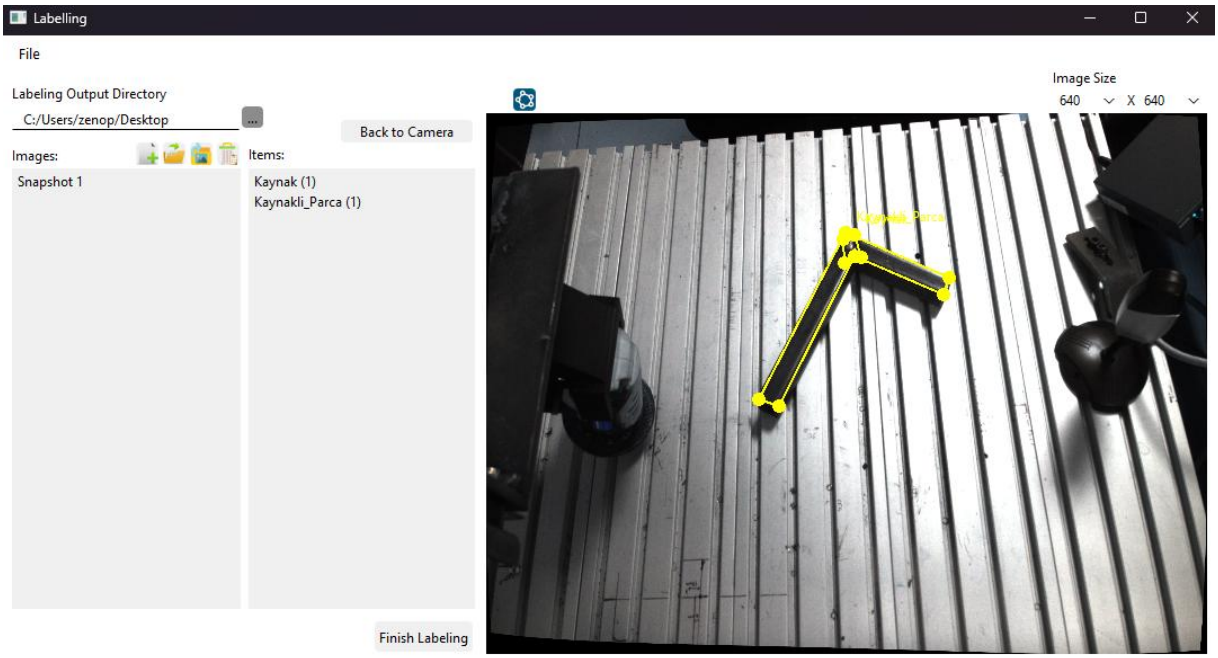
Şekil 3.31'deki iki fonksiyon, bu dönüşümlerin merkezini oluşturmaktadır. Hem kamera arayüzünde hem de görüntüleme sürecinde ara filtre katmanı olarak kullanılmaktadırlar.

3.5. Yapay Zekâ Tabanlı Kaynak Tespit Modelinin Geliştirilmesi

Sistemin otonom olarak kaynak dikişlerini bulup işlemesi, eğitilmiş bir derin öğrenme modelinin bu özel nesneleri bir görüntü üzerinde doğru bir şekilde tespit etme yeteneğine dayanır. Bu bölümde, bu yeteneği kazandırmak için geliştirilen uçtan uca yapay zekâ işlem hattı ayrıntılı olarak ele alınmıştır. Bu süreç; özel bir veri kümesinin oluşturulmasından, modelin eğitilmesine ve nihai olarak çıkarım için doğrulanmasına kadar olan tüm metodolojik adımları kapsamaktadır. Geliştirilen mimarinin temel hedeflerinden biri, bu adımların tamamını ana kontrol yazılımına entegre ederek, endüstriyel bir ortamda bile hızlı ve pratik bir şekilde modelin güncellenmesine olanak tanımaktır.

3.5.1. Veri Kümesi Oluşturma ve Etiketleme Arayüzü

Denetimli öğrenmeye dayalı derin öğrenme modellerinin başarısı, doğrudan üzerine eğitildikleri veri kümesinin kalitesine ve miktarına bağlıdır. Bu tez çalışmasına özgü kaynak dikişi tespiti gibi niş bir endüstriyel görev için hedef ortam şartlarına yönelik hazır veri kümeleri bulunmama ihtimali göz önünde bulundurulduğundan, bu çalışmanın bir parçası olarak özel bir veri toplama ve etiketleme aracı geliştirilmiştir. Şekil 3.32'de veri etiketleme ekranının görüntüsü yer almaktadır.



Şekil 3.32 Veri etiketleme ekranı ve etiketlenen bir nesne

Geliştirilen veri seti oluşturma arayüzü, operatörün etiketleme sürecini verimli bir şekilde yönetmesini sağlar. Operatör, bu arayüzü kullanarak mevcut görüntü dosyalarını sisteme yükleyebilir veya doğrudan CNC üzerine monte edilmiş kameranın canlı akışından anlık görüntüler alarak kendi veri kümesini oluşturabilir ve aynı anlık görüntüleri sisteme eklediği veri setinde de kullanarak önceki veri setlerini güncelleyebilir.

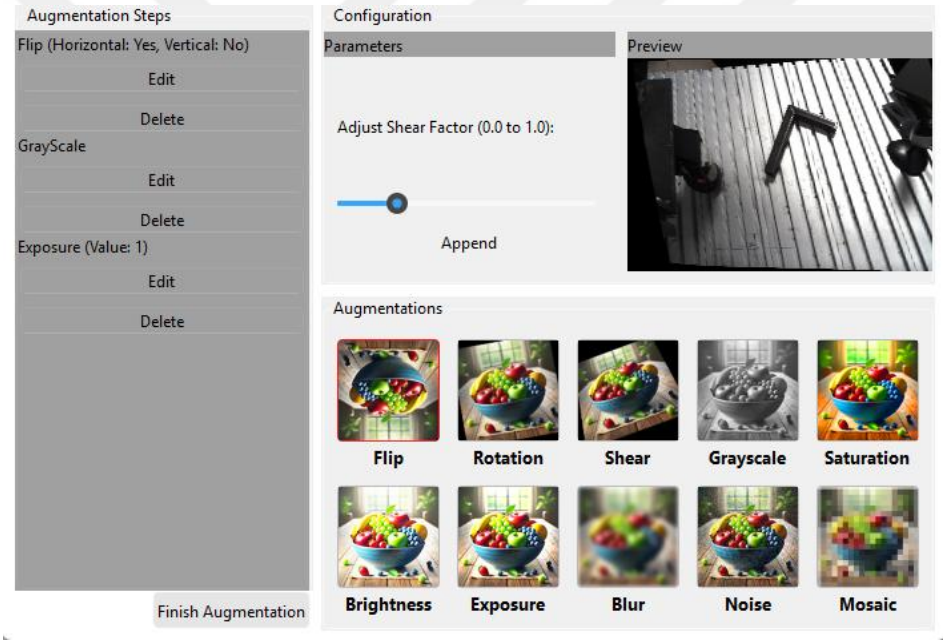
Etiketleme işleminin kendisi, üzerine çizim yapılabilen özelleştirilmiş bir QLabel bileşeni üzerinde gerçekleştirilir. Bu arayüzde operatör, tespit edilmesi istenen kaynak dikişlerinin etrafını hassas bir şekilde çevreleyen poligonlar çizer. Bir poligonun çizimi tamamlandığında, sistem operatörden bu şekle bir sınıf etiketi (örn. "kaynak") atamasını ister. Yazılım, her bir görüntü için bu etiket bilgilerini (sınıf adı, şekil tipi ve poligonun köşe koordinatları) bellekte saklayarak, eğitim için gerekli olan "görüntü-etiket" çiftlerini oluşturur. Bu manuel fakat esnek yaklaşım, modele neyi öğrenmesi gerektiğinin hassas bir şekilde öğretilmesini sağlar.

3.5.2. Veri Artırma Stratejilerinin Belirlenmesi

Derin öğrenme modellerinin, farklı koşullar altında dahi tutarlı bir başarım sergileyebilmesi (genelleme yapabilmesi), çok çeşitli verilerle eğitilmesini gerektirir. Sınırlı sayıda toplanan orijinal görüntülerle bu çeşitliliği sağlamak zordur. Bu sorunu aşmak ve

modelin aşırı öğrenmesini (overfitting) önlemek için veri artırma (data augmentation) teknikleri, mevcut veri kümesini yapay olarak zenginleştirmek ve çeşitlendirmek amacıyla kullanılır (Shorten ve & Khoshgoftaar, 2019).

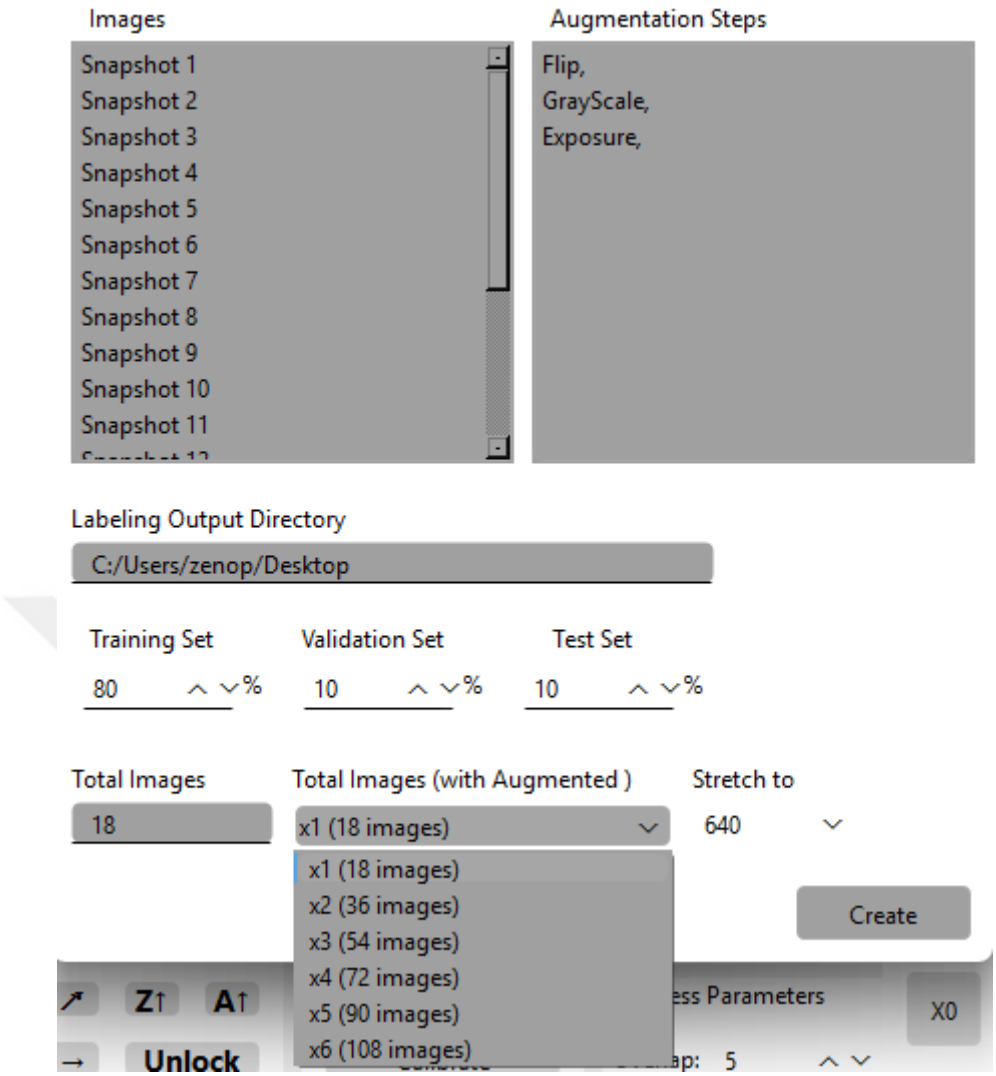
Bu tez kapsamında geliştirilen özel bir arayüz, operatörün bir dizi artırma tekniğini bir işlem hattı olarak tanımlamasına olanak tanır. Operatör, arayüzdeki butonlar aracılığıyla geometrik (döndürme, çevirme vb.) ve fotometrik (parlaklık, gürültü ekleme vb.) dönüşümlerden istediğini seçer. Her seçim için, o dönüşüme özel parametrelerin ayarlanabildiği bir kontrol paneli belirir. Operatör, yaptığı ayarın etkisini bir ön izleme penceresinde anlık olarak gördükten sonra bu operasyonu ve ayarlarını işlem hattına dahil eder. Bu süreçte anlık bir görüntü işleme yapılmaz; bunun yerine, bir sonraki aşamada kullanılacak olan bir "tarif" listesi oluşturulur. Şekil 3.33'de geliştirilen veri artırma ekranının görüntüsü yer almaktadır.



Şekil 3.33 Veri artırma ekranı ve seçilen dönüşüm adımları

3.5.3. Veri Seti Oluşturma ve Bölümleme Arayüzü

Veri etiketleme ve artırma stratejileri tanımlandıktan sonraki adım, bu girdileri kullanarak nihai eğitim veri setini oluşturmaktır. Veri seti oluşturma hattının bu son bölümünü idare eden bölümleme penceresi, bu yoğun işlem öncesinde son konfigürasyonların yapıldığı bir kontrol merkezi görevi görür. Şekil 3.34'de bu ekran gösterilmektedir.



Şekil 3.34 Veri seti oluşturma ekranı ve bölümleme arayüzü

Bu arayüzde operatöre şu bilgiler sunulur ve ayarlar talep edilir:

- *Özet Bilgiler:* Etiketlenmiş tüm görüntülerin ve tanımlanmış artırma adımlarının bir listesi gösterilerek sürecin girdileri teyit edilir.
- *Veri Seti Boyutlandırma:* Kullanıcı, eğitimde kullanılacak görüntülerin nihai çözünürlüğünü (örn. 640x640 piksel) seçer.
- *Veri Bölümleme:* Modelin performansını objektif bir şekilde ölçebilmek için veri setinin bölünmesi gerekir. Bu ekranda kullanıcı, oluşturulacak toplam veri setinin yüzde kaçının eğitim (training), yüzde kaçının doğrulama (validation) ve yüzde kaçının test (test) için ayrılacağını belirler. Eğitim seti modelin parametrelerini öğrenmesi için, doğrulama seti eğitim sırasında hiperparametre ayarı ve aşırı ezberlemeyi kontrol etmek

için, test seti ise eğitim tamamen bittikten sonra modelin son ve tarafsız performansını ölçmek için kullanılır.

- *Çıktı Konumu:* Oluşturulacak veri setinin kaydedileceği disk konumu seçilir.

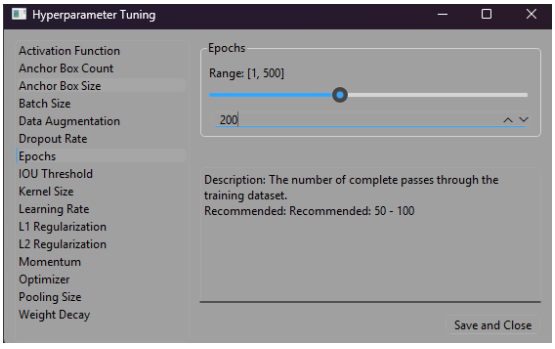
Tüm bu ayarlar yapıldıktan sonra bu konfigürasyonları girdi olarak alan QThread tabanlı iş parçacığı arka planda çalışmaya başlayarak artırılmış veri setini diske yazar.

3.5.4. Hiper Parametre Ayarlama ve Eğitimi Başlatma

Bu aşama, model eğitimini tetiklemeden önceki son kullanıcı adımlarını içerir.

- *Hiper Parametre Ayarlama:* Bir derin öğrenme modelinin öğrenme süreci, öğrenme oranı (learning rate), yığın boyutu (batch size), epoch sayısı gibi bir dizi hiper parametre tarafından yönetilir. Bu parametrelerin doğru seçilmesi, modelin başarımı üzerinde doğrudan etkilidir. Geliştirilen hiper parametre seçim ekranı ise, operatörün bu kritik değerleri kod yazmadan, kaydırıcılar, metin kutuları ile açıklama metinleri dahilinde ayarlamasına olanak tanır.
- *Eğitimi Başlatma:* Bu arayüz, eğitimin başlatılması için son aşamadır. Operatör burada iki temel konumu belirtir: bir önceki adımda oluşturulan ve veri setinin yapısını tanımlayan “dataset.yaml” dosyasının yolu ve eğitim sonucunda oluşacak olan model ağırlıkları, loglar gibi çıktı dosyalarının kaydedileceği klasör. "Eğitimi Başlat" butonuna tıklandığında, uygulama tüm bu bilgileri (veri seti yolu, çıktı konumu ve ayarlanmış hiper parametreler) Python betiğinin bilgi almak amaçlı denetleyeceği ve konumu sabit (Dökümanlar klasörü) “parameters.json” adlı dosyaya yazar ve Python eğitim betiğini tetikler.

Şekil 3.35’te bu parametrelerin ayarlandığı uygulama ekranı yer almaktadır.



Şekil 3.35 Hiper parametre konfigürasyon ekranı

3.5.5. YOLOv8 Mimarisi ile Modelin Arka Planda Eğitilmesi

Şekil 3.36'de yer alan Python betiği bir mikro servis gibi çalışarak hazırlanmış veri setini arka planda eğitmekte ve eğitim durumunu loglamaktadır.

```
try:
    model = YOLO(model_path)
    status_thread = Thread(target=monitor_status_flag)
    status_thread.daemon = True  # Program kapandığında thread otomatik sonlanır
    status_thread.start()

    update_json_status("progress", "Preparing training")

    for epoch in range(1, training_params["epochs"] + 1):
        try:
            print(f"Current Epoch: {epoch}/{training_params['epochs']}")
            update_json_status("progress", f"{epoch}/{training_params['epochs']}")
            model.train(
                data=config["dataPath"],
                epochs=1,  # Her epoch ayrı çalıştırılır
                batch=training_params["batch_size"],
                imgsz=training_params["img_size"],
                lr0=training_params["learning_rate"],
                project=config["trainingOutputDir"],
                name='training_run',
                verbose=True,
                exist_ok=True,
                workers=0
            )
            print(f"Epoch {epoch}/{training_params['epochs']} tamamlandi");

        except Exception as epoch_error:
            update_json_status("error", f"Epoch {epoch} sırasında hata: {str(epoch_error)}")
            print(f"Epoch {epoch} sırasında hata: {epoch_error}")

    update_json_status("progress", "Validating model")
    metrics = model.val()

    results = {
        "mAP_50": metrics.box.map50 if model_type == "Detection" else metrics.seg.map50,
        "mAP_50-95": metrics.box.map if model_type == "Detection" else metrics.seg.map,
    }
    update_json_status("results", results)

    results_df = pd.DataFrame([results])
    output_csv = os.path.join(config["trainingOutputDir"], "training_results.csv")
    results_df.to_csv(output_csv, index=False)
```

Şekil 3.36 YOLOv8 mimarisi ile eğitim yapan Python main bloğu

Bu Python betiği, C++ arayüzünden tamamen bağımsız bir süreç olarak çalışan ve asıl eğitimi gerçekleştiren motordur. Çalışma mantığı şu şekildedir:

1. *Konfigürasyonu Okuma:* Betik, başlatıldığında C++ uygulamasının oluşturduğu "parameters.json" dosyasını okuyarak tüm görev tanımını (veri yolu, hiper parametreler vb.) alır.
2. *Modeli Başlatma:* Transfer öğrenmesi (Pan ve & Yang, 2009) tekniğinden faydalanarak, genel nesneleri tanımak üzere önceden eğitilmiş standart bir YOLOv8 modelini yükler.
3. *Eğitim Döngüsü:* Ultralytics kütüphanesinin train() fonksiyonunu çağırarak, özel olarak oluşturulmuş kaynak dikişi veri seti üzerinde modeli yeniden eğitmeye başlar.

4. *Gerçek Zamanlı Raporlama:* Betik, tamamladığı her epoch sonrasında, ilerleme durumunu yine aynı "parameters.json" dosyasına yazar. C++ arayüzü bu dosyayı sürekli izleyerek kullanıcıya canlı bir ilerleme çubuğu sunar.
5. *Doğrulama ve Kayıt:* Eğitim tamamlandığında, modelin başarımını doğrulama seti üzerinde test eder ve mAP gibi metrikleri hesaplayarak yine JSON dosyasına kaydeder. En iyi başarım gösteren ve son eğitim döngüsünün model ağırlıklarını "best.pt" ve "last.pt" adıyla diske yazar ve son olarak, modeli C++ üzerinde de daha kolay bir entegrasyonla çalışabilmesi adına ONNX formatına (Microsoft ve ark., 2017) dönüştürür.

3.6. Uçtan Uca Otonom İşlem Metodolojisi

Önceki bölümlerde detaylandırılan donanım altyapısı, yazılım mimarisi, sistem kalibrasyonları ve yapay zekâ modeli, bu aşamada bütünleşik bir işlem hattı olarak bir araya gelir. Bu bölüm, sistemin hazırlık aşamasından çıkıp, bir iş parçası üzerindeki hedefi (kaynak dikişini) otonom olarak tespit etme, üç boyutlu uzayda konumlandırma ve hassas bir şekilde yüzey işleme görevini tamamlama sürecini adım adım açıklamaktadır. Bu uçtan uca metodoloji, sistemin gerçek zamanlı algılama, karar verme ve eyleme geçme kabiliyetinin temelini oluşturur.

3.6.1. ROI Üzerinde Eğitilmiş Model ile Gerçek Zamanlı Kaynak Tespiti

Otonom işlemin ilk adımı, metrik olarak ölçeklendirilmiş ROI görüntüsü üzerinde kaynak dikişinin yerinin hassas bir şekilde tespit edilmesidir. Bu görev, C++ tabanlı ana kontrol uygulaması ile Python tabanlı yapay zekâ çıkarım betiği arasında kurulan gerçek zamanlı bir haberleşme köprüsü aracılığıyla gerçekleştirilir. Bu hibrit mimari, C++'ın gerçek zamanlı donanım kontrolü ve arayüz yönetimindeki gücü ile Python'un zengin yapay zekâ ekosistemini bir araya getirme zorunluluğundan doğmuştur. İki farklı süreç arasındaki bu veri akışı, aşağıdaki adımlarla yönetilir:

1. *Arka Plan Sürecinin Başlatılması***:** Ana kontrol uygulaması ilk kez başlatıldığında, tanımlayıcı Python betiğini QProcess sınıfı aracılığıyla kalıcı bir arka plan süreci olarak tetikler. Bu betik, başlatıldıktan sonra bir komut bekleme döngüsüne girer. Bu yaklaşımın temel amacı, her bir görüntü tespiti için Python yorumlayıcısını ve ağır yapay zekâ modelini yeniden yüklemenin getireceği zaman gecikmesini ortadan

kaldırarak, sistemi gerçek zamanlı çıkarım için sürekli hazır halde tutmaktır. Şekil 3.37'de arka plan yorumlayıcısının main yapısı yer almaktadır.

```python
class ObjectDetection:
    """
    def __init__(self, model_path: str, class_names: list):

    def detect(self, frame: np.ndarray):
        """Frame üzerinde tespiti çalıştır."""
        return self.model(frame)

    def extract_polygons(self,
                         results,
                         confidence_threshold: float = 0.7,
                         min_contour_area: float   = 100.0) -> list:

        detections = []
        if not hasattr(results[0], 'masks') or results[0].masks is None:
            return detections
        model_h, model_w = results[0].orig_shape
        # Gerçek frame boyutları:
        orig_h, orig_w  = results[0].orig_shape  # Ultralitycs v8+ otomatik ölçeklenmiş çıkış veriyor
        cls_ids = results[0].boxes.cls.cpu().numpy()
        confs   = results[0].boxes.conf.cpu().numpy()

        for mask_xy, class_id, conf in zip(results[0].masks.xy, cls_ids, confs):

        return detections
def log_error(message: str):
def load_config():
def update_state_and_polygons(state: str, detections: list):
def load_class_names(yaml_path: str) -> list:
def main():
    detector = None

    while True:
        pt_path, state, yaml_path = load_config()

        # Modeli bir kez başlat
        if pt_path and detector is None:
            try:
                class_names = load_class_names(yaml_path)
                detector    = ObjectDetection(pt_path, class_names)
            except Exception as e:
                log_error(f"Model init failed: {e}")
                time.sleep(SLEEP_SECONDS)
                continue

        # Başla komutu geldiyse
        if state == 'start' and detector:
            if ROI_IMAGE_PATH.exists():
                frame = cv2.imread(str(ROI_IMAGE_PATH))
                if frame is None:
                    log_error("ROI image load failed")
                else:
                    results    = detector.detect(frame)
                    detections = detector.extract_polygons(results)
                    update_state_and_polygons('detected', detections)

        print("epoch completed")
        time.sleep(SLEEP_SECONDS)
```

Şekil 3.37 Arka plan Python yorumlayıcısı

2. *Görüntünün Hazırlanması ve İletimi:* "MVCameraThread" iş parçacığı, kameradan aldığı, distorsiyonu giderilmiş görüntüye, Bölüm 3.4.4'te açıklanan homografi matrisini uygulayarak metrik ölçekli (1px ≈ 1mm) ROI görüntüsünü oluşturur. Bu nihai görüntü, iki sürecin de erişebildiği, önceden tanımlanmış ve diğer parametrelere ortak klasör yolu olan "..Dokümanlar/AIPolishing" dosya yoluna kaydedilir. Bu dosya, C++'tan Python'a gönderilen ana veri yükünü temsil eder.
3. *Çıkarım Komutunun Tetiklenmesi:* Görüntüyü kaydettikten hemen sonra, C++ uygulaması paylaşılan tanımlayıcı JSON dosyası üzerinde değişiklik yapar. Bu dosyaya, kullanılacak olan eğitilmiş modelin (.pt) ve sınıf isimlerini içeren YAML dosyasının yollarını yazar. Tüm parametrik hazırlıklar yapıldıktan sonra JSON dosyası içerisindeki,

"detectionState" adlı bir anahtarın değerini "start" olarak günceller. Bu JSON dosyası, iki süreç arasında bir işaretleşme (semaphore) mekanizması görevi görerek Şekil 3.37'de yer alan Python betiğine "işe başla" komutunu iletir.

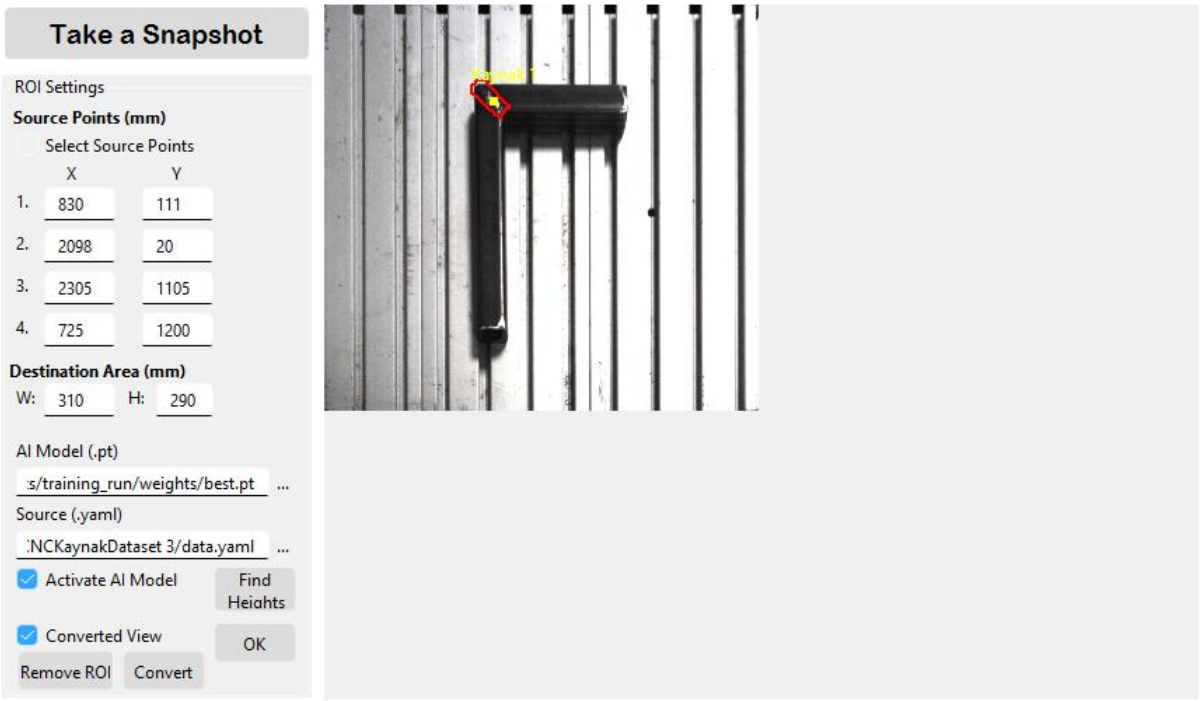
4. *Yapay Zekâ ile Tespit:* Arka planda beklemede olan python betiği, periyodik olarak bu JSON dosyasını kontrol eder. "detectionState" anahtarının "start" olarak değiştiğini algıladığında bekleme döngüsünden çıkar ve aşağıdaki işlemleri yapar:
    - Paylaşılan dosya yolundan ROI görüntüsünü okur.
    - Görüntüyü, önceden belleğe yüklenmiş olan eğitilmiş YOLOv8 modeline girdi olarak verir.
    - Model, görüntü üzerinde çıkarım yaparak kaynak dikişlerinin sınıfını, güven skorunu ve segmentasyon maskesini (poligon köşe noktalarını) içeren sonuçları üretir.
5. *Sonuçların Geri Yazılması:* Python betiği, modelin ham çıktısını işler. Belirli bir güven skorunun üzerindeki tespitleri geçerli kabul ederek, her bir tespit edilen kaynak dikişi için sınıf adını ve poligon köşe noktalarının listesini çıkarır. Bu yapılandırılmış veriyi, aynı tanımlayıcı JSON dosyasının üzerine yazarak günceller ve en son adım olarak "detectionState" anahtarının değerini "detected" olarak değiştirir. Bu durum değişikliği, C++ uygulamasına görevin tamamlandığı ve sonuçların okunmaya hazır olduğu sinyalini verir.
6. *Sonuçların Alınması ve İşlenmesi:* Python betiğini tetikledikten sonra bir yoklama döngüsüne giren MVCameraThread, JSON dosyasındaki durumun "detected" olarak değiştiğini gördüğünde dosyayı okur. Dosya içindeki poligon koordinatlarını ayrıştırarak C++ tarafındaki QVector gibi veri yapılarında saklar. Bu veriler, artık bir sonraki adım olan derinlik hesaplama ve hareket planlama için hazır hale gelmiştir. Eş zamanlı olarak, tespit edilen bu poligonlar arayüzdeki ROI görüntüsü üzerine çizilerek operatöre görsel geri bildirim sağlanır. Ayrıca, tespit edilen her bir kaynak poligonu için, poligonu oluşturan tüm köşe noktalarının aritmetik ortalaması alınarak geometrik bir merkez noktası hesaplanır. Bu merkez noktası, poligonla birlikte arayüzde belirgin bir şekilde işaretlenir. Bu işlemin temel amacı, bir sonraki adım olan derinlik hesaplaması için kararlı ve tek bir referans noktası oluşturmaktır. Sistemin, kontrollü yanal hareket sonrası iki görüntü arasındaki piksel kaymasını (disparite) ölçeceği nokta, bu hesaplanan merkez noktası olacaktır. Şekil 3.38'de okunan parametreler dahilinde yapılan çizim kodları yer almaktadır.

```
while (true)
{
    if (waiter.elapsed() > 3000) break;

    QFile jsonCheckFile(jsonPath);
    if (jsonCheckFile.open(QIODevice::ReadOnly))
    {
        QByteArray data = jsonCheckFile.readAll();
        QJsonParseError parseError;
        QJsonDocument doc = QJsonDocument::fromJson(data, &parseError);
        jsonCheckFile.close();

        if (parseError.error == QJsonParseError::NoError && doc.isObject())
        {
            QJsonObject obj = doc.object();
            QString state = obj.value("detectionState").toString();
            if (state == "detected")
            {
                QPainter painter(&qimg);
                weldPoints.clear(); // Tüm noktaları sıfırla

                QJsonArray detections = obj.value("polygons").toArray();
                for (const QJsonValue &detVal : detections)
                {
                    QJsonObject detObj = detVal.toObject();
                    QString name = detObj.value("name").toString();

                    QJsonArray polyArr = detObj.value("polygon").toArray();
                    QPolygon poly;
                    for (const QJsonValue &ptVal : polyArr)
                    {
                        QJsonArray coord = ptVal.toArray();
                        if (coord.size() >= 2)
                        {
                            int x = coord.at(0).toInt();
                            int y = coord.at(1).toInt();
                            poly << QPoint(x, y);
                        }
                    }

                    if (!poly.isEmpty())
                    {
                        // Poligon çiz
                        QJsonArray colorArr = detObj.value("color").toArray();
                        int b = colorArr.at(0).toInt();
                        int g = colorArr.at(1).toInt();
                        int r = colorArr.at(2).toInt();
                        QPen pen(QColor(r, g, b));
                        pen.setWidth(2);
                        painter.setPen(pen);
                        painter.drawPolygon(poly);

                        // Eğer kaynak noktasıysa, merkezini hesaplayıp listeye ekle
                        if (name == "kaynak") {
                            QPoint center(0, 0);
                            QVector<QPoint> polyPoints;
                            for (const QPoint &pt : poly) {
                                center += pt;
                                polyPoints.append(pt);
                            }
                            center /= poly.size();

                            QString label = QString("Kaynak %1").arg(weldPoints.size() + 1);

                            QPen yellowPen(Qt::yellow);
                            yellowPen.setWidth(6);
                            painter.setPen(yellowPen);
                            painter.drawPoint(center);
                            painter.drawText(poly.boundingRect().topLeft(), label);

                            WeldPointInfo info;
                            info.name = label;
                            info.center = center;
                            info.height = -1.0; // henüz bilinmiyor
                            info.polygons=polyPoints;
                            weldPoints.append(info);
                        }
                    }
                }

                qDebug() << "Detection completed and drawn!";
                break;
            }
        }
    }

    QThread::msleep(30); // 30ms bekle
}
```

Şekil 3.38 Periyodik döngülerle son kaydedilen resmin AI çıktılarını bekleyen ve çıktı sonuçlarına göre çizimi gerçekleştiren kod bloğu

Bu asenkron ve dosya tabanlı haberleşme mimarisi, farklı teknolojilerle geliştirilmiş iki süreci birbirine gevşek bağlı (loosely-coupled) bir yapıda entegre ederek, gerçek zamanlı yapay zekâ destekli robotik kontrol uygulamaları için esnek ve güçlü bir çözüm sunmaktadır (Bruyninckx, 2001).

Şekil 3.39'da tüm aşamaların başarıyla sonuçlanmasının ardından oluşturulmuş nihai ROI görselinin kamera penceresi üzerindeki görüntüsü yer almaktadır.



Şekil 3.39 Tespit edilen kaynaklı nesnenin kamera ekranı üzerindeki çıktısı

3.6.2. Tek Kamerayla Stereo Görüntüleme: Öteleme ve Paralaks ile Derinlik Hesabı

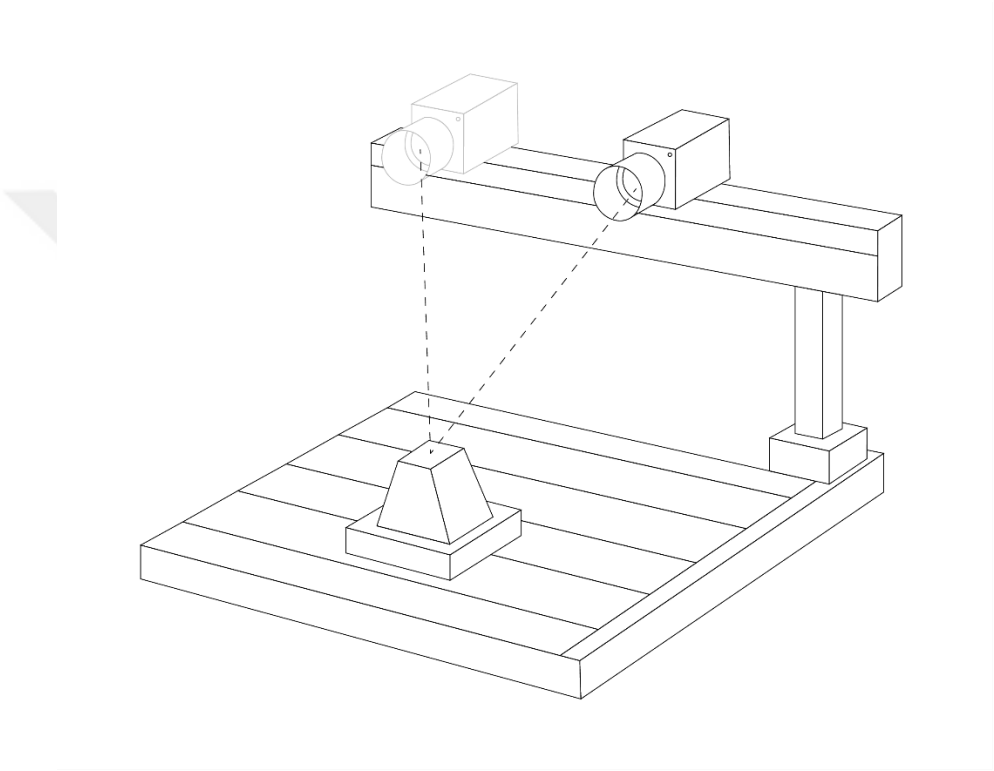
Sistemin üçüncü boyutu, yani derinliği algılaması, insan görüşünü taklit eden ancak bunu tek bir kamera ile gerçekleştiren yenilikçi bir harekete dayalı stereo yaklaşımı ile sağlanır. Stereo vizyonun temel prensibi, bir nesnenin bilinen iki farklı noktadan gözlemlendiğinde, bu iki görüntüdeki görünür konum kaymasından (paralaks veya disparite) yola çıkarak nesnenin mesafesini hesaplamaktır (Hartley ve & Zisserman, 2004). Bu çalışmada, iki ayrı kamera kullanmak yerine, kameranın monte edildiği CNC ekseninin hassas hareket kabiliyetinden faydalanılarak bu etki yaratılmıştır.

Bu metodoloji, bir önceki adımda kaynak noktasının tespiti tamamlandıktan hemen sonra başlar ve aşağıdaki adımları takip eder:

1. *İlk Konumda Tespit ve Referans Alma:* Yapay zekâ modeli, kaynak dikişini ve onun geometrik merkez noktasını ilk pozisyonda tespit eder. Bu merkez noktasının, metrik ölçekli ROI görüntüsü üzerindeki (x1,y1) piksel koordinatları referans olarak kaydedilir.
2. *Kontrollü Öteleme Hareketi:* Yazılım, referans koordinatları kaydettikten sonra, GRBL kontrolcüsüne bağlı olduğu eksen üzerinde yeterli paralaksı oluşturacak 100 mm

öteleme komutunu gönderir. Bu komut, kameranın bağlı olduğu X eksenini, önceden bilinen ve sabit bir temel mesafesi kadar yana öteler. Bu hareket, sanal bir "ikinci kamera" pozisyonu yaratır.

3. *İkinci Konumda Tespit:* Öteleme hareketi tamamlandıktan sonra, sistem anlık olarak yeni bir görüntü yakalar ve aynı kaynak noktasını bu yeni görüntü üzerinde tekrar tespit eder. Bu ikinci görüntüdeki merkez noktasının yeni (x2,y2) koordinatları da kaydedilir. Şekil 3.40'da bu yöntemin fiziksel bir tasviri yer almaktadır.



Şekil 3.40 Kamera 1. ve 2. Pozisyonun odaklandığı kaynaklı noktanın tasviri

4. *Disparite (Paralaks) Hesabı:* Derinlik hesaplamasının temel girdisi olan disparite (Δx), bu iki görüntüdeki merkez noktasının yatay eksendeki piksel koordinatları arasındaki mutlak fark olarak hesaplanır. ROI görüntüsünün 1px ≈ 1mm ölçeğine sahip olması sayesinde, bu piksel farkı doğrudan metrik bir kayma değeri olarak kabul edilebilir. Denklem 3.1'de gösterilmediktedir.

$$\Delta x = |x_1 - x_2| \tag{3.1}$$

5. *Derinlik (Yükseklik) Hesabı:* Son adımda, hesaplanan bu metrik disparite (Δx), sistemin daha önceden ampirik olarak kalibre edilmiş doğrusal modeline girdi olarak verilir ve

kaynak noktasının referans düzlemine olan Z derinliği (yüksekliği) Denklem 3.2'deki gibi hesaplanır.

$$Z = \frac{\Delta x - \mathrm{b}}{a} \tag{3.2}$$

Denklem 3.2'te yer alan a ve b katsayıları, sistemin stereo geometrisine özgü olan ve ön testler sırasında farklı derinliklerdeki nesneler için ölçülen disparite değerleri ile elde edilmiş kalibrasyon sabiteleridir. Bu yöntem, ek bir derinlik sensörüne veya ikinci bir kameraya ihtiyaç duymadan, robotun kendi hareket kabiliyetini bir ölçüm aracı olarak kullanarak maliyet-etkin ve bütünleşik bir 3B algılama çözümü sunar.

3.6.3. 3B Konum Verisinin Elde Edilmesi ve Arayüzde Doğrulanması

Bir önceki adımda hesaplanan Z derinlik bilgisi, ilk görüntüdeki X ve Y konum bilgisi ile birleştirilerek her bir kaynak noktası için tam bir üç boyutlu koordinat vektörü (X,Y,Z) oluşturulur. Bu aşama, ham algı verisinin, robotun hareket planlamasında kullanabileceği anlamlı ve eyleme dönüştürülebilir bir bilgiye dönüştürüldüğü son adımdır.

Oluşturulan bu 3B konum verisi, ana arayüzde yer alan ikinci bir tab ekranında operatöre sunulur. Bu panel, yalnızca bir doğrulama ekranı olmakla kalmaz, aynı zamanda nihai robot yörüngesinin planlandığı ve görselleştirildiği kritik bir adımdır. Arayüzde, tespit edilen kaynak poligonları ve bu poligonların merkez noktaları ROI görüntüsü üzerinde gösterilir. Buna ek olarak, sistem her bir poligonun geometrisini analiz ederek, poligonun iskeletini oluşturan bir orta hat işlem deseni hesaplar. Bu orta hat, poligonun karşılıklı köşe noktalarının aritmetik ortalamalarının alınmasıyla elde edilen bir dizi ardışık noktadan oluşur. Hesaplanan bu yörünge, görüntü üzerinde turkuaz renkli bir hat olarak görselleştirilir. Bu görselleştirmenin temel amacı, operatöre, makinenin yüzey işleme sırasında takip edeceği gerçek takım yolunu net bir şekilde göstermektir. Nihai G-kodları, bu turkuaz hattı oluşturan noktalar referans alınarak üretilir. Her bir kaynak hattı için hesaplanan Z (yükseklik) değeri de bu görselleştirmeyle birlikte, isminin yanında düzenlenebilir bir metin kutusunda listelenir. Bu bütüncül sunum, operatörün hem tespitin doğruluğunu hem hesaplanan yörüngeyi hem de derinlik değerini tek bir ekranda görerek süreci onaylamasına veya gerekli manuel düzeltmeleri yapmasına olanak tanır. Şekil 3.41'de bu panelin görüntüsü yer almaktadır.

Şekil 3.41 Tespit edilen kaynaklı bir parçanın derinlik hesabı sonrası çıkan tahmin değeri ve kaynak hattı

Bu arayüzün temel amacı, bir "insan-döngüde" (human-in-the-loop) denetim mekanizması sağlamaktır (Sheridan, 1992). Operatör, sistemi başlatmadan önce hesaplanan derinlik değerlerini gözden geçirebilir. Eğer aydınlatma koşulları, yüzey yansımaları veya olası bir yapay zekâ hatası nedeniyle bariz bir şekilde yanlış bir derinlik değeri hesaplanmışsa, operatör bu değeri doğrudan metin kutusu üzerinden manuel olarak düzeltme yetkisine sahiptir. Bu son kontrol ve düzeltme imkânı, sistemin algılama hataları karşısında dahi üretim sürecini durdurmadan, operatörün tecrübesiyle desteklenerek görevini başarıyla tamamlamasını sağlar. Tüm veriler operatör tarafından onaylandıktan sonra, bu nihai ve doğrulanmış 3B koordinat listesi, bir sonraki aşama olan hareket planlama ve G-kodu üretme modülüne aktarılır.

3.6.4. Hedef Koordinatların Hesaplanması: Kinematik ve Perspektif Ofsetlerin Entegrasyonu

Önceki adımlarda elde edilen 3B konum verisi (X,Y,Z), takım ucunun gitmesi gereken nihai hedefi tam olarak temsil etmez. Bu nedenle görüntü düzlemindeki bir piksel koordinatını, 4 eksenli bir robotun anlayacağı fiziksel bir komuta dönüştürmek gerekir. Bunu sağlamak için geliştirilen prensip iki temel geometrik problemin matematiksel olarak çözülmesini gerektirmiştir: Takımın, işlenecek yüzeye dik durmasını sağlayan A ekseni dönüşünün neden olduğu kinematik sapma ve hedef noktanın Z derinliğinin neden olduğu perspektif sapması. Bu

bölümde, ham konum verisinin bu iki ofset türüyle nasıl entegre edilerek nihai hedef koordinatlarının hesaplandığı detaylandırılmaktadır. Bu hesaplamalar, ana işlem iş parçacığı içerisinde, takım yolu üzerindeki her bir nokta için gerçek zamanlı olarak yürütülür.

Süreç, ana arayüz ekranında görselleştirilen turkuaz takım yolunu oluşturan her bir nokta için aşağıdaki adımları izler:

3.6.4.1. Açısal Hizalanmanın Belirlenmesi

İlk olarak, takımın iş parçasına her zaman iş parçasının sahip olduğu açıya paralel bir açıyla temas etmesi hedeflenir. Bu, özellikle eğrisel yollar üzerinde çalışırken yüzey kalitesini en üst düzeye çıkarmak için kritik bir adımdır. Bunu başarmak için yazılım, takım yolunu oluşturan mevcut nokta ile bir sonraki nokta arasındaki vektörün, yatay eksenle yaptığı açıyı hesaplar. Denklem 1.3'te gösterildiği gibi, bu açı iki nokta arasındaki dikey ve yatay bileşenlerin atan2 fonksiyonu ile arctanjantı alınarak bulunur. “atan2” fonksiyonu, standart arctan fonksiyonundan farklı olarak, vektörün bulunduğu kartezyen çeyreği otomatik olarak dikkate alarak 0 ile 360 derece arasında doğru ve kesin bir sonuç verir; Denklem 3.3’te gösterildiği gibidir.

$$\alpha_{\text{hedef}} = \text{atan2}(y_2 - y_1, x_2 - x_1) \tag{3.3}$$

Hesaplanan bu hedef açısı, A ekseninin o anki hedef dönüş açısı olarak belirlenir. Bu sayede takım ucu, hareket ettiği yörünge segmentine her zaman dik bir duruş kazanarak en verimli işleme pozisyonunu korur.

3.6.4.2. Kinematik Ofsetin Hesaplanması

A ekseni, takımı hedef açıya getirmek için döndüğünde, takım ucu fiziksel olarak bir yay üzerinde hareket eder ve bu durum, takım ucunun X-Y düzleminde istenen yörüngeden sapmasına neden olur. Kinematik ofset telafisinin amacı, bu sapmayı önceden hesaplayarak G-kodu komutuna eklemek ve böylece takım ucunun hedeflenen yörünge üzerinde kalmasını sağlamaktır. Bu hesaplama, Bölüm 3.4.3'te tanımlanan fiziksel makine parametrelerine dayanır.

Öncelikle, takım ucunun A ekseninin merkezine olan efektif dönme yarıçapı (r2), Denklem 3.4 kullanılarak hesaplanır.

$$r_2 = (l \cdot \cos(\theta)) + r_1 \tag{3.4}$$

Burada “l” takım boyunu, “θ” takımın montaj açısını ve r1 A ekseninin merkezi ile aparat montaj noktasının fiziksel yarıçapını temsil eder.

İkinci adımda, takım ucunun hedef açısına döndüğünde bulunacağı ideal konum, Denklem 3.5 ve Denklem 3.6'daki gibi hesaplanır.

$$X_{kin} = r_2 \backslash \cos(\alpha_{\{hedef\}}) \tag{3.5}$$

$$Y_{kin} = r_2 \backslash \sin(\alpha_{\{hedef\}}) \tag{3.6}$$

Son olarak, bu değerler kullanılarak nihai kinematik ofset vektörü Denklem 3.7'deki gibi bulunur.

$$\overrightarrow{O_{kın}} = \begin{bmatrix} -\frac{r_2}{2} + X_{kin} - C_{cal} \\ Y_{kin} \end{bmatrix} \tag{3.7}$$

Bu denklemdeki “C_{cal}” terimi, sistemin mekanik montajından kaynaklanan ve deneysel olarak belirlenmiş sabit bir kalibrasyon ofsetini (kodda 32 mm) temsil eder.

3.6.4.3. Perspektif Ofsetin Hesaplanması

İkinci telafi edilmesi gereken sapma, kameranın perspektifinden kaynaklanır. Bir noktanın Z eksenindeki derinliği arttıkça, o noktanın 2B görüntü düzlemindeki izdüşümü merkezden dışa doğru kayar. Bu kayma miktarı, noktanın derinliğine (Z) ve kameranın geometrik konumuna (Hcam,Dcam,Wcam) bağlıdır. Y ekseni ve X ekseni için perspektif ofseti (Oy ve Ox), Denklem 3.8 ve Denklem 3.9 ile hesaplanır.

$$O_y = \cot\left(arctan\left(\frac{H_{cam}}{W_{cam}+\left(\frac{H_{img}}{2}-p_y\right)}\right)\right) * Z \tag{3.8}$$

$$O_x = \cot\left(arctan\left(\frac{H_{cam}}{|D_{cam}-p_x|}\right)\right) * Z \tag{3.9}$$

Bu denklemlerde px ve py, hedef noktanın ROI görüntüsü üzerindeki piksel koordinatlarıdır. H_{img} ise hedef ROI görüntüsünün yüksekliğidir. Aynı şekilde bu değer çalışma alanının yüksekliği anlamına da gelmektedir. Buradan da Denklem 3.10’da gösterilen perspektif ofset matrisi çıkmış olur.

$$\overrightarrow{O_{per}} = \begin{bmatrix} O_x \\ O_y \end{bmatrix} \quad (3.10)$$

3.6.4.4. Nihai Hedef Koordinatların Entegrasyonu

Son adımda, takım yolundaki bir piksel noktasının gerçek dünyadaki karşılığı olan nihai hedef G-kodu koordinatı her iki ofset vektörünün de eklenmesiyle bulunur. Bu bütünleşik hesaplama Denklem 3.11'de gösterilmiştir.

$$\overrightarrow{P_{tar}} = \overrightarrow{P_{pix}} + \overrightarrow{O_{kın}} + \overrightarrow{O_{per}} \quad (3.11)$$

Burada Ppiksel= "[px,py]T" olarak tanımlanmıştır. Bu çok aşamalı koordinat dönüşümü, robotik sistemlerde sensör verisinin (görüntü) makine hareketine dönüştürülmesindeki temel zorlukları ele alan bir yaklaşımdır (Spong ve ark., 2006). Elde edilen bu nihai hedef vektörünün X ve Y bileşenleri, hesaplanan Z derinliği ve A açısı ile birleştirilerek "G90 X... Y... Z... A..." formatındaki mutlak pozisyon komutu oluşturulur ve GRBL kontrolcüsüne gönderilir.

3.6.5. Otonom Yüzey İşleme: Hareket ve Prosesin Yürütülmesi

Önceki adımlarda, iş parçası üzerindeki hedef bölgeler tespit edilmiş, bu bölgelerin 3B uzaydaki tam konumları hesaplanmış ve bu konumlara ulaşmak için gereken takım yolu sanal olarak planlanmıştı. Bu bölüm, tüm bu hesaplanmış verilerin fiziksel eyleme dönüştürüldüğü, yani otonom yüzey işleme prosesinin başlatıldığı ve yürütüldüğü nihai metodolojiyi açıklamaktadır. Bu süreç, ana işlemden sorumlu olan iş parçacığı tarafından yönetilir ve tüm sistem bileşenlerinin senkronize bir şekilde çalışmasını gerektirir.

Proses, operatörün arayüz üzerinden "İşlemi Başlat" komutunu vermesiyle tetiklenir. Ana iş parçacığı, ana arayüz ekranında operatör tarafından onaylanmış olan kaynak hatlarının listesini alarak işlem döngüsünü başlatır. Bu döngü, listedeki her bir kaynak hattı için aşağıdaki adımları sırasıyla uygular

3.6.5.1. Hedef Bölgeye Güvenli Yaklaşma

Her yeni kaynak hattına geçmeden önce, çarpışmaları önlemek ve yumuşak bir geçiş sağlamak için çok adımlı bir yaklaşma stratejisi izlenir. İlk olarak, bir önceki konum ile hedef kaynak hattının başlangıç noktası arasındaki Z yükseklik farkı kontrol edilir. Takımın yanal

hareket sırasında iş parçasına sürtünmesini engellemek amacıyla, Z ekseni daima daha yüksek olan pozisyona öncelik verilerek hareket ettirilir. Genellikle, takım ucu önce hedef noktanın birkaç milimetre üzerinde güvenli bir "geçiş yüksekliğine" kaldırılır. Ardından, Bölüm 3.6.4'te hesaplanan hem kinematik hem de perspektif ofsetlerle tamamen telafi edilmiş hedef başlangıç noktasına X, Y ve A eksenlerinin eş zamanlı hareketiyle ilerlenir. Bu başlangıç pozisyonuna ulaşıldıktan sonra Z ekseni, Bölüm 3.6.6'da daha detaylı anlatılan PID kontrol prosesiyle alçaltılarak takım, yüzeye temas ettirilir.

3.6.5.2. Yüzey İşleme Deseninin Takip Edilmesi

Takım yüzeye temas ettikten sonra, asıl yüzey işleme süreci başlar. Yazılım, ana arayüz ekranında turkuaz renkle görselleştirilen orta hat takım yolunu oluşturan nokta dizisini takip eder. Döngü, bu dizideki her bir nokta için nihai hedef koordinatlarını hesaplar ve ilgili mutlak pozisyon G-kodu komutunu GRBL kontrolcüsüne gönderir. Bu, takımın kaynak hattının geometrisini hassas bir şekilde takip etmesini sağlar. Yüzey kalitesini artırmak ve homojen bir işleme sağlamak amacıyla, bu takım yolu üzerinde, kullanıcı tarafından arayüzde önceden tanımlanmış olan tekrar sayısı kadar ileri-geri hareket gerçekleştirilir. Bu çoklu geçiş deseni, tek geçişte kaldırılamayan yüzey pürüzlerini ortadan kaldırarak daha pürüzsüz bir sonuç elde edilmesini hedefler. Bu tür programlanmış takım yollarının oluşturulması ve uygulanması, modern Bilgisayar Destekli İmalat (CAM) sistemlerinin temelini oluşturur (Lee, 1999).

3.6.5.3.Proses Kontrolü ve Güvenlik Mekanizmaları

İşlem sırasında sistemin kararlılığını sağlamak için çeşitli kontrol mekanizmaları aktif olarak çalışır.

- *Arabellek Yönetimi:* Yazılım, GRBL kontrolcüsünün komut arabelleğinin durumunu (Bf parametresi) sürekli olarak izler. Eğer arabellek dolmaya yaklaşırsa, yeni G-kodu komutlarının gönderimini anlık olarak duraklatır ve arabellek boşaldığında gönderime devam eder. Bu akış kontrolü, özellikle çok sayıda kısa ve hızlı hareket içeren karmaşık yörüngelerde, makinenin takılmadan ve sarsıntısız bir şekilde hareket etmesini garanti eder.
- *Operatör Müdahalesi:* Operatör, işlem sırasında herhangi bir anda arayüz üzerinden süreci duraklatma veya tamamen durdurma yetkisine sahiptir. "Duraklat" komutu verildiğinde GRBL'e anlık besleme durdurma sinyali (Feed Hold) gönderilir. "Durdur"

komutu ise acil durum sıfırlama komutu (Reset) göndererek tüm hareketleri anında keser ve sistemi güvenli bir duruma alır.

- *İşlemin Tamamlanması ve Geri Çekilme:* Bir kaynak hattı üzerindeki tüm geçişler tamamlandığında, takım Z ekseninde dikey olarak güvenli bir yüksekliğe geri çekilir. Ardından, bir sonraki kaynak hattına geçmek üzere döngünün başına döner. Tüm kaynak hatları işlendikten sonra, makine programın sonunda tanımlanmış olan genel başlangıç veya "park" pozisyonuna geri dönerek görevi tamamlar.
-

3.6.6. PID Kontrolü ile Gerçek Zamanlı Yüzey Takibi ve Basınç Kontrolü

Önceki adımlarda hesaplanan 3B takım yolu, makro düzeyde bir yörünge tanımlar. Ancak, kaynak dikişleri gibi endüstriyel yüzeyler mikro düzeyde dalgalanmalara, pürüzlere ve beklenmedik geometri değişikliklerine sahiptir. Sadece önceden planlanmış bir yörüngeyi takip etmek (açık döngü kontrol), takımın bazı yerlerde yüzeye hiç temas etmemesine, bazı yerlerde ise aşırı basınç uygulayarak hem iş parçasına hem de takıma zarar vermesine neden olabilir. Benzer şekilde yüzey tıraşlandıkça yüzey üzerindeki pürüzler giderilir ve yükseklik değeri son hesaplanan değere göre mikro ölçeklerde azalmaya başlar. Bu noktada makinenin hem hedef yolu izlemesi hem de mevcut Z pozisyonunu her hareket iterasyonunda güncellemesi gerekmektedir. Diğer bir deyişle yüksek kalitede ve homojen bir yüzey işleme elde etmek için, sistemin anlık yüzey değişikliklerini "hissetmesi" ve Z ekseni pozisyonunu buna göre gerçek zamanlı olarak dinamik bir şekilde ayarlaması gerekir. Bu nedenle bu çalışmada hedeflenen hassasiyetin yakalanması adına Oransal (Proportional) Kontrol Sistemi tasarlanmış ve uygulanmıştır. Bu yapı, genel olarak PID (Oransal-İntegral-Türevsel) denetleyicilerin bir alt kümesi olup, endüstriyel süreçlerde kararlılığı ve hızlı yanıtı sağlamak için yaygın olarak kullanılır (Åström ve & Hägglund, 1995). Sistem de bu özellikten faydalanarak hedef çıktıya hızlı ve osilasyona uğramadan yaklaşır.

3.6.6.1. Sensör Verisinin Fiziksel Anlama Dönüştürülmesi: Transfer Fonksiyonu

Kontrol döngüsünün ilk ve en kritik adımı, Bölüm 3.1.3'te açıklanan Hall effect sensöründen gelen ham veriyi, kontrolcünün anlayabileceği anlamlı bir fiziksel birime dönüştürmektir. Sensörün elektriksel çıktısı, mıknatısa olan mesafeyle (ve dolayısıyla yay sıkışmasıyla) doğrudan doğrusal bir ilişkiye sahip değildir; bu ilişki, Grafik 1.0'da da görüleceği gibi, eksponansiyel bir karakteristiğe sahiptir. Bu non-lineerliği ortadan kaldırmak

ve sensör okumasını, fiziksel sıkışma miktarıyla orantılı, tekrarlanabilir bir değere dönüştürmek için deneysel bir transfer fonksiyonu türetilmiştir. Transfer fonksiyonu çıkışında sıkışma miktarını hedef değere dönüştürmek için kontrollü Z hareketleri yapılması gerektiğinden transfer fonksiyonu metrik çıktı verecek şekilde modellenmiştir. Bu sayede hatanın birimi mm olarak ele alınabilmiş ve GRBL sistemine gönderilen son kod parçasında Z değerine artı olarak eklenebilmiştir.

Bu fonksiyonun elde edilmesi, kontrollü bir veri toplama ve modelleme sürecini içermiştir:

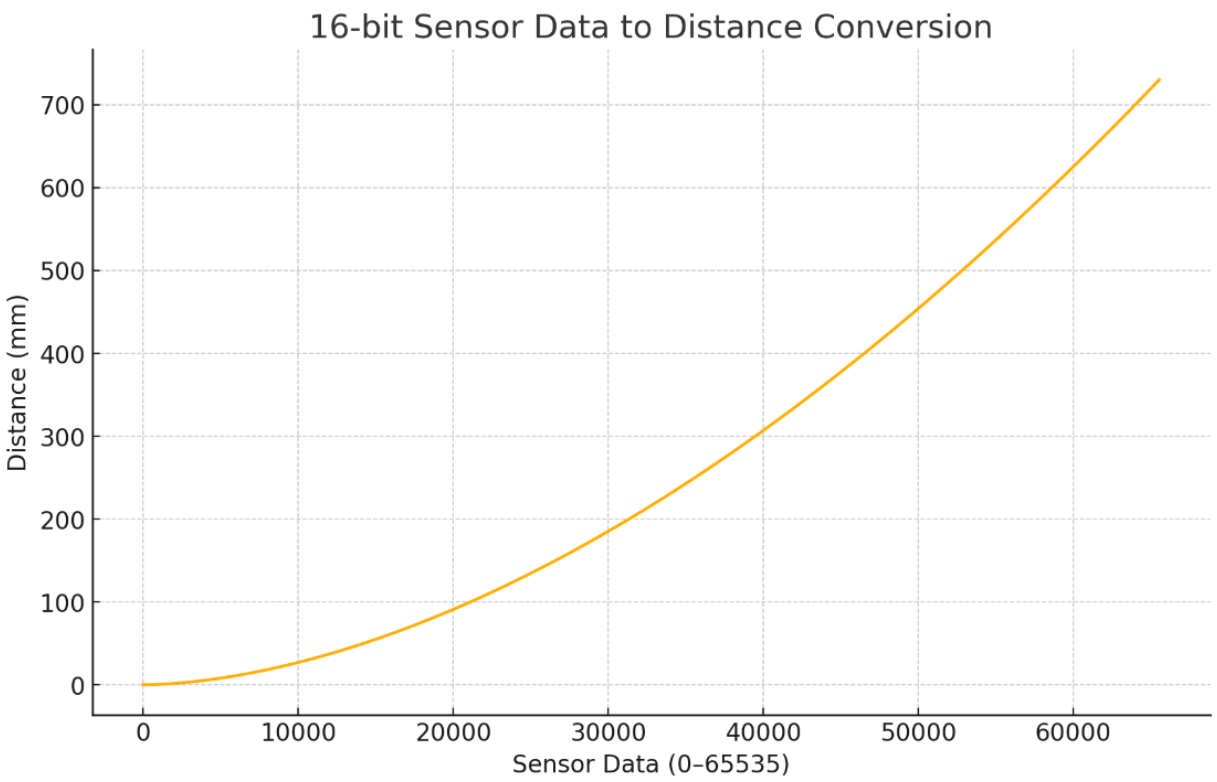
- *Deneysel Veri Toplama:* İlk olarak, sistemin Z ekseni, yüksek hassasiyetli G-kodu komutları kullanılarak 0,1 mm'lik çok küçük ve hassas adımlarla hareket ettirilmiştir. Her bir adımda yayın sıkışma miktarı değiştirilmiş ve bu adıma karşılık gelen ham sensör çıktısı kaydedilmiştir. Bu süreç, sensörün tüm çalışma aralığını kapsayacak şekilde yaklaşık 700 veri noktasından (sıkışma miktarı, sensör çıktısı) oluşan bir nokta bulutu oluşturmuştur. Buna ek olarak yayın hareket alanının yüksekliği de daha hassas bir şekilde sınırlandırılabilmiştir.
- *MATLAB ile Eğri Uydurma:* Elde edilen bu ayrık veri seti, matematiksel modelleme ve analiz için MATLAB ortamına aktarılmıştır. MATLAB'ın Curve Fitting Toolbox'ı kullanılarak, veri noktalarının eksponansiyel davranışını en iyi şekilde temsil eden bir üstel fonksiyon modeli oluşturulmuştur. Bu araç, veri setindeki gürültüyü en aza indirerek ve "En Küçük Kareler (least squares)" gibi istatistiksel regresyon yöntemleri kullanarak en uygun matematiksel denklemi ve katsayılarını türetir (Montgomery ve ark., 2012).
- *Veri Ölçekleme ve Normalizasyon:* MATLAB'ın ürettiği ilk denklem, sensörden gelen ham verinin orijinal bit çözünürlüğüne göreydi. Ancak yazılımın ilerleyen aşamalarında, gelen verinin standart bir 16-bit tamsayı aralığına ölçeklenmesi ihtiyacı doğmuştur. Tüm veri toplama ve eğri uydurma sürecini tekrarlamak yerine, mevcut matematiksel modele pratik bir adaptasyon yapılmıştır. Denklemdeki C_2 (128) sabiti, 16-bit'e ölçeklenmiş veriyi, orijinal formülün beklediği daha düşük çözünürlüklü aralığa geri normalize etmek amacıyla sonradan eklenmiştir. Bu iki aşamalı normalizasyon yaklaşımı, mevcut matematiksel modelin farklı bir veri ölçeğine pratik bir şekilde adapte edilmesini sağlamıştır. Yüksek çözünürlüklü yeni sensör verisinin çözünürlüğü bilgisayar tarafında indirgendiğinde "float" ve "double" gibi ondalıklı sayı

değişkenlerinde tutulabildiği için bu normalizasyon, sensör çözünürlüğüne olumsuz bir etkide bulunmamıştır.

Bu sürecin sonunda, ham sensör okumasını (v) alıp onu kontrol döngüsünde kullanılabilecek doğrusallaştırılmış metrik bir değere dönüştüren nihai transfer fonksiyonu, Denklem 3.12'deki gibi elde edilmiştir.

$$T(v) = \frac{(v^{1.755}.C_1)}{C_2} \tag{3.12}$$

Bu denklemde v ham sensör okumasını, C_1 (0,0003294) ve C_2 (128) ise bu deneysel süreçle belirlenmiş kalibrasyon ve normalizasyon sabitelerini temsil eder. Denklemin grafiği Şekil 3.42'de gösterildiği gibidir.



Şekil 3.42 Hall Effect Sensörü Karakteristik Eğrisi ve MATLAB ile Uydurulan Fonksiyon Modeli

4. ARAŞTIRMA BULGULARI VE TARTIŞMA

Bu bölümde, önceki bölümde ayrıntılı olarak açıklanan metodolojilerin deneysel olarak doğrulanması ve sistemin genel performansının değerlendirilmesi amacıyla gerçekleştirilen testlerin sonuçları sunulmakta ve tartışılmaktadır. Gerçekleştirilen deneyler, sistemin temel yeteneklerini; yani 3B konum belirleme doğruluğunu, yüzey takip hassasiyetini, yapay zekâ modelinin tespit başarımını ve tüm bileşenlerin bütünleşik performansını ölçmeyi hedeflemektedir. Elde edilen bulgular, geliştirilen mimarinin endüstriyel uygulamalardaki potansiyelini ve literatürdeki benzer çalışmalara kıyasla getirdiği özgün katkıları ortaya koymaktadır.

4.1. 3B Konum Belirleme Testleri ve Sonuçları

Bu bölümde, geliştirilen tek kameralı ve hareketli paralaks tabanlı üç boyutlu konum belirleme sisteminin performansı, hassasiyeti ve mevcut sınırlılıkları, sistematik bir veri toplama süreci ve detaylı hata analizi yöntemleriyle incelenmiştir. Amaç, sistemin Z eksenindeki (derinlik) ölçüm doğruluğunu ve bu eksendeki hataların, nihai X ve Y konum koordinatlarına olan etkisini (hata yayılımı) ortaya koymaktır.

4.1.1. Deneysel Kurulum ve Sistematik Veri Toplama Yöntemi

Sistemin performansını çalışma alanı genelinde kapsamlı bir şekilde haritalandırmak amacıyla sistematik bir yaklaşım benimsenmiştir. Bu kapsamda;

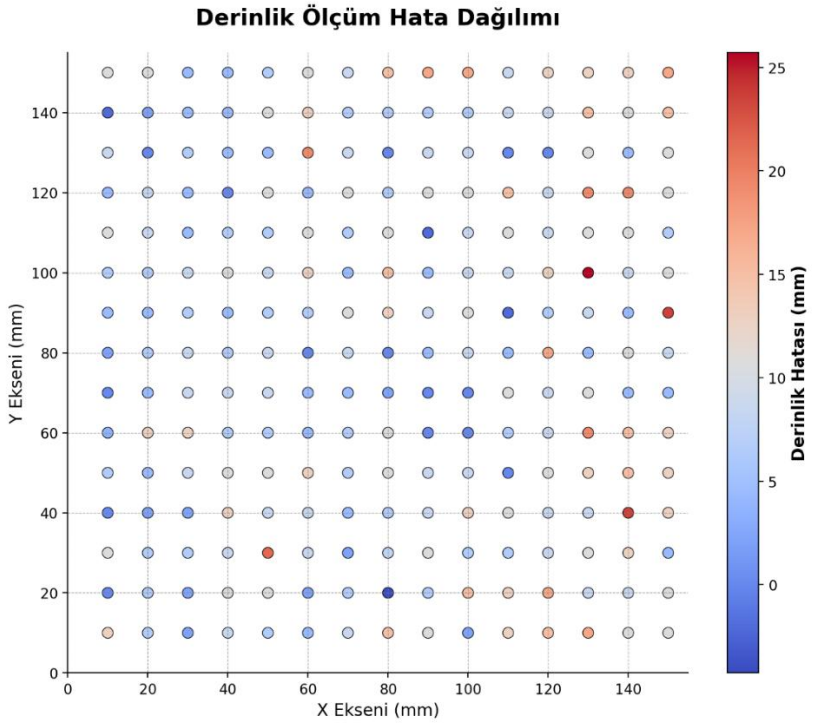
1. *Referans Nesne:* Yüksekliği hassas bir şekilde ölçülmüş (40 mm) ve geometrisi bilinen bir referans nesnesi kullanılmıştır.
2. *Test Alanı:* Çalışma tablası üzerinde 150 mm x 150 mm'lik bir alan tanımlanmıştır.
3. *Veri Toplama:* Bu alan içerisinde, X ve Y eksenlerinde 10 mm'lik adımlarla ilerlenerek toplam 15x15 farklı noktadan veri toplanmıştır.

Her bir noktada, sistem tam algılama döngüsünü (görüntü yakalama, paralaks için X ekseninde hareket, ikinci görüntü yakalama, derinlik ve ofset hesaplama) icra etmiş ve nesnenin (X, Y, Z) koordinatlarını tahmin etmiştir. Bu yöntem, sadece birkaç noktadaki ortalama hatayı bulmak yerine, hatanın çalışma alanı üzerindeki dağılımını ve karakteristiğini anlamamızı sağlamıştır.

4.1.2. Z Ekseni (Derinlik) Ölçüm Sonuçları ve Analizi

Toplanan 225 veri noktasının her biri için, sistem tarafından ölçülen derinlik ile referans nesnesinin gerçek derinliği arasındaki fark alınarak bir hata değeri hesaplanmıştır. Bu hata değerlerinin çalışma alanı üzerindeki dağılımı Şekil 4.1'de görselleştirilmiştir. Şekil 4.1'deki

renk haritası, hatanın hem yönünü hem de büyüklüğünü göstermektedir. Coolwarm renk haritası kullanılarak, pozitif hatalar (sistemin nesneyi olduğundan daha yakın ölçtüğü durumlar) kırmızı tonlarıyla, negatif hatalar (nesneyi olduğundan daha uzak ölçtüğü durumlar) ise mavi tonlarıyla temsil edilmiştir. Rengin doygunluğu, hatanın mutlak değerinin büyüklüğü ile doğru orantılıdır. Sıfıra yakın, yani daha doğru ölçümler ise beyaz renkle gösterilmektedir.



Şekil 4.1 Derinlik analizinin konumlara göre hata dağılımı

Şekil 4.1 incelendiğinde, aşağıdaki kritik bulgulara ulaşılmıştır:

- *Sistematik Hata Paterni:* Gözlemlenen hatalar, rastgele bir dağılım göstermek yerine, oldukça **sistematik ve periyodik bir yapıya** sahiptir. Bu durum, hatanın kaynağının anlık gürültü veya tesadüfi etkenlerden ziyade, sistemin modellemesindeki temel bir yanlılıktan kaynaklandığını göstermektedir.
- *X Ekseni Bağımlılığı:* En belirgin desen, hatanın nesnenin Y konumundan çok, X konumuna (görüntü üzerindeki yatay piksel konumu) bağlı olarak dramatik bir şekilde değişmesidir. Grafik üzerindeki net dikey bantlar, bu güçlü korelasyonun kanıtıdır.
- *Doğrusal Modelin Yetersizliği:* Gözlemlenen periyodik (tekrarlayan) hata paterni, derinlik hesabında kullanılan ve tezin önceki bölümlerinde Denklem 3.2 olarak belirtilen doğrusal modelin, gerçek fiziksel ilişkiyi modellemede yetersiz kaldığını ortaya koymaktadır. Gerçekte, piksel kayması ile derinlik arasındaki ilişkinin, çalışma

alanı genelinde doğrusal olmadığı anlaşılmaktadır. Bu durum, sistemin doğruluğundaki en temel sınırlılığı oluşturmaktadır.

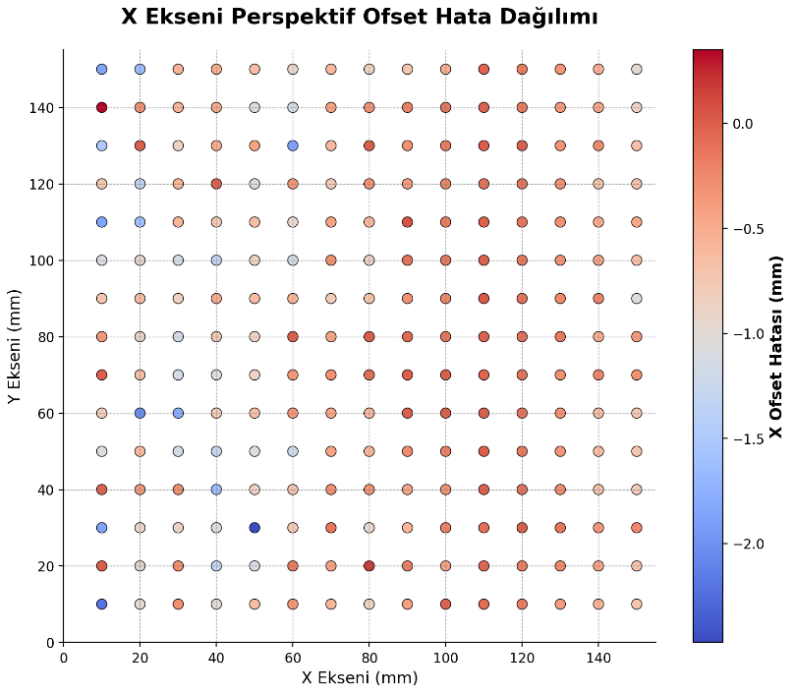
4.1.3. Hatanın X ve Y Eksenlerindeki Konumlandırmaya Etkisi (Hata Yayılımı Analizi)

Perspektif bozulmasını telafi etmek için kullanılan ve Denklem 3.8 ile Denklem 3.9'da formüle edilen ofset hesaplamaları, doğrudan derinlik (z) değerini bir girdi olarak kullanır. Dolayısıyla Z ekseninde tespit edilen bu sistematik hatanın, nihai X ve Y koordinatlarını nasıl etkilediğini anlamak kritik öneme sahiptir. Çünkü matematiksel modellere dayalı kinematik ve perspektif dönüşümler oranlanmış bir görüntü üzerinde hatalı sonuç veremezler. Tüm girdileri sabit ve bilinen denklem modelleri olduklarından, doğru girdiler için doğru çıktılar vermek zorundadırlar. Fakat tahmine dayalı bir çıktının bağımsız parametre olarak kullanıldığı perspektif dönüşüm denklemleri, değişkendeki hatadan etkilenir ve bu hatayı daha yüksek veya düşük bir çarpanla çıktısına yansıtabilir. Dolayısıyla, z değerindeki bir hata, kaçınılmaz olarak hesaplanan ofset değerlerine ve nihai X, Y konumlarına yayılacaktır.

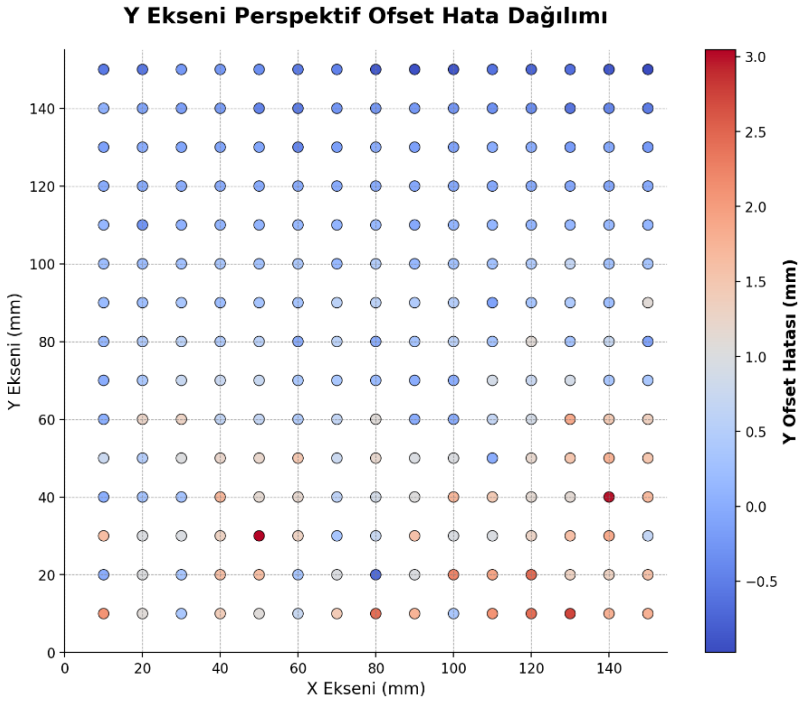
Bu hata yayılımını analiz etmek için, her bir nokta için iki farklı ofset değeri hesaplanmıştır:

1. **İdeal Ofset:** Gerçek derinlik değeri (z = 40 mm) kullanılarak hesaplanan teorik ofset.
2. **Ölçülen Ofset:** Sistemin ölçtüğü hatalı derinlik değeri kullanılarak hesaplanan ofset.

Bu iki değer arasındaki fark, "Ofset Hatası" olarak tanımlanmıştır. Bu hatanın X ve Y bileşenleri için dağılım haritaları sırasıyla Şekil 4.2 ve Şekil 4.3'te sunulmuştur.



Şekil 4.2 Tahmin derinliğine bağlı hesaplanan ofset değerlerinin X eksenindeki hata dağılımı



Şekil 4.3 Tahmin derinliğine bağlı hesaplanan ofset değerlerinin Y eksenindeki hata dağılımı

Bu iki grafik incelendiğinde yapılan çıkarımlar şunlardır:

- *Hata Karakteristiğinin Korunumu:* Her iki ofset hata haritası da Şekil 4.1'deki ana derinlik hata haritasının temel karakteristiğini (dikey bantlaşma ve periyodik yapı) aynen yansıtmaktadır. Bu, ofset hatalarının yeni ve bağımsız bir problem olmadığını, aksine doğrudan derinlik hatasının bir sonucu olduğunu matematiksel olarak doğrulamaktadır.
- *Yayılımın Kanıtı:* Denklem 3.8 ve 3.9'da formüle edilen ofset hesaplama denklemleri, derinlik hatasını konuma bağlı bir katsayıyla çarparak X ve Y eksenlerine taşımaktadır. Şekil 4.2'deki X ofset hata haritası, bu durumun en net kanıtı olarak neredeyse tamamen Şekil 4.1'deki derinlik hatası desenini kopyalamaktadır. Öte yandan Şekil 4.3 (Y Ofset Hatası), daha katmanlı bir bilgi sunar: Grafikte, Z ekseninden miras kalan baskın dikey bantlaşma deseninin üzerine, Y ekseni boyunca uzanan yatay bir renk gradyanının bindiği açıkça görülmektedir. Bu ikincil desen, Y ofsetini hesaplayan katsayının, beklendiği gibi nesnenin dikey piksel konumuna duyarlı olmasından kaynaklanır. Kısacası, bu grafikler derinlik hatasının X ve Y konumlarına olan yayılımını ve bu yayılımın her eksen için farklı bir katsayıyla gerçekleştiğini net bir şekilde ortaya koymaktadır.

4.1.4. Nicel Sonuçların Değerlendirilmesi

Sistematik analizin ortaya koyduğu desenlerin yanı sıra, sistemin genel performansını nicel olarak özetlemek için 225 ölçüm noktasından elde edilen veriler istatistiksel olarak incelenmiştir. Sonuçlar Çizelge 4.1'de özetlenmiştir.

Çizelge 4.1 Görüntü üzerinde sistematik olarak yapılan ölçümlerin çıktısı

Hesaplanmış X (mm)	Hesaplanmış Y (mm)	Derinlik Tahmini Z (mm)	Derinlik Hatası (mm)	X Perspektif Hatası (mm)	Y Perspektif Hatası (mm)
147,205	289,189	27,131	12,869	-2,221	2,099
163,111	293,259	33,555	6,445	-0,966	1,089
176,448	294,888	37,837	2,163	-0,283	0,370
185,927	289,763	31,413	8,587	-1,007	1,408
198,555	292,185	33,555	6,445	-0,639	1,079
213,445	294,615	35,696	4,304	-0,335	0,735
221,216	291,902	31,413	8,587	-0,574	1,434
226,750	287,832	24,989	15,011	-0,885	2,420
241,539	289,481	29,272	10,728	-0,406	1,755
262,046	294,888	37,837	2,163	-0,018	0,370
263,793	289,189	27,131	12,869	-0,077	2,099
276,343	287,832	24,989	15,011	-0,179	2,420
283,517	286,475	22,848	17,152	-0,380	2,731
304,352	291,610	29,272	10,728	-0,557	1,787
313,933	291,610	29,272	10,728	-0,704	1,787
152,179	284,270	39,979	0,021	-0,004	0,003
163,111	284,666	33,555	6,445	-0,966	1,010
179,698	284,053	37,837	2,163	-0,273	0,337
182,985	279,899	29,272	10,728	-1,303	1,608
194,696	280,964	29,272	10,728	-1,123	1,624
214,371	282,970	37,837	2,163	-0,166	0,334
221,111	282,518	33,555	6,445	-0,432	0,990
238,007	286,869	44,261	-4,261	0,183	-0,681
246,888	282,518	33,555	6,445	-0,194	0,990
247,853	279,391	24,989	15,011	-0,432	2,239
265,913	279,650	27,131	12,869	-0,038	1,924
275,114	274,921	22,848	17,152	-0,174	2,448
288,584	281,209	31,413	8,587	-0,253	1,303
303,555	282,278	31,413	8,587	-0,436	1,316
312,868	277,770	29,272	10,728	-0,688	1,575
145,723	280,964	29,272	10,728	-1,874	1,624
164,185	282,518	33,555	6,445	-0,956	0,990
173,852	281,444	33,555	6,445	-0,867	0,980
184,858	281,209	31,413	8,587	-1,020	1,303
187,295	274,393	18,565	21,435	-2,471	3,044

208,384	282,278	31,413	8,587	-0,731	1,316
224,123	282,970	37,837	2,163	-0,136	0,334
181,824	240,540	32,261	7,739	-0,953	0,725
230,893	277,770	29,272	10,728	-0,569	1,575
246,888	281,444	33,555	6,445	-0,194	0,980
257,629	281,444	33,555	6,445	-0,095	0,980
268,267	279,070	31,413	8,587	-0,003	1,277
276,671	279,899	29,272	10,728	-0,133	1,608
287,111	276,470	27,131	12,869	-0,351	1,866
307,300	282,749	35,696	4,304	-0,242	0,663
306,627	277,022	39,848	0,152	-0,008	0,022
150,443	272,134	37,837	2,163	-0,363	0,300
175,364	271,051	37,837	2,163	-0,286	0,297
180,062	267,991	27,131	12,869	-1,617	1,710
196,621	268,376	31,413	8,587	-0,876	1,145
208,384	271,584	31,413	8,587	-0,731	1,185
220,996	273,039	35,696	4,304	-0,289	0,603
231,851	271,777	33,555	6,445	-0,333	0,891
230,945	259,938	31,413	8,587	-0,455	1,042
250,015	267,991	27,131	12,869	-0,331	1,710
267,090	269,253	29,272	10,728	-0,014	1,445
277,891	268,376	31,413	8,587	-0,121	1,145
290,723	267,307	31,413	8,587	-0,279	1,132
289,043	263,753	16,424	23,576	-0,709	2,989
310,428	265,871	27,131	12,869	-0,780	1,671
150,222	256,740	33,555	6,445	-1,084	0,753
165,977	255,779	35,696	4,304	-0,627	0,497
172,026	257,683	31,413	8,587	-1,177	1,014
184,050	255,413	29,272	10,728	-1,287	1,232
196,825	254,348	29,272	10,728	-1,091	1,216
202,319	256,332	27,131	12,869	-1,208	1,495
221,111	256,740	33,555	6,445	-0,432	0,753
230,893	253,284	29,272	10,728	-0,569	1,200
245,811	254,475	31,413	8,587	-0,272	0,975
252,227	255,544	31,413	8,587	-0,193	0,988
271,887	259,240	39,979	0,021	-0,000	0,003
277,736	253,284	29,272	10,728	-0,149	1,200
286,051	257,392	27,131	12,869	-0,332	1,515
297,446	257,232	24,989	15,011	-0,631	1,763
307,249	257,392	27,131	12,869	-0,722	1,515
109,821	182,940	36,501	3,499	-0,791	0,040
157,804	243,613	27,131	12,869	-2,026	1,261
170,523	245,733	27,131	12,869	-1,792	1,300
184,592	244,925	33,555	6,445	-0,768	0,644
196,407	248,148	33,555	6,445	-0,659	0,674
210,208	249,306	35,696	4,304	-0,355	0,457

220,037	247,074	33,555	6,445	-0,442	0,664
228,764	245,831	29,272	10,728	-0,601	1,086
246,857	251,622	39,979	0,021	-0,001	0,002
261,004	248,357	39,979	0,021	-0,000	0,002
269,444	248,148	33,555	6,445	-0,013	0,674
277,891	246,989	31,413	8,587	-0,121	0,883
279,113	242,245	20,707	19,293	-0,306	1,853
298,501	242,460	24,989	15,011	-0,654	1,447
309,368	247,853	27,131	12,869	-0,761	1,339
148,914	237,475	39,979	0,021	-0,004	0,002
162,741	236,360	35,696	4,304	-0,647	0,377
173,095	234,157	31,413	8,587	-1,164	0,726
182,719	235,227	31,413	8,587	-1,046	0,739
196,621	234,157	31,413	8,587	-0,876	0,726
209,129	236,360	35,696	4,304	-0,362	0,377
220,996	234,203	35,696	4,304	-0,289	0,364
236,042	234,211	37,837	2,163	-0,099	0,183
249,033	237,475	39,979	0,021	-0,001	0,002
259,916	238,563	39,979	0,021	-0,000	0,002
266,025	233,056	29,272	10,728	-0,030	0,890
276,822	235,227	31,413	8,587	-0,108	0,739
286,253	233,056	29,272	10,728	-0,280	0,890
305,142	233,124	35,696	4,304	-0,228	0,357
315,931	235,281	35,696	4,304	-0,295	0,371
152,610	225,543	37,837	2,163	-0,357	0,156
164,185	223,444	33,555	6,445	-0,956	0,446
173,095	222,395	31,413	8,587	-1,164	0,581
185,666	223,444	33,555	6,445	-0,758	0,446
197,690	221,325	31,413	8,587	-0,862	0,568
226,180	225,504	39,979	0,021	-0,001	0,002
231,909	222,395	31,413	8,587	-0,443	0,581
249,033	226,592	39,979	0,021	-0,001	0,002
258,754	223,415	35,696	4,304	-0,057	0,298
256,504	221,325	31,413	8,587	-0,141	0,568
270,621	223,415	35,696	4,304	-0,016	0,298
271,962	220,298	22,848	17,152	-0,097	1,110
292,197	223,415	35,696	4,304	-0,149	0,298
301,158	222,410	29,272	10,728	-0,508	0,727
238,900	164,913	31,413	8,587	-0,357	-0,124
149,795	207,233	35,696	4,304	-0,727	0,198
162,741	214,784	35,696	4,304	-0,647	0,245
176,000	212,703	33,555	6,445	-0,847	0,347
187,553	211,548	35,696	4,304	-0,495	0,225
199,629	209,481	33,555	6,445	-0,630	0,317
208,222	210,555	33,555	6,445	-0,550	0,327
217,053	212,828	29,272	10,728	-0,781	0,580

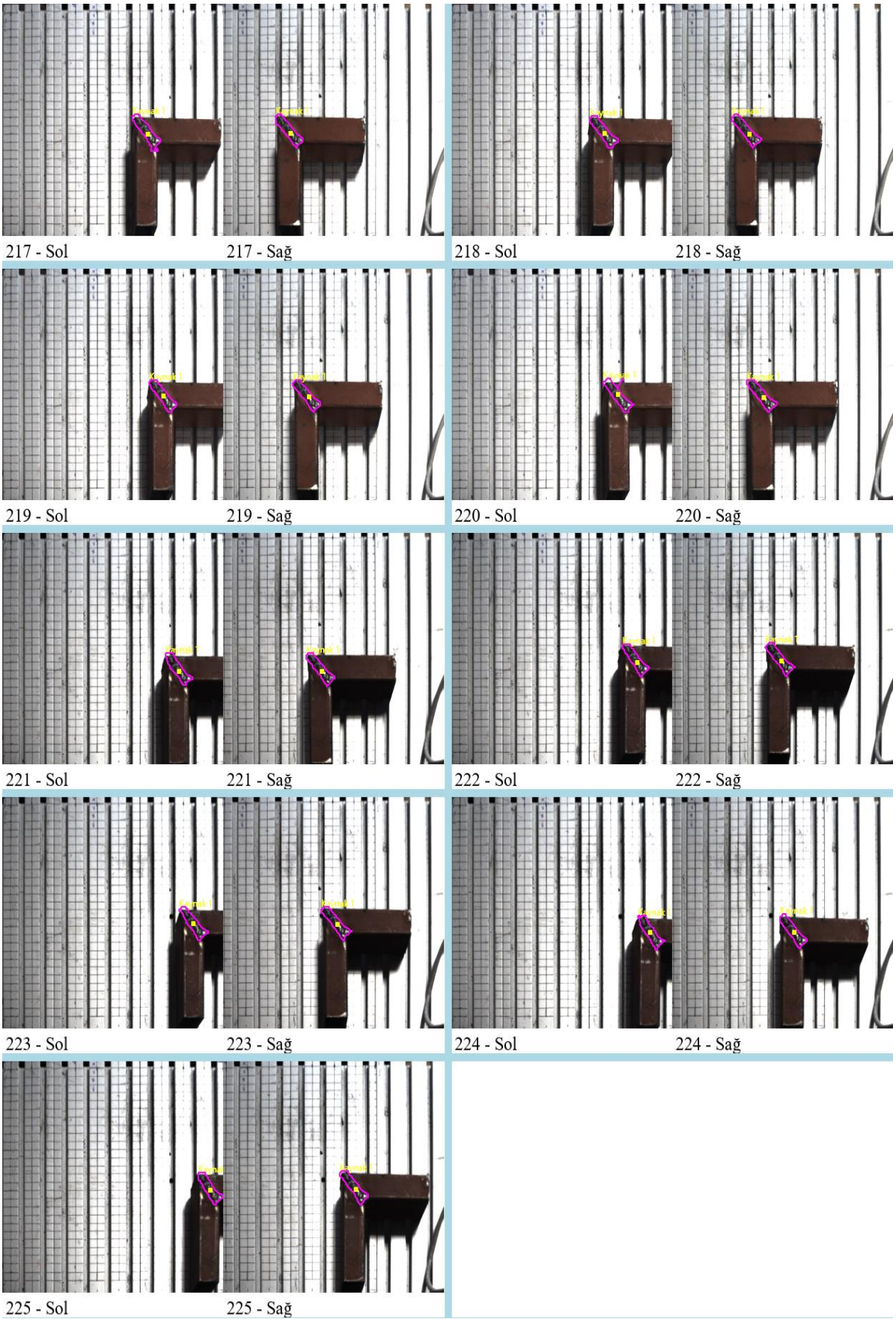
229,877	206,517	27,131	12,869	-0,701	0,579
242,603	211,701	31,413	8,587	-0,312	0,450
253,250	207,505	29,272	10,728	-0,226	0,498
276,338	212,340	42,120	-2,120	0,025	-0,113
280,185	211,629	33,555	6,445	-0,112	0,337
288,584	209,562	31,413	8,587	-0,253	0,424
304,064	214,784	35,696	4,304	-0,222	0,245
300,442	207,795	16,424	23,576	-1,093	1,105
148,074	198,740	33,555	6,445	-1,104	0,219
164,185	197,666	33,555	6,445	-0,956	0,209
175,234	196,730	31,413	8,587	-1,138	0,267
181,920	197,924	29,272	10,728	-1,319	0,351
196,621	198,869	31,413	8,587	-0,876	0,293
205,499	196,978	27,131	12,869	-1,149	0,404
220,996	198,602	35,696	4,304	-0,289	0,145
225,695	197,088	24,989	15,011	-0,907	0,474
247,966	199,681	35,696	4,304	-0,123	0,152
256,504	198,869	31,413	8,587	-0,141	0,293
267,198	198,869	31,413	8,587	-0,010	0,293
275,452	199,098	27,131	12,869	-0,137	0,443
276,387	193,369	14,283	25,717	-0,308	0,675
303,555	198,869	31,413	8,587	-0,436	0,293
311,804	196,859	29,272	10,728	-0,671	0,335
146,788	184,084	29,272	10,728	-1,858	0,139
137,386	153,096	31,413	8,587	-1,602	-0,269
175,687	184,578	35,696	4,304	-0,568	0,059
186,740	183,703	33,555	6,445	-0,748	0,080
197,481	183,703	33,555	6,445	-0,649	0,080
206,407	185,148	29,272	10,728	-0,944	0,156
221,111	184,777	33,555	6,445	-0,432	0,090
228,764	185,148	29,272	10,728	-0,601	0,156
249,014	190,480	42,120	-2,120	0,057	-0,047
254,365	184,968	31,413	8,587	-0,167	0,122
267,090	183,019	29,272	10,728	-0,014	0,123
278,960	184,968	31,413	8,587	-0,134	0,122
287,318	184,084	29,272	10,728	-0,296	0,139
299,028	185,148	29,272	10,728	-0,476	0,156
315,629	187,999	33,555	6,445	-0,439	0,120
149,795	173,790	35,696	4,304	-0,727	-0,007
161,332	174,274	31,413	8,587	-1,308	-0,009
174,608	174,869	35,696	4,304	-0,574	-0,001
189,180	177,621	39,979	0,021	-0,002	0,000
194,696	173,437	29,272	10,728	-1,123	-0,024
210,208	174,869	35,696	4,304	-0,355	-0,001
217,053	173,437	29,272	10,728	-0,781	-0,024
233,999	174,037	33,555	6,445	-0,313	-0,009

243,668	171,308	29,272	10,728	-0,373	-0,057
253,250	172,373	29,272	10,728	-0,226	-0,040
263,681	172,819	24,989	15,011	-0,093	-0,047
278,960	173,205	31,413	8,587	-0,134	-0,022
279,113	172,182	20,707	19,293	-0,306	-0,078
293,753	173,228	20,707	19,293	-0,710	-0,049
311,804	173,437	29,272	10,728	-0,671	-0,024
145,292	163,581	31,413	8,587	-1,505	-0,140
167,414	161,297	39,979	0,021	-0,003	-0,000
171,703	165,444	33,555	6,445	-0,887	-0,088
187,553	159,765	35,696	4,304	-0,495	-0,094
199,420	161,923	35,696	4,304	-0,422	-0,080
199,639	159,634	20,707	19,293	-1,884	-0,424
219,077	161,442	31,413	8,587	-0,600	-0,166
237,063	162,385	39,979	0,021	-0,001	-0,000
243,672	162,511	31,413	8,587	-0,298	-0,153
254,365	164,650	31,413	8,587	-0,167	-0,127
274,063	165,650	39,979	0,021	-0,000	-0,000
286,034	158,032	39,979	0,021	-0,001	-0,001
288,382	161,727	29,272	10,728	-0,312	-0,203
308,379	167,317	35,696	4,304	-0,248	-0,047
311,804	160,662	29,272	10,728	-0,671	-0,220
152,831	151,133	42,120	-2,120	0,349	0,072
166,696	147,529	37,837	2,163	-0,313	-0,085
175,687	151,135	35,696	4,304	-0,568	-0,147
188,632	148,977	35,696	4,304	-0,488	-0,160
197,890	146,822	29,272	10,728	-1,074	-0,432
205,499	147,163	27,131	12,869	-1,149	-0,512
222,185	147,185	33,555	6,445	-0,422	-0,256
231,851	150,407	33,555	6,445	-0,333	-0,226
245,814	149,333	33,555	6,445	-0,204	-0,236
254,407	148,259	33,555	6,445	-0,125	-0,246
267,198	149,679	31,413	8,587	-0,010	-0,311
280,030	147,540	31,413	8,587	-0,148	-0,337
284,784	147,495	24,989	15,011	-0,360	-0,590
299,028	147,887	29,272	10,728	-0,476	-0,416
307,998	150,661	24,989	15,011	-0,858	-0,522
145,723	139,370	29,272	10,728	-1,874	-0,546
161,693	137,240	29,272	10,728	-1,629	-0,579
176,765	137,111	35,696	4,304	-0,561	-0,233
187,553	137,111	35,696	4,304	-0,495	-0,233
197,481	137,518	33,555	6,445	-0,649	-0,345
207,471	140,434	29,272	10,728	-0,928	-0,530
219,077	136,847	31,413	8,587	-0,600	-0,468
227,805	135,888	24,989	15,011	-0,862	-0,839
238,348	137,313	22,848	17,152	-0,727	-0,923

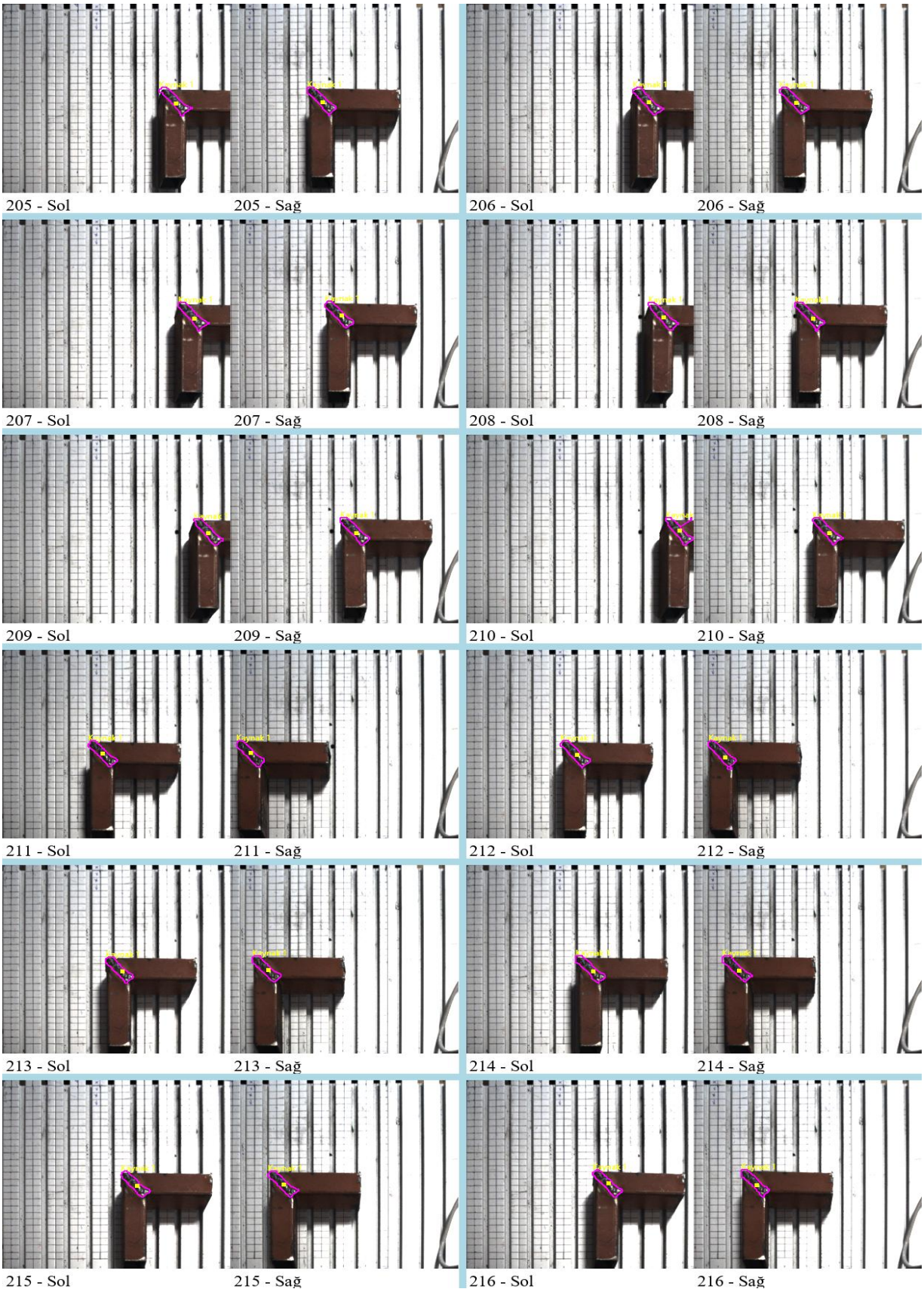
246,752	139,414	22,848	17,152	-0,521	-0,872
266,128	125,084	31,413	8,587	-0,023	-0,612
277,572	135,505	27,131	12,869	-0,176	-0,726
284,991	139,744	27,131	12,869	-0,312	-0,648
296,650	128,085	27,131	12,869	-0,527	-0,863
307,677	135,213	22,848	17,152	-0,972	-0,975

Çizelge 4.1'de sunulan nicel sonuçların elde edildiği temel veri kaynağını görsel olarak belgelemek amacıyla, 225 test noktasının tamamı için deney sırasında yakalanan görüntü çiftleri aşağıda sunulmuştur. Şekil 4.4'ten Şekil 4.22'ye kadar olan şekillerde, her bir test noktası için kameranın ilk konumundan ("Sol" olarak etiketlenmiş) ve 100 mm ötelenmiş ikinci konumundan ("Sağ" olarak etiketlenmiş) alınan görüntüler yan yana gösterilmektedir. Her bir görüntü çifti arasındaki yatay piksel kayması (paralaks), Bölüm 3'te açıklanan derinlik hesaplama metodolojisi için temel girdi olarak kullanılmıştır. Bu ham görüntülerin sunulması, yapılan ölçümlerin görsel bir kaydını oluşturarak çalışmanın şeffaflığına ve sonuçların doğrulanabilirliğine katkıda bulunmaktadır.

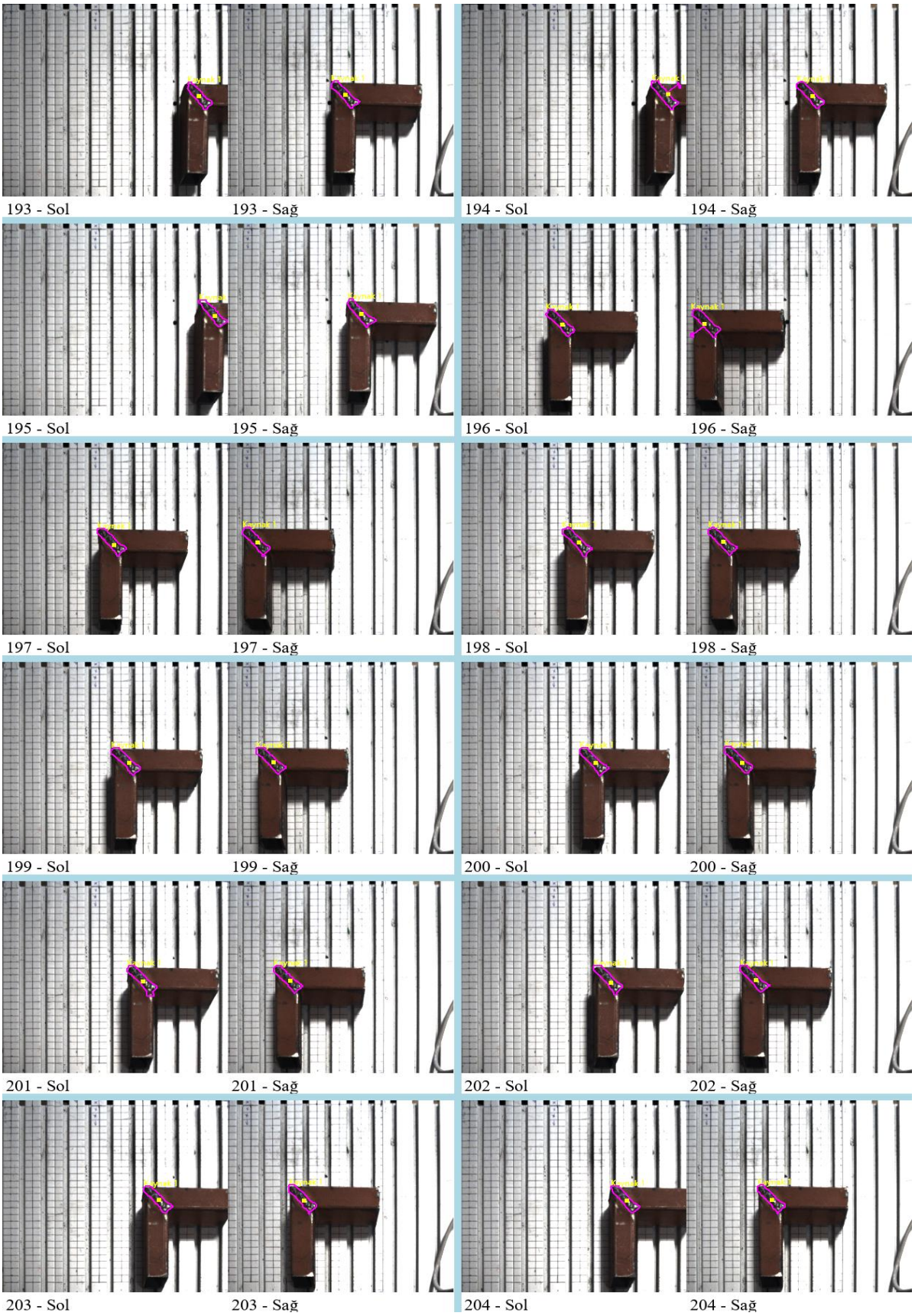
Sunulan görüntü çiftleri incelendiğinde, her bir test noktası için "Sol" ve "Sağ" olarak etiketlenmiş görüntüler arasında, beklenen yatay kayma (paralaks) etkisinin net bir şekilde gözlemlendiği görülmektedir. Bu durum, tezin metodoloji bölümünde açıklanan hareketli paralaks yönteminin temelini oluşturan Δx değerinin, 225 noktanın tamamı için başarılı bir şekilde elde edildiğini görsel olarak teyit etmektedir. Ayrıca, farklı konumlardaki görüntü çiftleri arasında yapılan nitel bir karşılaştırma, paralaks kaymasının büyüklüğünün çalışma alanı genelinde sabit olmadığını, nesnenin kameranın görüş alanındaki konumuna göre küçük de olsa farklılıklar gösterdiğini ortaya koymaktadır. Bu gözlem, Z eksenindeki sistematik hatanın temel nedenlerinden biri olan, piksel kayması ile gerçek dünya derinliği arasındaki ilişkinin tam olarak doğrusal olmaması bulgusunu görsel düzeyde destekleyen önemli bir kanıttır.



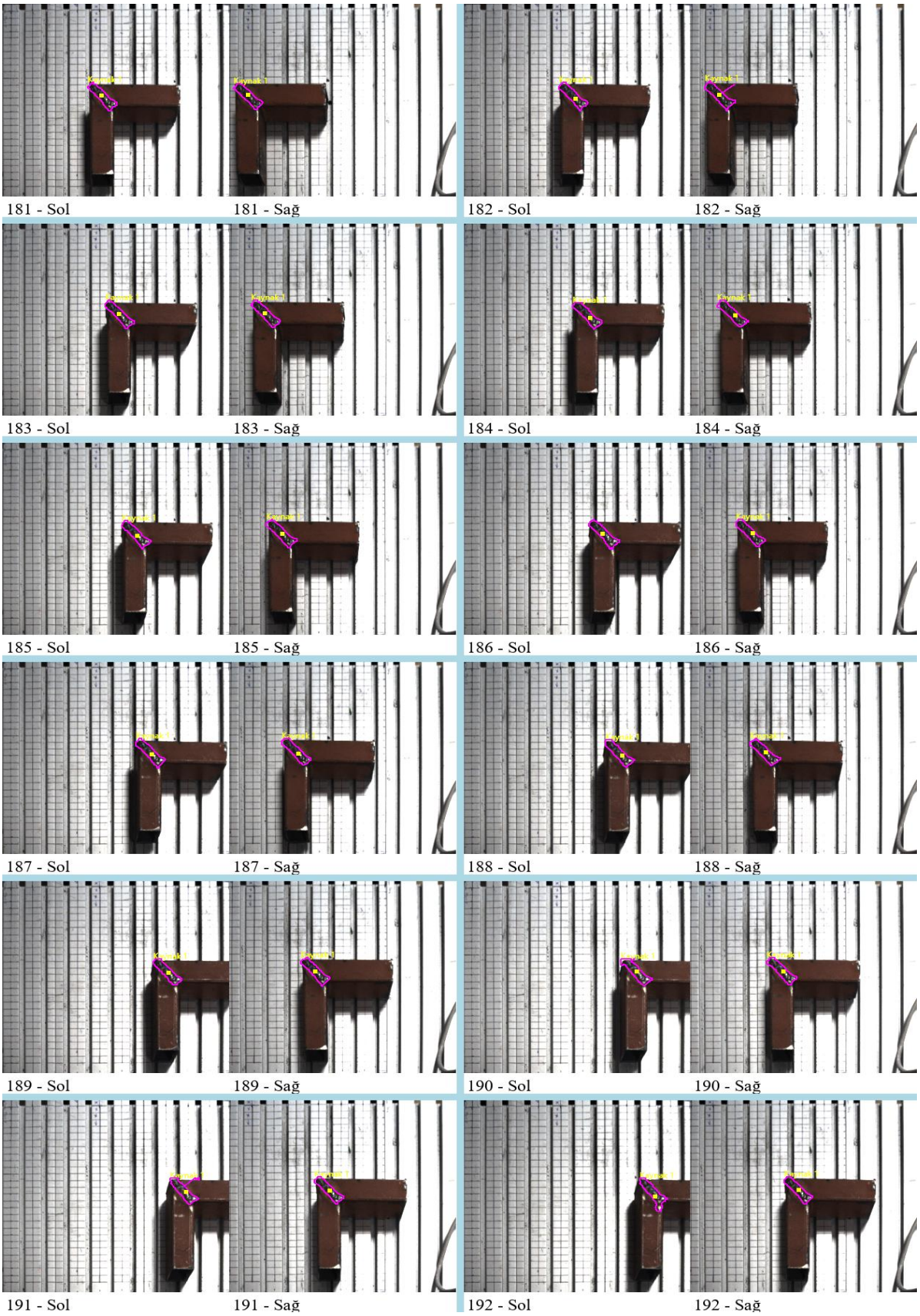
Şekil 4.4: Deney setindeki 217-225 arası test noktaları için paralaks görüntü çiftleri.



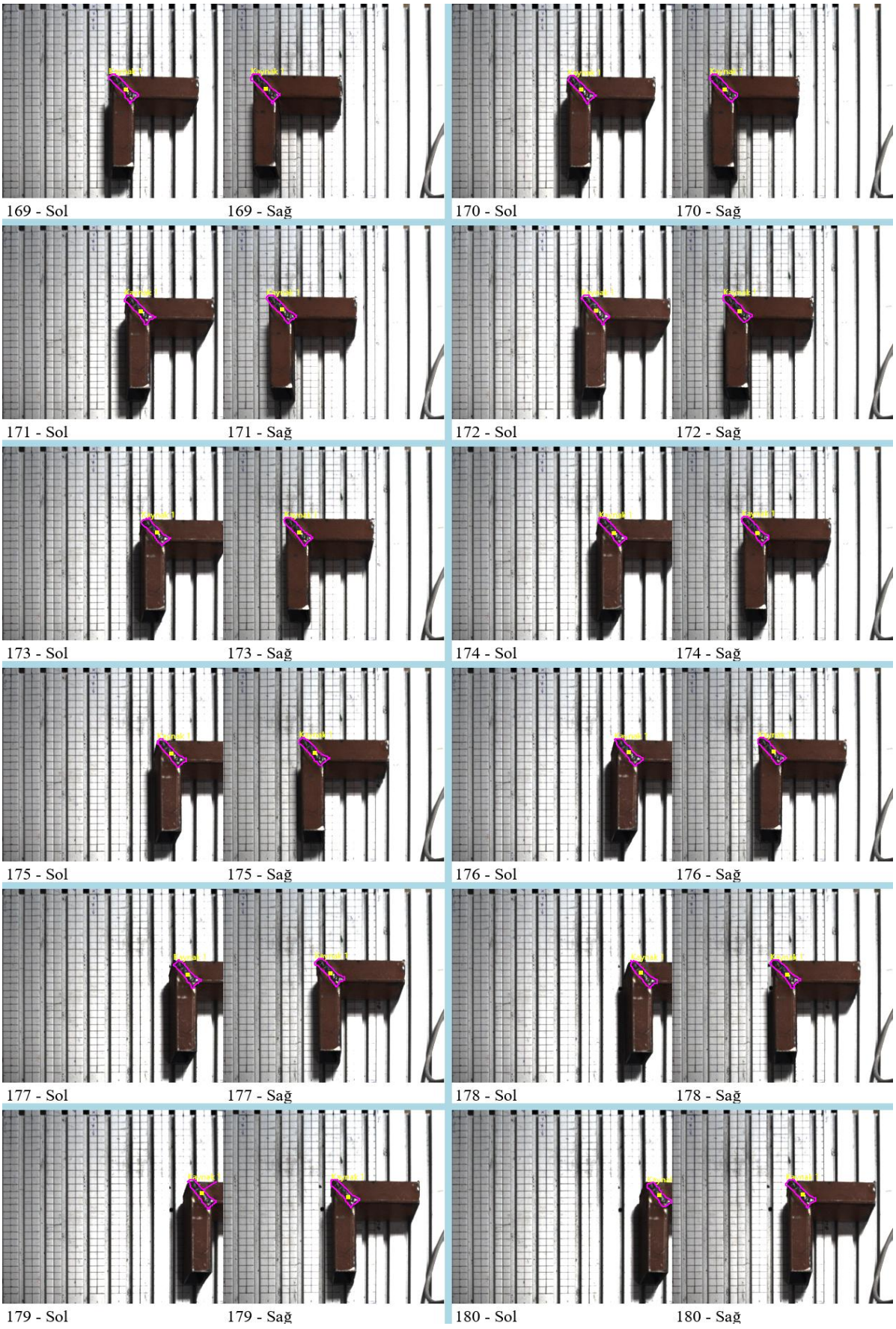
Şekil 4.5: Deney setindeki 205-216 arası test noktaları için paralaks görüntü çiftleri.



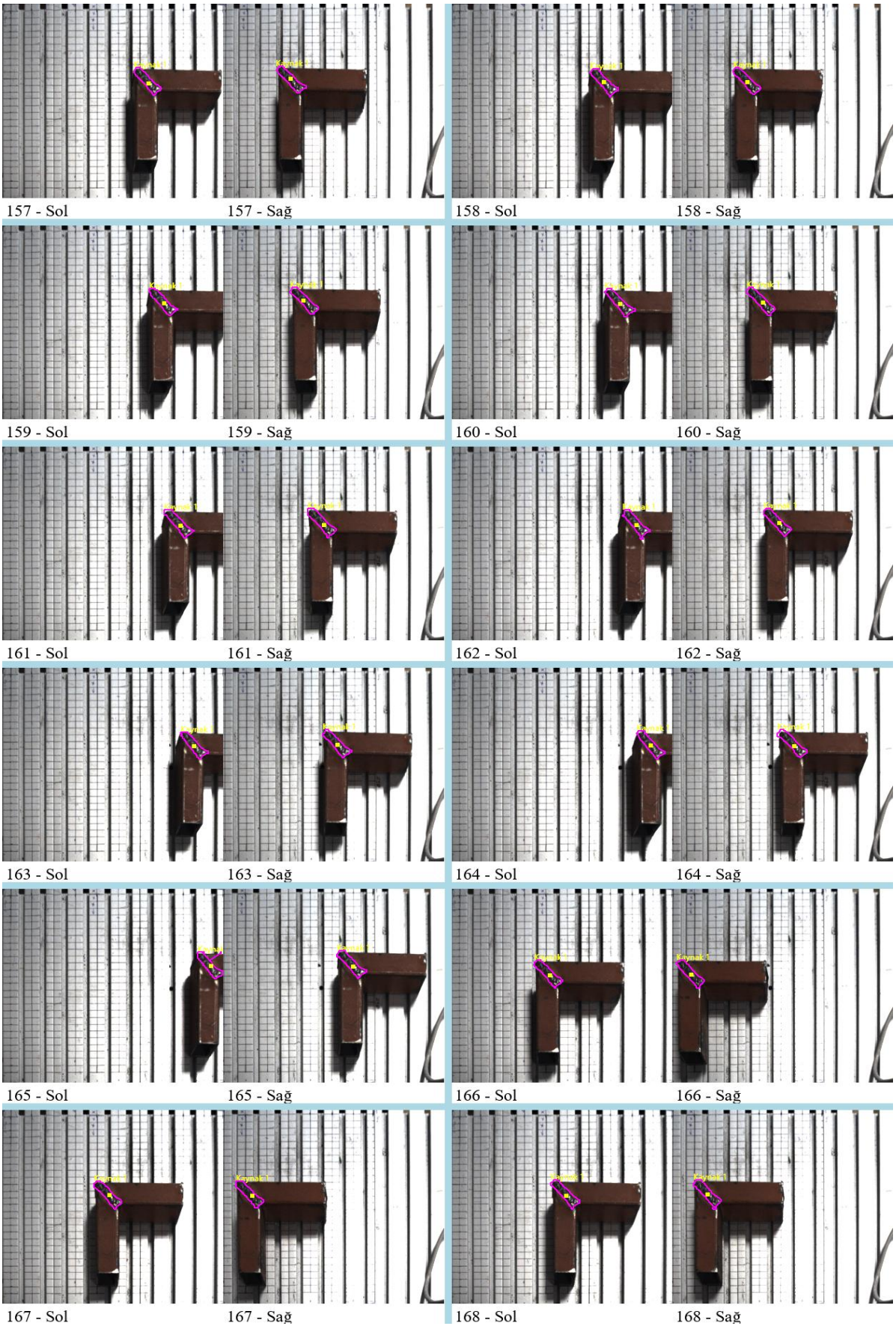
Şekil 4.6: Deney setindeki 193-204 arası test noktaları için paralaks görüntü çiftleri.



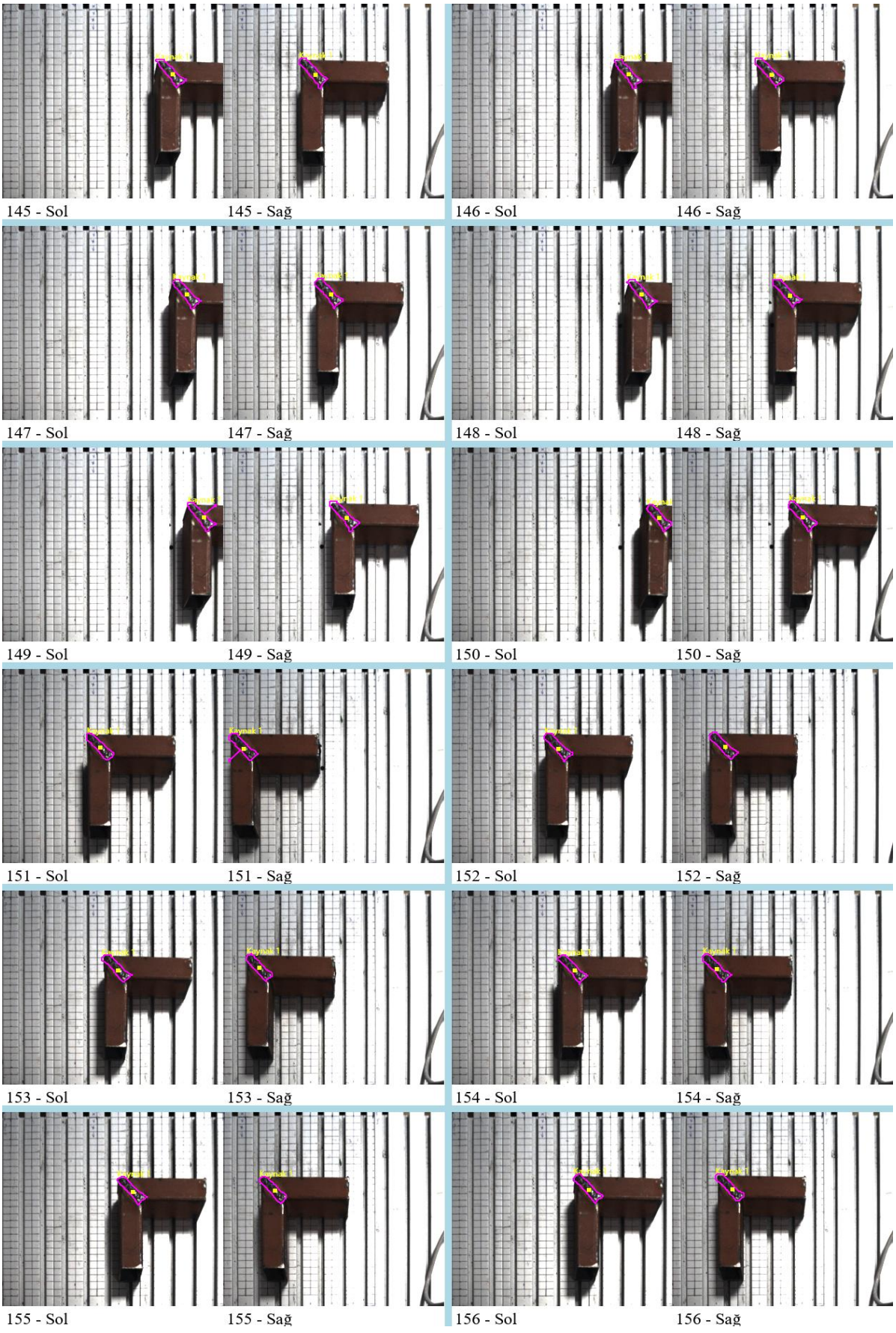
Şekil 4.7: Deney setindeki 181-192 arası test noktaları için paralaks görüntü çiftleri.



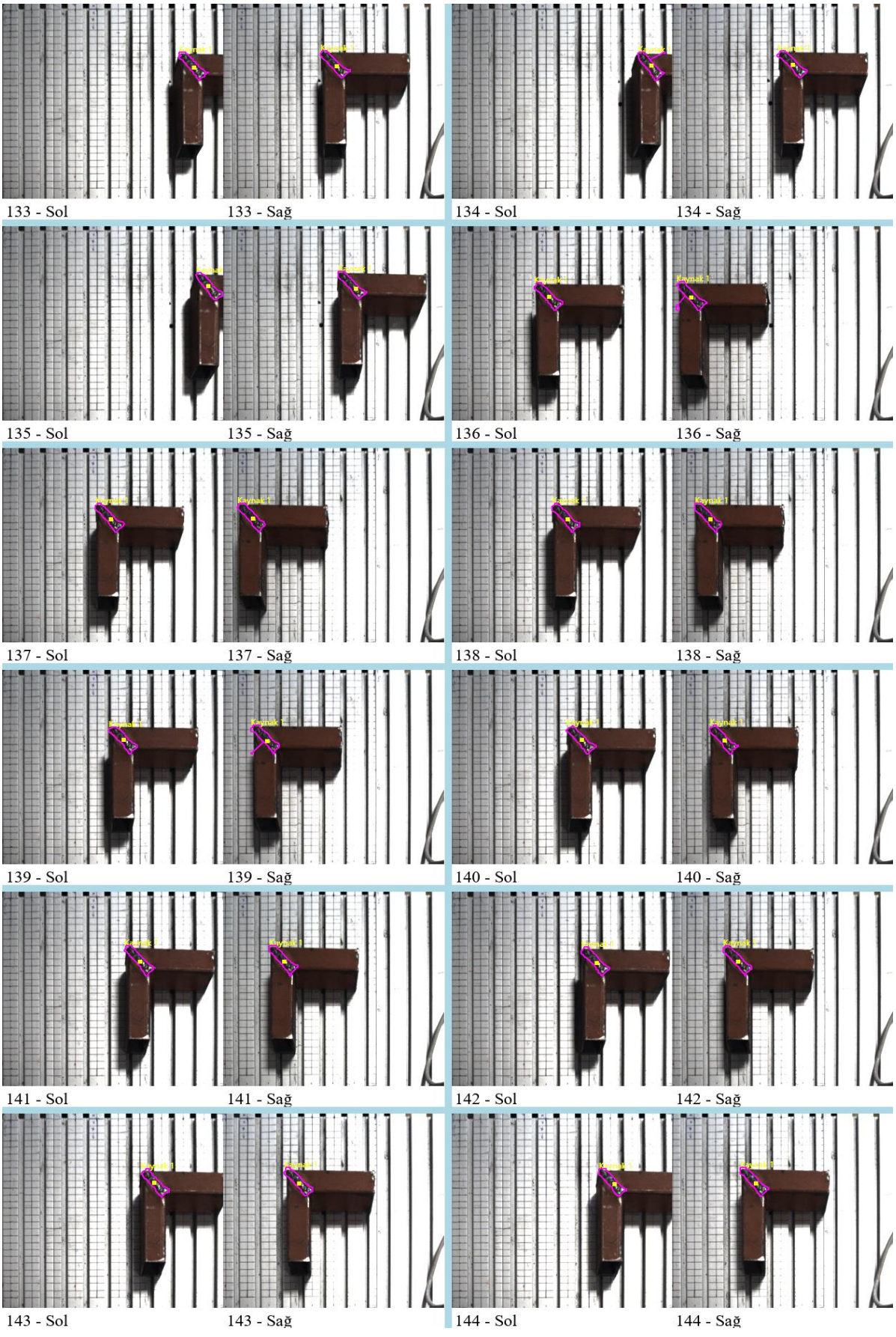
Şekil 4.8: Deney setindeki 169-180 arası test noktaları için paralaks görüntü çiftleri.



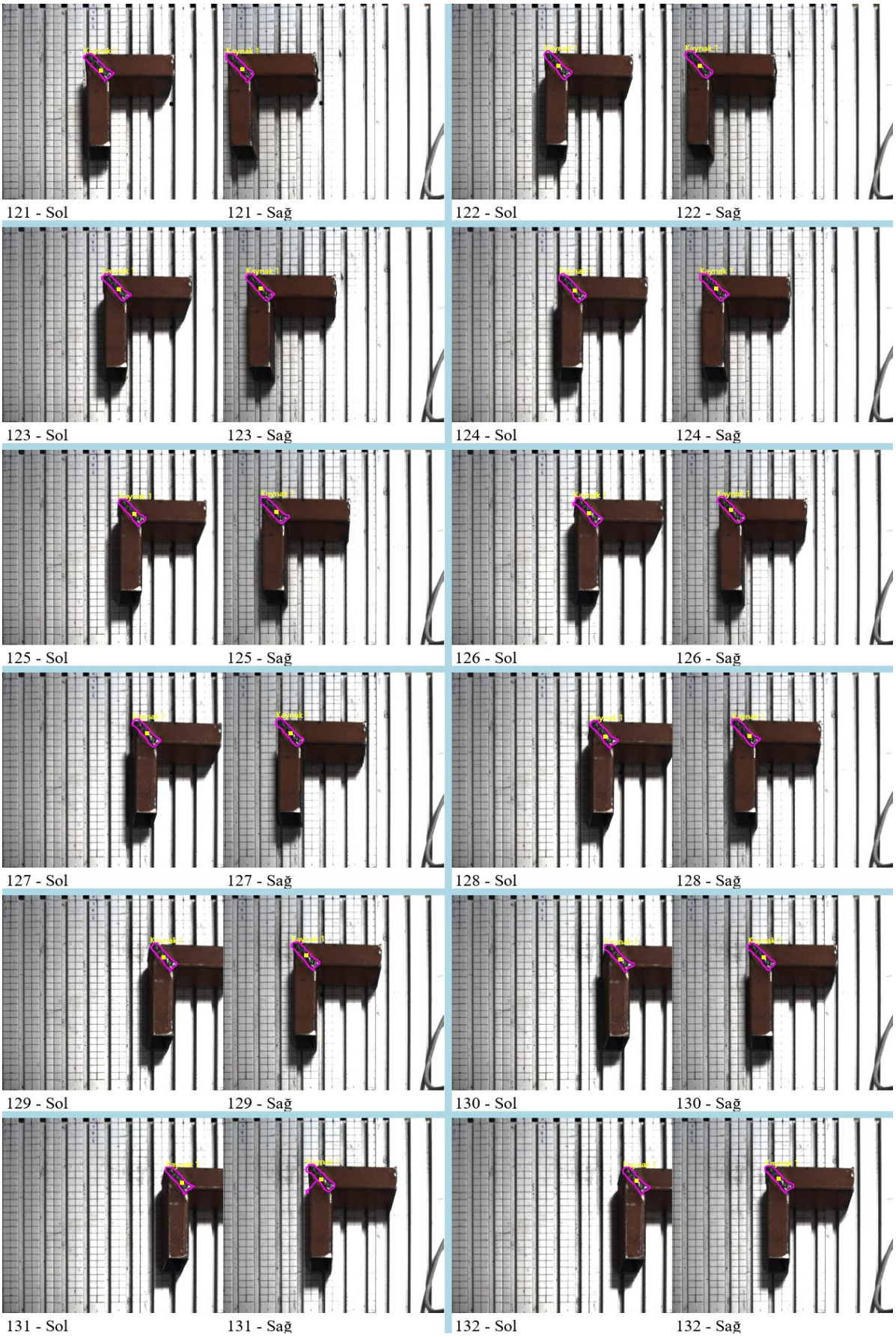
Şekil 4.9: Deney setindeki 157-168 arası test noktaları için paralaks görüntü çiftleri.



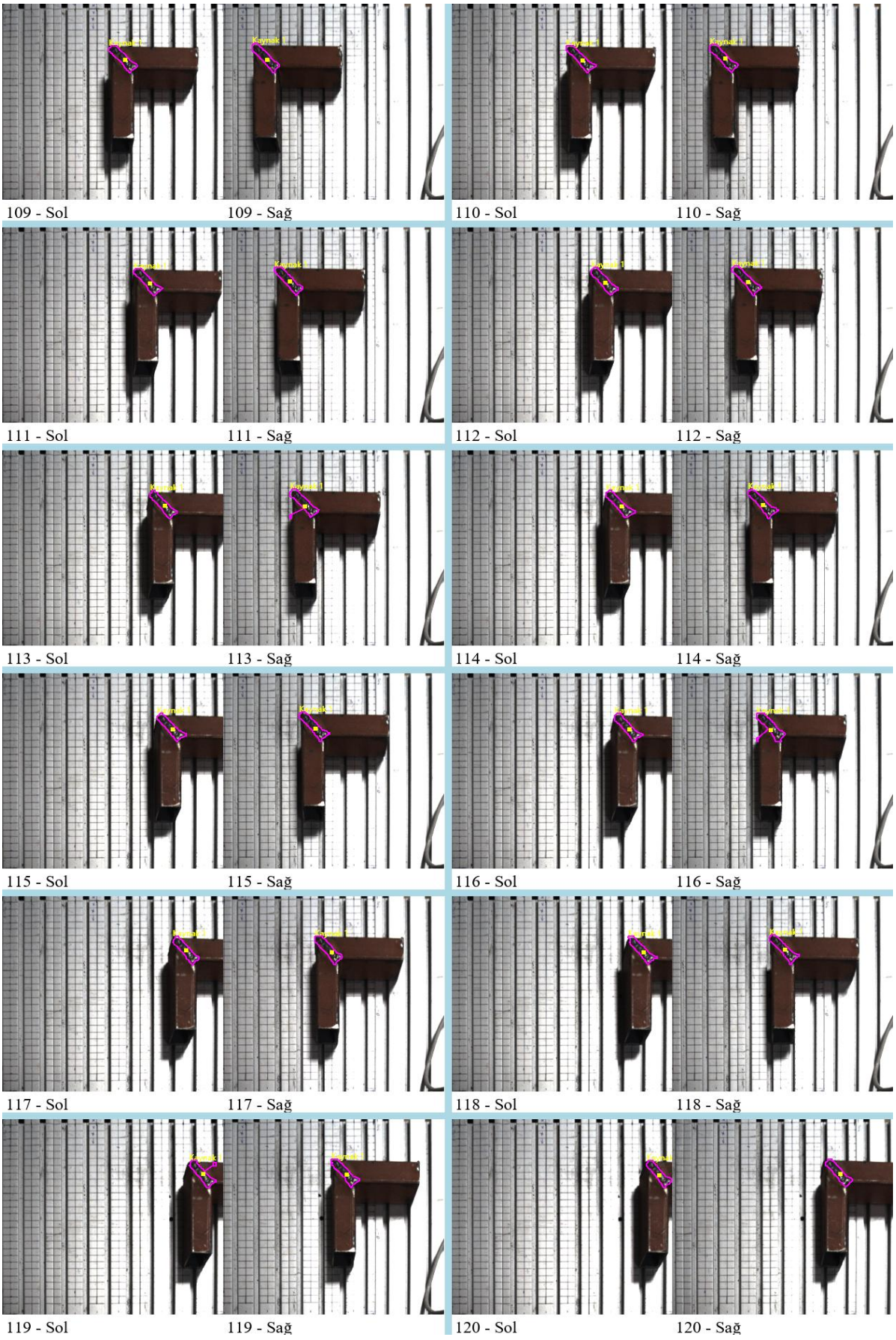
Şekil 4.10: Deney setindeki 145-156 arası test noktaları için paralaks görüntü çiftleri.



Şekil 4.11: Deney setindeki 133-144 arası test noktaları için paralaks görüntü çiftleri.



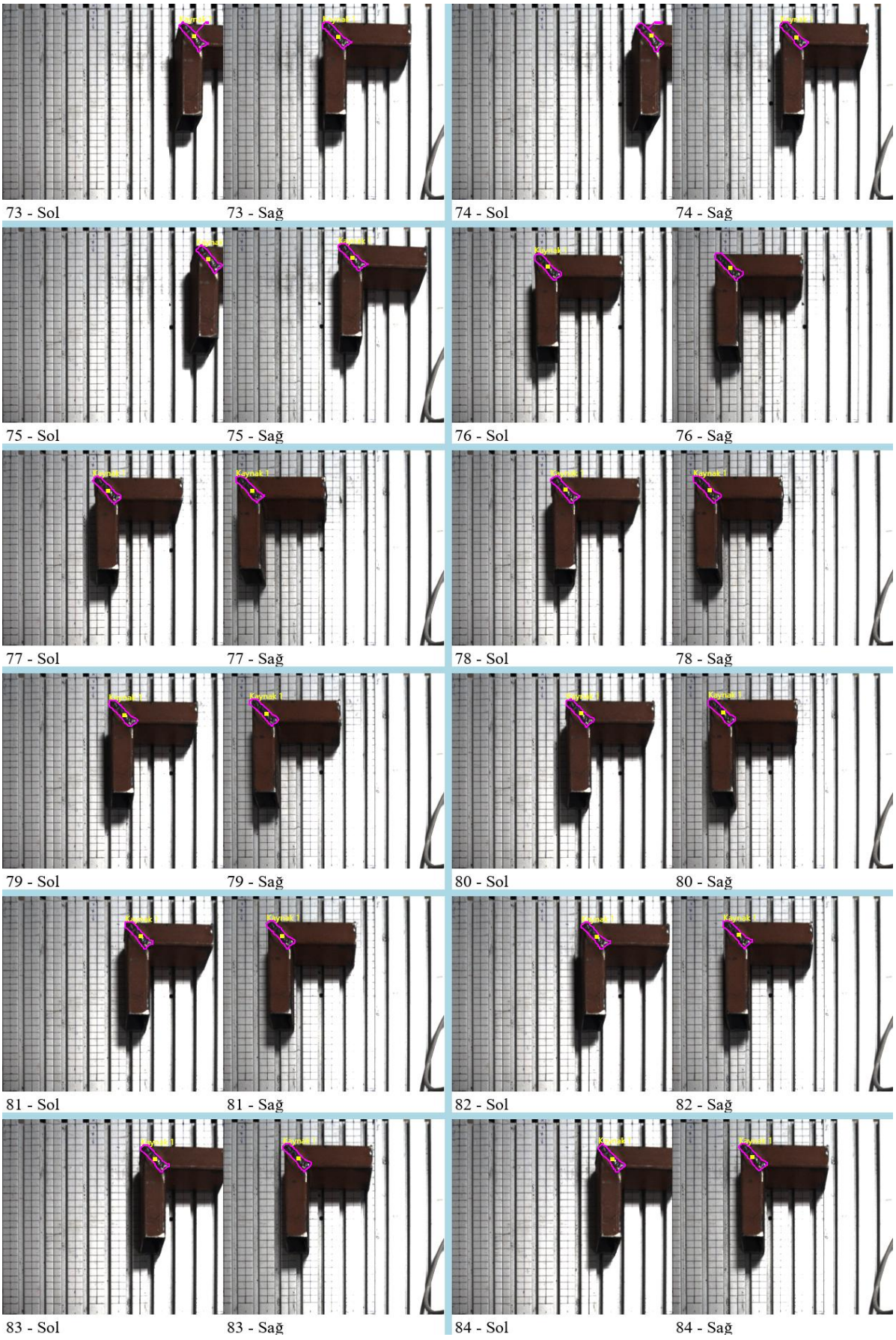
Şekil 4.12: Deney setindeki 121-132 arası test noktaları için paralaks görüntü çiftleri.



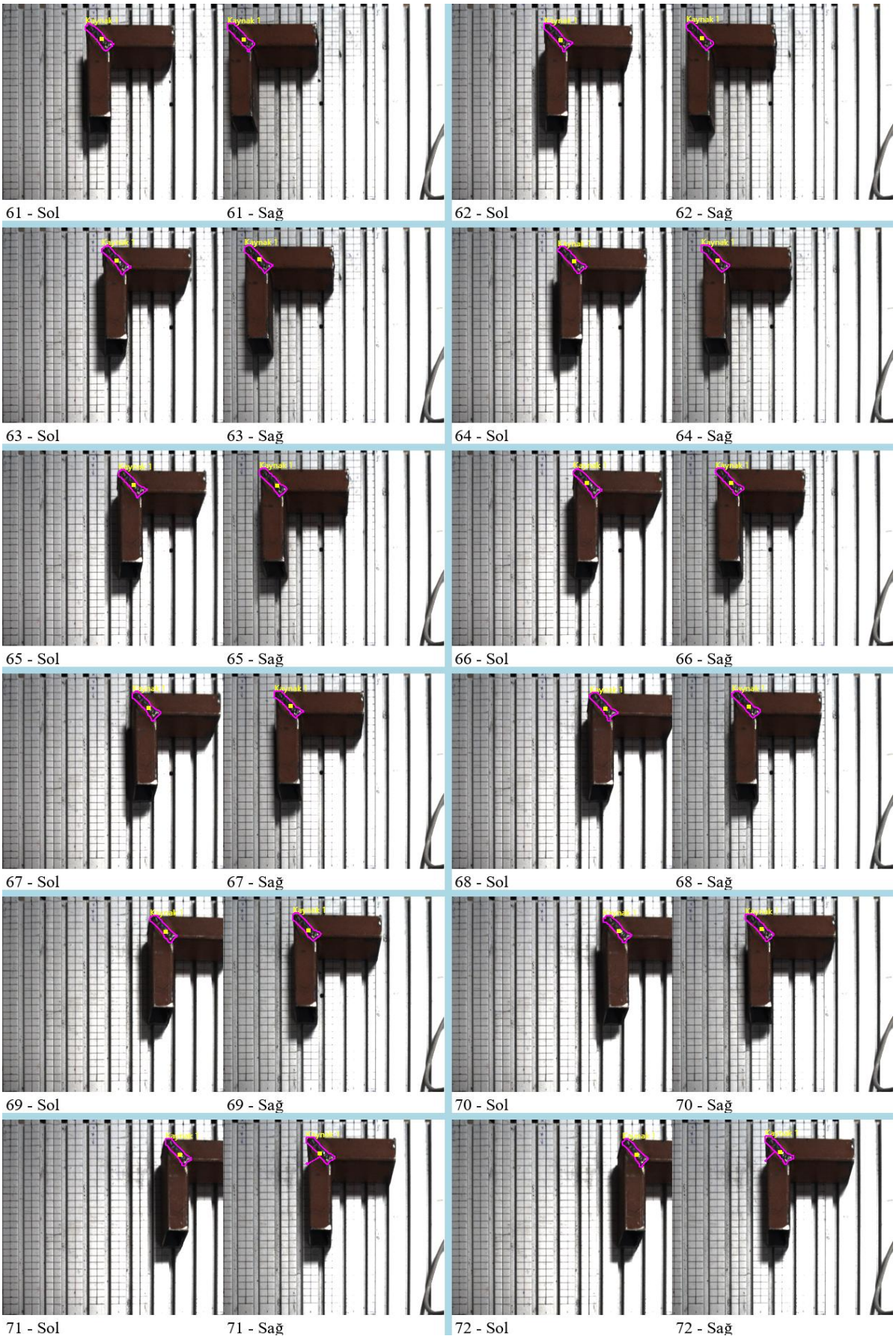
Şekil 4.13: Deney setindeki 109-120 arası test noktaları için paralaks görüntü çiftleri.

Şekil 4.14: Deney setindeki 97-108 arası test noktaları için paralaks görüntü çiftleri.

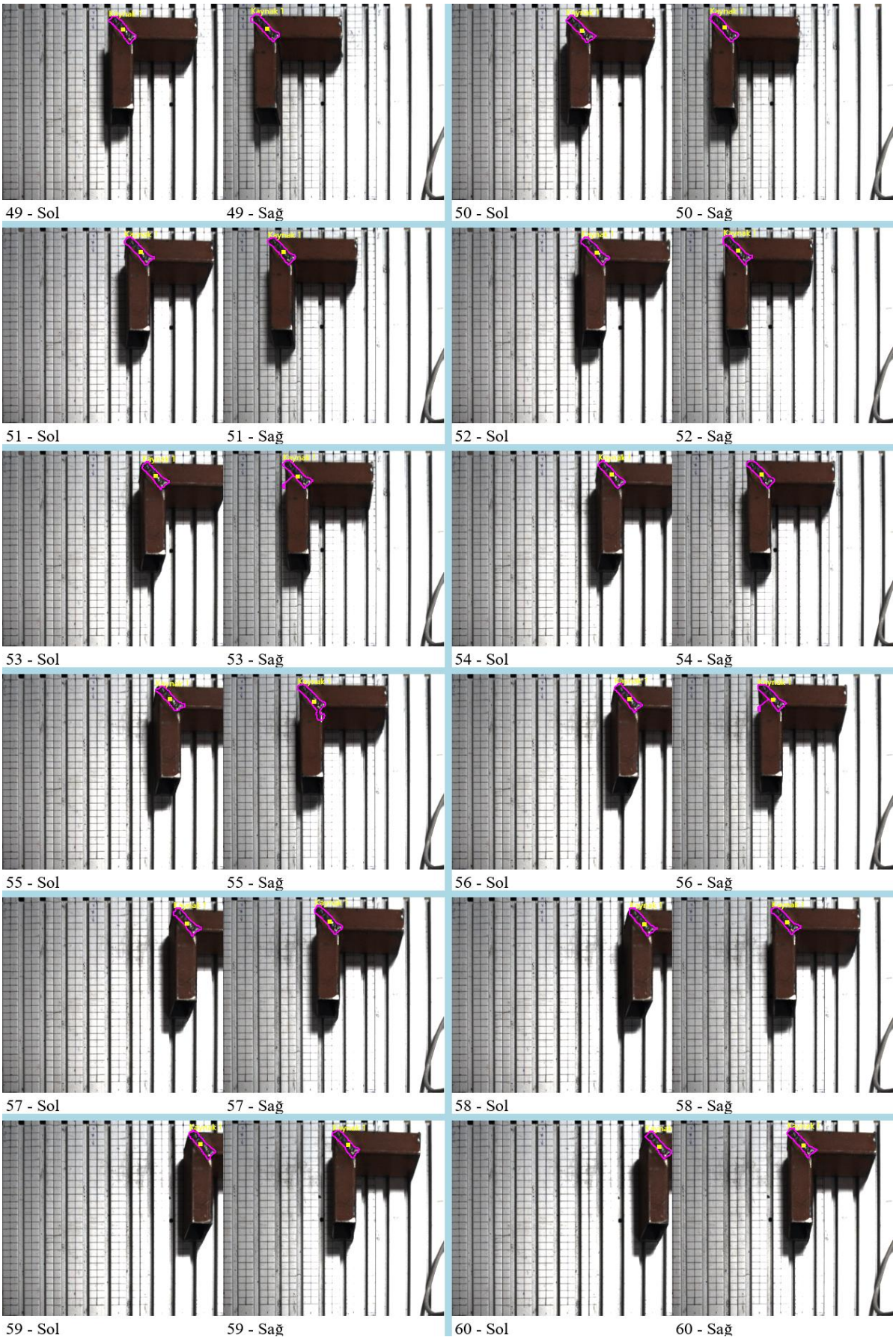
Şekil 4.15: Deney setindeki 85-96 arası test noktaları için paralaks görüntü çiftleri.



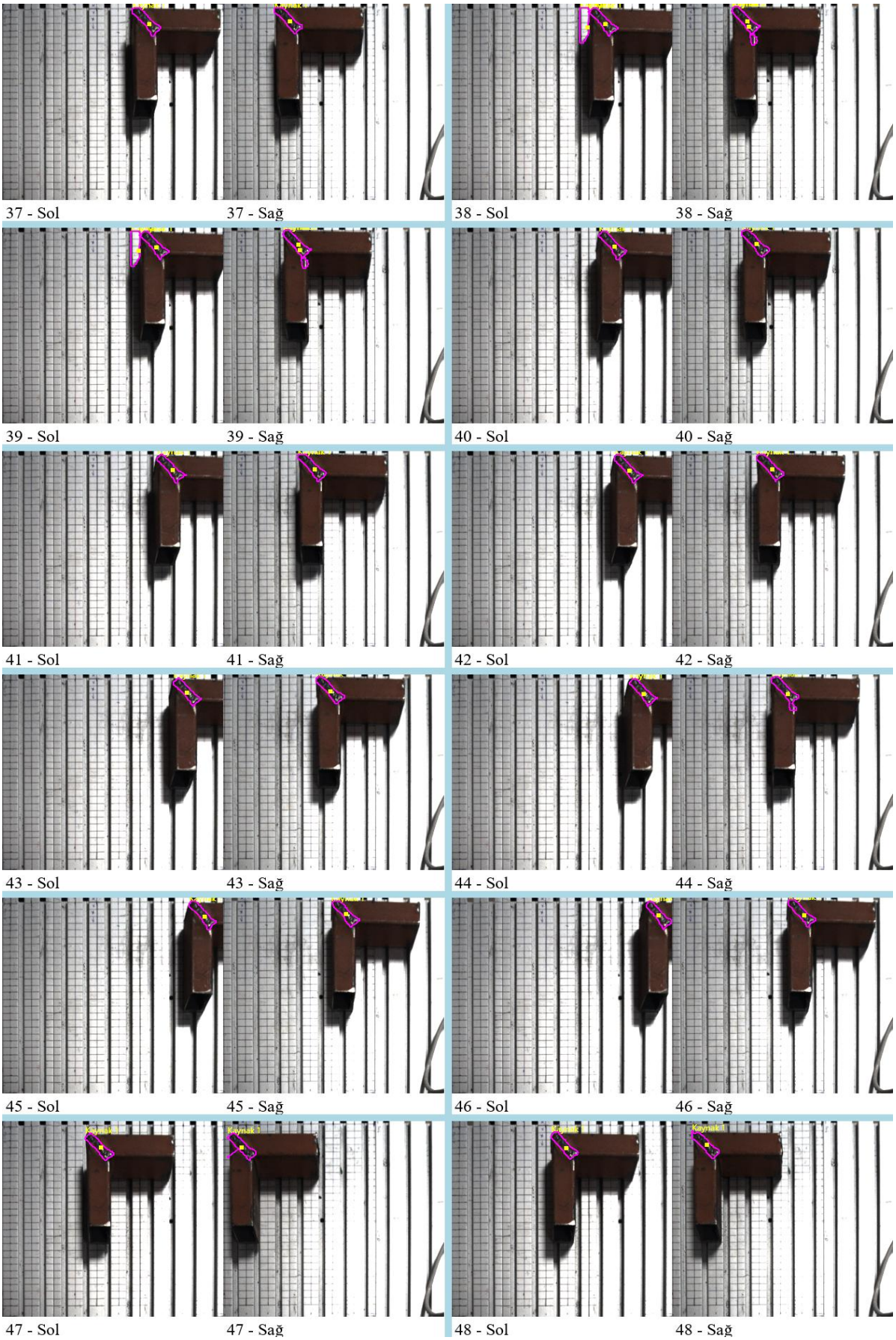
Şekil 4.16: Deney setindeki 73-84 arası test noktaları için paralaks görüntü çiftleri.



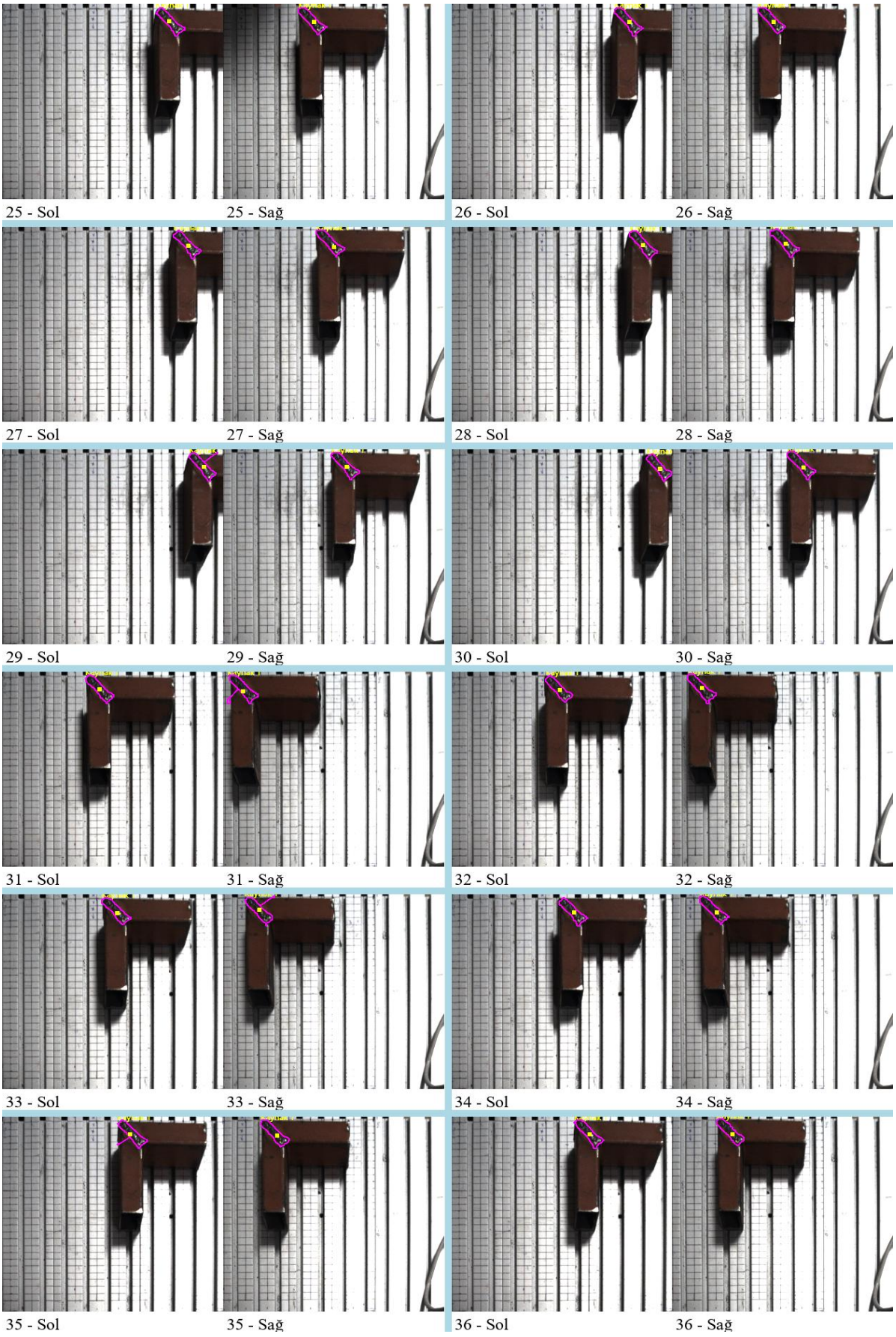
Şekil 4.17: Deney setindeki 61-72 arası test noktaları için paralaks görüntü çiftleri.



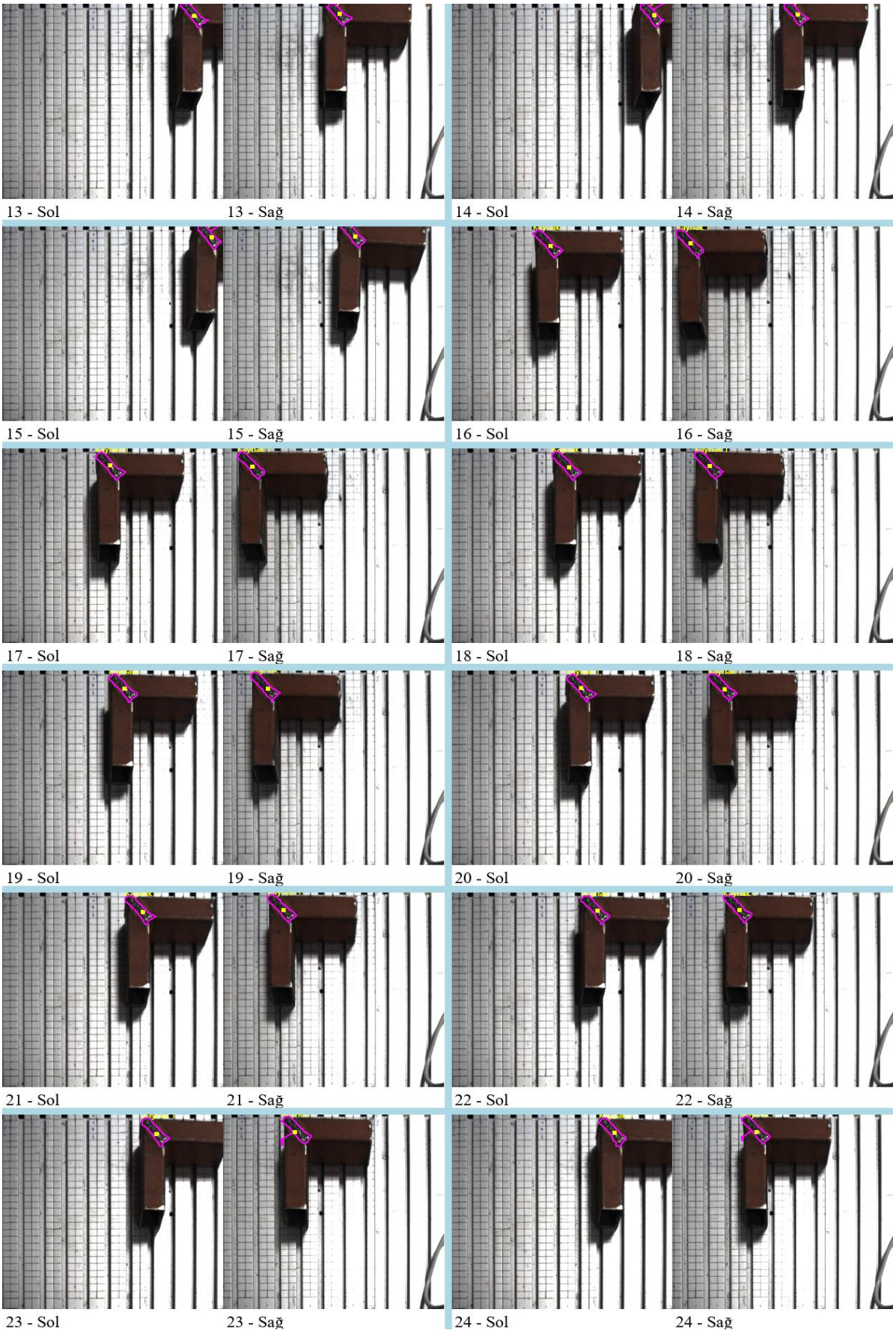
Şekil 4.18: Deney setindeki 49-60 arası test noktaları için paralaks görüntü çiftleri.



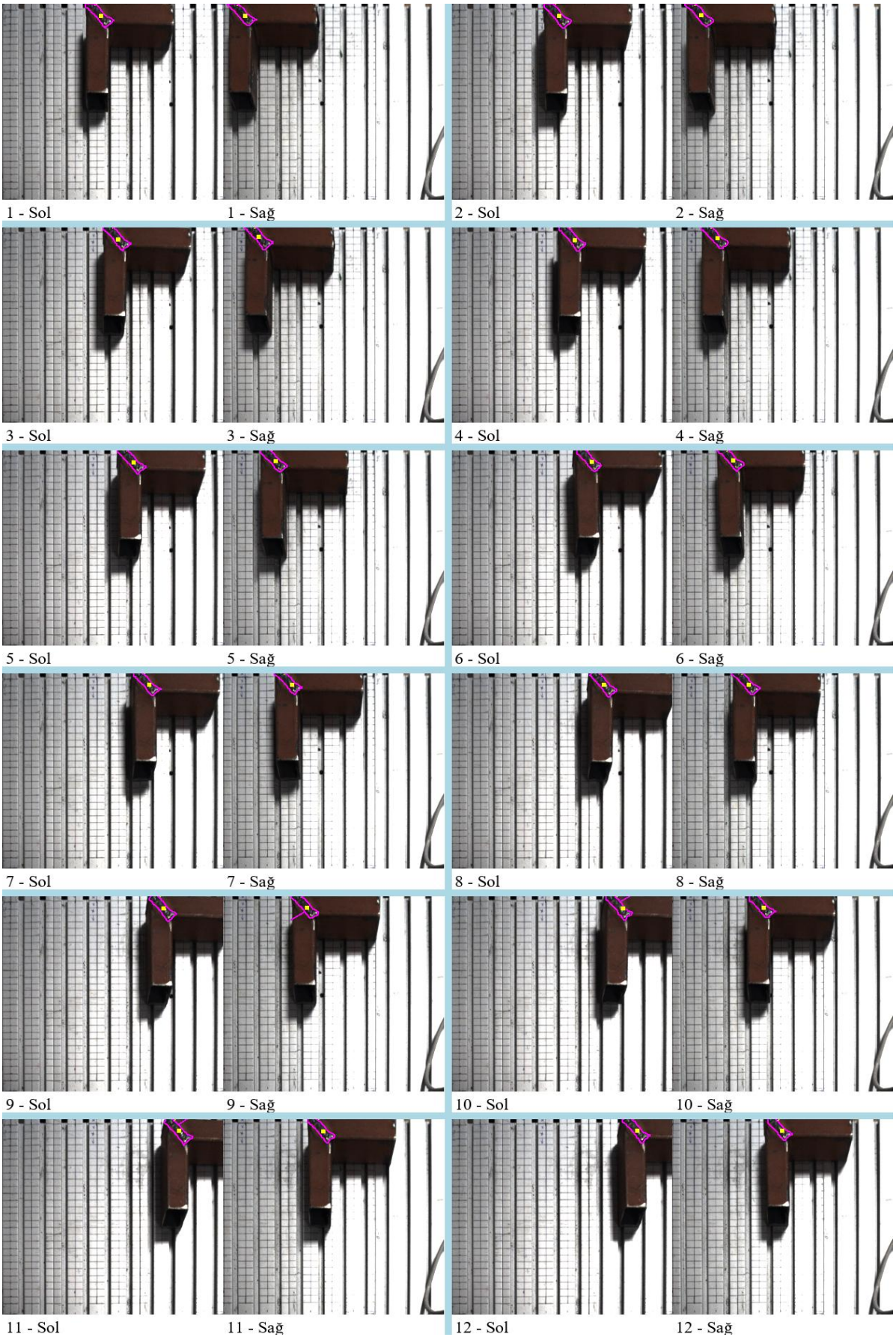
Şekil 4.19: Deney setindeki 37-48 arası test noktaları için paralaks görüntü çiftleri.



Şekil 4.20: Deney setindeki 25-36 arası test noktaları için paralaks görüntü çiftleri.



Şekil 4.21: Deney setindeki 13-24 arası test noktaları için paralaks görüntü çiftleri.



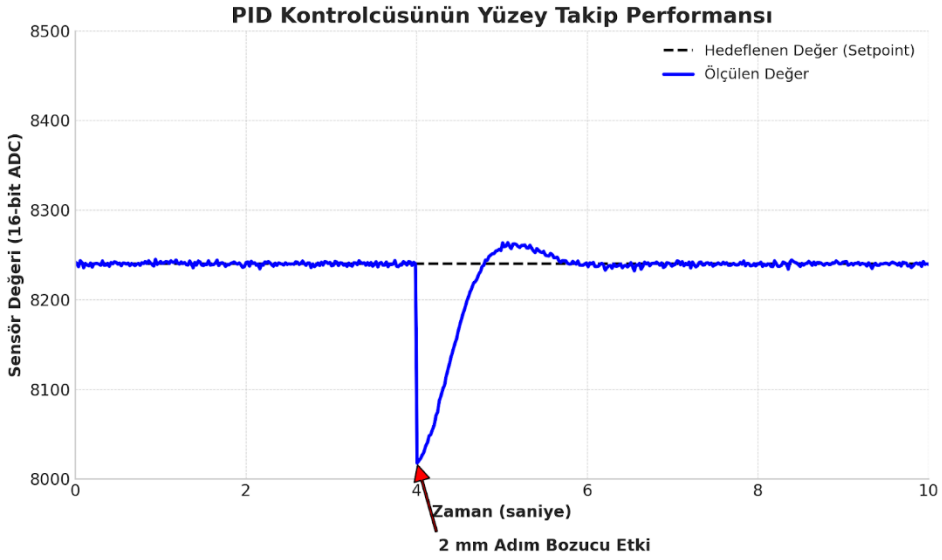
Şekil 4.22 Deney setindeki 1-12 arası test noktaları için paralaks görüntü çiftleri.

4.2. Yüzey Takibi ve İşleme Performansı

Bu testler, sistemin PID kontrolcüsünün, engebeli bir yüzey üzerinde takım basıncını sabit tutma ve yüzey topolojisini takip etme yeteneğini ölçmeyi amaçlamaktadır.

Çalışma tablasına, üzerinde 2 mm yüksekliğinde ani bir basamak bulunan eğimli bir test parçası yerleştirilmiştir. Takım ucu yüzeye temas ettirildikten sonra PID kontrol döngüsü aktif edilmiş ve X ekseninde sabit bir hızla hareket ettirilerek takımın bu profili takip etmesi sağlanmıştır. Bu sırada, Hall Effect sensöründen gelen ham veri ve Z eksenine uygulanan düzeltme komutları anlık olarak kaydedilmiştir.

Şekil 4.4'te, test sırasındaki sensör verilerinin zaman içindeki değişimi gösterilmektedir. Grafikte, hedeflenen sabit sensör değeri ile yüzeydeki geometri değişikliğine rağmen gerçek zamanlı olarak ölçülen sensör değeri karşılaştırılmaktadır.



Şekil 4.4 Hall Effect sensörünün yüzey takibinin nesne engeli sonucundaki reaksiyonu

Şekil 4.4'te sunulan grafik, geliştirilen P-kontrol sisteminin performansını ve metodolojinin başarısını birden fazla açıdan kanıtlamaktadır. Grafikte en dikkat çekici nokta, sistemin harici bir bozucu etkiye karşı gösterdiği tepkidir. Takım, test parçasındaki 2 mm'lik ani basamağa ulaştığında, bu durum kontrol döngüsü için bir adım bozucu etkisi yaratır ve sensör değerinde anlık bir düşüşe neden olur. Sistemin bu ani hataya saniyenin onda birinden daha kısa bir sürede tepki vermesi, Z eksenini hızla yukarı yönlü hareket ettirerek sensör değerini tekrar hedef setpoint seviyesine getirmesi, kontrolcünün yanıt verme hızının ne kadar yüksek olduğunu göstermektedir. Bu hızlı reaksiyon kabiliyeti, gerçek bir kaynak dikişinin

üzerindeki ani pürüzler veya geometri değişiklikleri karşısında takımın yüzeyle temasını kaybetmemesi veya yüzeye zarar vermemesi için hayati öneme sahiptir.

Grafikteki tepki incelendiğinde, kontrolcünün düzeltme sonrası hedef değer etrafında kalıcı bir salınıma girmediği ve minimum aşım ile hızla kararlı hale geldiği görülmektedir. Bu durum, Bölüm 3.6.6'da bahsedilen deneysel ayarlama süreciyle belirlenen Kp (oransal kazanç) değerinin optimum bir dengede olduğunu göstermektedir. Kazanç, sistemi kararsızlığa sürükleyecek kadar agresif olmamakla birlikte, bozucu etkileri bertaraf edecek kadar da hızlıdır.

Test boyunca sensör değerinin hedeflenen değer etrafındaki RMS (Karekök Ortalama Kare) sapmasının 0,008 mm gibi son derece düşük bir değerde hesaplanmış olmasıdır. Bu metrik, sadece bir istatistiksel doğruluk ölçütü değildir. Hall Effect sensörünün çıktısı, yayın sıkışma miktarıyla doğrudan ilişkili olduğundan, bu düşük RMS hatası, yayın sıkışmasının ve dolayısıyla takımın iş parçasına uyguladığı temas kuvvetinin tüm süreç boyunca neredeyse sabit kaldığını göstermektedir. Sabit temas kuvveti ise, homojen malzeme kaldırma ve yüksek kalitede bir yüzey elde etmenin temel gerekliliğidir. Bu bulgu, tezin özetinde belirtilen 0,01 mm RMS hata payı hedefinin başarıyla karşılandığını ve geliştirilen PID kontrol sisteminin, standart bir CNC platformuna daha gelişmiş, kuvvete duyarlı robotların yüzey takip kabiliyetini kazandırdığını kanıtlamaktadır.

4.3. Yapay Zekâ Modelinin Tespit Başarısı

Bu bölümde, kaynak noktalarını tespit etmek amacıyla, Bölüm 3.5'te açıklanan metodolojiyle geliştirilen ve eğitilen özel YOLOv8 segmentasyon modelinin nicel ve nitel performans sonuçları sunulmakta ve tartışılmaktadır.

Bu çalışma için, öncelikle 49 adet özgün kaynak dikişi görüntüsünden oluşan bir temel veri kümesi oluşturulmuştur. Modelin farklı koşullar altında dahi kararlı çalışabilmesi ve genelleme yeteneğinin artırılması amacıyla, Bölüm 3.5.2'de açıklanan arayüz aracılığıyla, her biri farklı bir amaca hizmet eden altı temel veri artırma tekniğinden oluşan bir işlem hattı tanımlanmıştır:

- **Geometrik Artırmalar:**
    - *Döndürme (Rotation):* İş parçasının veya kameranın farklı açılarda olabileceği durumları simüle etmek için.
    - *Yatay Çevirme (Horizontal Flip):* Kaynak dikişlerinin simetrik veya ters yönlü görünümlerine karşı modeli dayanıklı hale getirmek için.

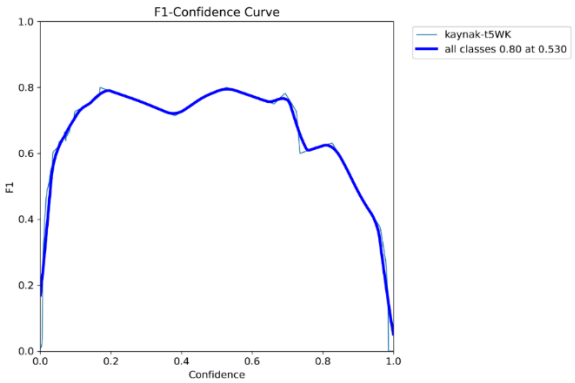
    - *Ölçek (Scale):* Kameranın iş parçasına olan uzaklığının değişmesiyle oluşacak boyut farklılıklarını telafi etmek için.
- **Fotometrik Artırmalar:**
    - *Parlaklık (Brightness):* Ortamdaki aydınlatma seviyesinin değiştiği senaryoları modellemek için.
    - *Doygunluk (Saturation):* Malzeme yüzeyindeki renk tonu farklılıklarına karşı duyarsızlık kazandırmak için.
    - *Gaussian Gürültüsü (Noise):* Kamera sensöründen kaynaklanabilecek olası elektronik gürültüye karşı modelin kararlılığını artırmak için.

Bu stratejilerle temel veri seti yaklaşık 6 kat zenginleştirilerek toplamda 294 görüntülük bir eğitim veri seti elde edilmiştir.

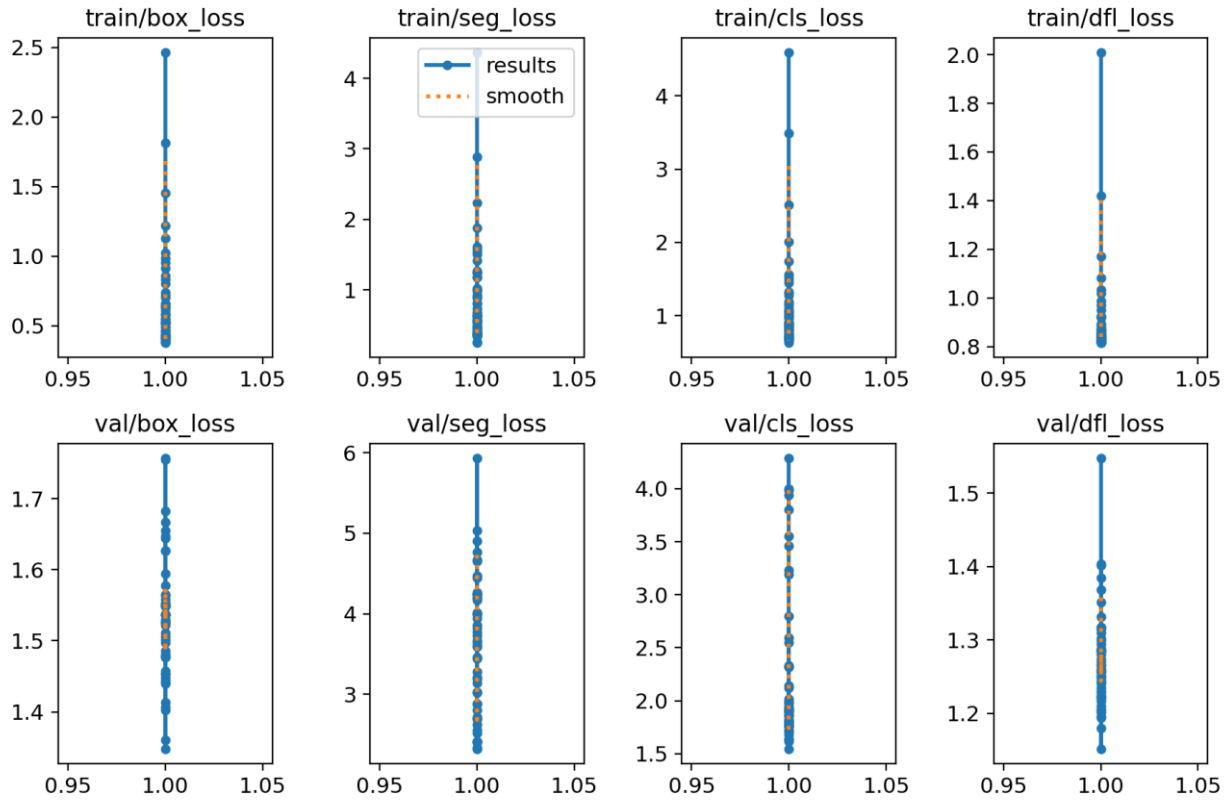
Eğitimde kullanılan temel hiper parametreler ise şunlardır:

- **Model:** yolov8n-seg.pt (ön-eğitimli, segmentasyon için nano model)
- **Epoch Sayısı:** 50
- **Yığın Boyutu (Batch Size):** 8
- **Görüntü Boyutu:** 640x640 piksel
- **Öğrenme Oranı (Learning Rate):** 0,01
- **IoU Eşiği (IoU Threshold):** 0,5

50 epoch'luk eğitim süreci sonunda, modelin doğrulama veri seti üzerindeki performansı ölçülmüştür. Eğitimin ilerlemesini gösteren F1 skoru Şekil 4.5'de ve metrik grafikleri Şekil 4.6'da sunulmuştur. Eğitimin sonunda elde edilen nihai performans metrikleri ise Çizelge 4.2'de özetlenmektedir.



Şekil 4.5 Yapılan 50 epokluk eğitimin F1 skoru



Şekil 4.6 Yapılan 50 iterasyon (Epoch)sonuçları

Çizelge 4.2 50 İterasyon (Epoch) eğitilmiş modelin eğitim çıktıları

Metrik	Değer (ortalama)	Değer (maksimum)	Değer (minimum)
Eğitim Box Kaybı	0,5	2,5	0,2
Eğitim Segmentasyon Kaybı	0,8	4,5	0,2
Eğitim Sınıf Kaybı	0,4	4,5	0,1
Eğitim DFL Kaybı	0,9	2,0	0,8
Validasyon Box Kaybı	1,5	1,8	1,3
Validasyon Segmentasyon Kaybı	2,5	6,0	2,0
Validasyon Sınıf Kaybı	2,2	4,2	2,0
Validasyon DFL Kaybı	1,2	1,6	1,1
Precision (B)	0,9	1,0	0,2
Recall (B)	0,8	1,0	0,2
Precision (M)	0,9	1,0	0,2
Recall (M)	0,85	1,0	0,2
mAP50 (B)	0,9	1,0	0,2
mAP50-95 (B)	0,45	0,5	0,2
mAP50 (M)	0,9	1,0	0,2
mAP50-95 (M)	0,45	0,5	0,2
F1 Skoru	0,80 (Güven 0,530)	0,85	0,20

Elde edilen bulgular, geliştirilen yapay zekâ işlem hattının ve seçilen modelin, hedeflenen görev için amaca uygun (fit for purpose) bir başarım sergilediğini göstermektedir. %90,5'lik mAP50(M) değeri, modelin kaynak dikişlerinin varlığını ve genel konumunu yüksek bir başarıyla tespit edebildiğini kanıtlamaktadır. Bu metrik, modelin tahmin ettiği segmentasyon

maskesinin, gerçek nesne alanıyla en az %50 oranında bir kesişime sahip olduğu durumları ölçer ve sistemin temel amacı olan kaynak bölgesinin merkezini bulma işlemi için bu doğruluk seviyesi oldukça yeterlidir. Yüksek Kesinlik (Precision) (%91.6) ve kabul edilebilir Duyarlılık (Recall) (%84.3) değerleri de bu bulguyu desteklemektedir.

Bununla birlikte, mAP50-95(M) metriğinin %38.5 gibi görece düşük bir seviyede kalması, daha derin bir analizi gerektirmektedir. Bu metrik, tespit edilen maskenin gerçek etiketle ne kadar mükemmel örtüştüğünü çok daha katı ve artan IoU eşiklerinde (%50'den %95'e) ölçtüğü için, modelin segmentasyon sınırlarındaki hassasiyetini yansıtır. Bu değerin düşük kalmasının birkaç olası nedeni bulunmaktadır:

- **Model Kapasitesi:** Bu çalışmada, gerçek zamanlı çıkarım hızına öncelik verildiği için YOLOv8 ailesinin en küçük ve en hızlı modeli olan **yolov8n-seg.pt** kullanılmıştır. Bu "nano" model, daha az sayıda parametreye sahip olduğu için, daha büyük modellere (örn. YOLOv8m-seg) kıyasla karmaşık veya düzensiz şekilli maskelerin sınırlarını piksel düzeyinde mükemmelleştirme kapasitesi daha sınırlıdır.
- **Veri Seti Homojenliği:** Başlangıç veri setinin 49 görüntü gibi küçük bir örneklemden oluşması, veri artırma teknikleri kullanılsa dahi, öğrenilecek özellik çeşitliliğini kısıtlamış olabilir. Model, genel kaynak formunu öğrenmiş ancak farklı ışık ve yüzey koşullarındaki sınır detaylarını tam olarak ezberleyememiş olabilir.

Sonuç olarak, elde edilen metrikler, seçilen hafif modelin, sistemin birincil ihtiyacı olan kaynak lokalizasyonunu başarıyla yerine getirdiğini, ancak sınır hassasiyetinde iyileştirme potansiyeli olduğunu göstermektedir. Bu durum, hız ve doğruluk arasında yapılan bilinçli bir mühendislik tercihinin sonucudur. Daha hassas maskeleme gerektiren gelecekteki çalışmalar için daha büyük bir modelle veya daha uzun süre ve optimize edilmiş hiper parametrelerle yapılacak bir eğitim, bu metriği de iyileştirebilecektir.

4.4. Sistemin Bütünleşik Performansının Değerlendirilmesi

Bu son test aşamasında, sistemin önceki bölümlerde ayrı ayrı doğrulanan tüm yeteneklerini (algılama, hesaplama, hareket, kontrol) bir araya getirerek, pratik bir endüstriyel görevi baştan sona otonom olarak tamamlama performansı niteliksel olarak değerlendirilmiştir.

Bu test için, üzerinde doğrusal ancak düzensiz ve pürüzlü geometrilere sahip çoklu kaynak dikişi bulunan, 20 mm yüksekliğinde standart bir demir profil kullanılmıştır. Sistemden, bu

kaynak dikişleri üzerinden belirlenen poligon alanının, her seferinde algılama ve işleme sürecini sıfırdan başlatarak otonom olarak işlemesi istenmiştir. Böylece yapay zekâ modelinin kaynaklı bölgenin görüntüsü üzerinde bozulma olarak sayılabileceği deformasyonlarla karşı direnci sınanmıştır. Bu metodun amacı, sistemin sadece tek bir görevi değil, aynı zamanda farklı konumlardaki hedefleri tekrar tekrar ve kararlı bir şekilde işleyebilme, yani tekrarlanabilirlik yeteneğini de gözlemlemektir.

Deney sonucunda işlenen parçalardan bir tanesi Şekil 4.23'te gösterilmektedir. Görselde, orijinal paslı ve ham kaynak yüzeyleri ile sistem tarafından otonom olarak tespit edilip taşlanarak pürüzsüzleştirilmiş ve metalik parlaklığa kavuşturulmuş bölgeler bir arada görülmektedir. Şekil 4.23'te görüldüğü gibi yapılan taşlama işlemine bir örnek verilmiştir.

Şekil 4.23 Birden fazla bağımsız taşlama prosesi sonrası aynı kaynak dikişine sahip profilin değişim görüntüsü örneği

Bu tek görsel üzerinden yapılan niteliksel analiz, tez çalışmasının temel hedeflerine ulaştığını birden fazla açıdan göstermektedir.

- *Entegrasyon ve Bütünleşik İşlem:* Taşlanmış bölgelerin varlığı, yapay zekâ tespiti, 3B konumlandırma, ofset hesaplamaları, takım yolu oluşturma ve PID kontrollü hareket

gibi birbirinden bağımsız geliştirilen tüm modüllerin, planlandığı gibi senkronize ve uyum içinde çalıştığını göstermektedir.

- *Yapay Zekâ ve Konumlandırma Doğruluğu:* İşlenmiş alanların, kaynak dikişlerinin karmaşık ve düzensiz geometrilerini hassas bir şekilde takip etmesi, yapay zekâ modelinin gerçekçi bir endüstriyel parça üzerinde doğru tespit yaptığını kanıtlar. Aynı zamanda, sistemin bu tespitlere dayanarak hesapladığı konum ve açı saptamalarının ve bunlara bağlı oluşturduğu takım yolu kinematiğinin de fiziksel olarak doğru olduğunu göstermektedir.
- *Derinlik Algısı ve Yüzey Takibinin Yeterliliği:* Her bir işlenmiş segment içindeki yüzeyin pürüzsüz ve homojen bir parlaklığa sahip olması, takımın işlem boyunca yüzeyle temasını kaybetmediğini ve aşırı basınç uygulamadığını gösterir. Bu durum, sistemin derinlik algısının yeterli olduğunu ve PID kontrolcüsünün, yüzey topolojisindeki mikro değişiklikleri başarılı bir şekilde telafi ederek sabit bir temas kuvveti uyguladığını dolaylı olarak göstermektedir.
- *Tekrarlanabilirlik ve Kararlılık:* Parça üzerindeki birden fazla farklı bölgenin, her seferinde süreç sıfırdan başlatılarak başarılı bir şekilde işlenmesi, sistemin tek seferlik bir başarı göstermediğini, aksine tekrarlanabilir ve kararlı bir yapıya sahip olduğunu ortaya koymaktadır.

Sonuç olarak, bu niteliksel bulgu, geliştirilen görüntü-yönelimli otomasyon mimarisinin, teorik altyapısını pratik bir endüstriyel göreve başarıyla dönüştürebildiğini ve hedeflenen otonom yüzey işleme kabiliyetine ulaştığını göstermektedir.

5. SONUÇLAR VE ÖNERİLER

Bu bölümde, tez kapsamında tasarlanan, geliştirilen ve test edilen bilgisayarlı görü tabanlı 4 eksenli robotik yüzey işleme sisteminden elde edilen nihai sonuçlar özetlenmekte ve bu çalışmanın bulguları ışığında, gelecekteki araştırmalar için potansiyel iyileştirme ve geliştirme alanlarına yönelik öneriler sunulmaktadır.

5.1. Sonuçlar

Bu tez çalışmasının temel amacı, endüstriyel yüzey işleme (polisaj, taşlama vb.) görevlerini otomatikleştirmek için, maliyetli donanımlara (çift kamera, lazer tarayıcı vb.) alternatif olabilecek, tek bir endüstriyel kamera kullanarak 3B konum tespiti yapabilen, esnek ve düşük maliyetli bir otomasyon mimarisi geliştirmekti. Bu hedefe ulaşmak üzere, 4 eksenli bir CNC robot, bir endüstriyel kamera ve özel bir yazılım altyapısını birleştiren bütünleşik bir sistem başarıyla tasarlanmış ve prototipi hayata geçirilmiştir.

Çalışma sonucunda elde edilen temel başarılar ve sonuçlar şu şekilde özetlenebilir:

1. *Tek Kamerayla Stereoskopik Derinlik Algılama Başarısı:* Geliştirilen harekete dayalı stereo vizyon metodolojisi, robotun kendi eksen hareketini bir ölçüm aracı olarak kullanarak, tek bir kamera ile üç boyutlu derinlik bilgisi elde etmenin mümkün ve pratik olduğunu kanıtlamıştır. Bu yöntem, ek donanım maliyetini ve sistem karmaşıklığını ortadan kaldırarak önemli bir avantaj sağlamıştır.
2. *Metrik Doğruluk için Etkin Yöntem Geliştirilmesi:* Perspektif dönüşümüne (homografi) dayalı metrik ölçekli ROI (1px ≈ 1mm) oluşturma yöntemi, karmaşık el-göz kalibrasyonuna ihtiyaç duymadan, görüntü üzerindeki piksel koordinatları ile gerçek dünya milimetre ölçüleri arasında doğrudan bir ilişki kurmayı başarmıştır. Bu, sistemin tüm ölçüm ve konumlandırma hesaplamaları için güvenilir bir temel oluşturmuştur.
3. *Entegre ve Esnek Yapay Zekâ İşlem Hattı:* Veri etiketleme, veri artırma, model eğitimi ve çıkarım adımlarının tamamını ana kontrol yazılımına entegre eden esnek bir işlem hattı başarıyla kurulmuştur. Eğitilen özel YOLOv8 segmentasyon modeli, %90,5'lik mAP50 başarım oranıyla, kaynak dikişlerinin yerini otonom işlem için yeterli bir doğrulukla tespit edebildiğini göstermiştir.
4. *Yüksek Hassasiyetli Yüzey Takibi:* Yaylı bir mekanizma ve Hall effect sensörüne dayalı olarak geliştirilen kapalı döngü **P-kontrol sistemi**, yüzeydeki ani geometri

değişikliklerine dahi son derece hızlı yanıt vererek takım basıncını sabit tutmayı başarmıştır. Ölçülen 0,008 mm'lik RMS takip hatası, hedeflenen 0,01 mm'lik hassasiyetin de ötesine geçerek, sistemin pürüzlü yüzeylerde dahi homojen bir işleme yapabildiğini kanıtlamıştır.

5. *Kabul Edilebilir Konumlandırma Doğruluğu:* Uçtan uca yapılan testlerde, sistemin nihai 3B konumlandırma hatasının 0-5 mm aralığında olduğu tespit edilmiştir. Bu hata payı, tez çalışmasının başlangıçtaki milimetre altı hedeflerinin gerisinde kalsa da geliştirilen prototipin, özellikle büyük ölçekli parçaların işlendiği birçok endüstriyel uygulama için kabul edilebilir bir başlangıç noktası sunduğunu göstermektedir.

Sonuç olarak bu çalışma; düşük maliyetli donanımlar, yenilikçi yazılım algoritmaları ve modern yapay zekâ teknikleri bir araya getirildiğinde, karmaşık endüstriyel otomasyon problemlerine yönelik esnek, modüler ve amaca uygun çözümlerin geliştirilebileceğini başarılı bir şekilde ortaya koymuştur.

5.2. Öneriler

Bu tez çalışmasında elde edilen başarıların ve tespit edilen sınırlılıkların ışığında, sistemin gelecekte daha da geliştirilmesi için aşağıdaki araştırma ve uygulama alanları önerilmektedir:

1. **Derinlik Hesaplama Doğruluğunun Artırılması:**
    - *Non-lineer Stereo Kalibrasyonu:* Mevcut durumda kullanılan ve derinliği disparite ile doğrusal bir ilişkiye sokan ampirik model yerine, daha fazla sayıda ve farklı derinliklerde veri toplayarak polinomsal veya çok değişkenli bir regresyon modeli geliştirilebilir. Bu, kameranın optik merkezinden uzaklaştıkça artan perspektif hatalarını daha iyi modelleyerek genel Z-ekseni doğruluğunu artırabilir.
    - *El-Göz Kalibrasyonu (Hand-Eye Calibration) Entegrasyonu:* Sisteme, robotun el (efektör) ve gözü (kamera) arasındaki hassas geometrik ilişkiyi hesaplayan standart bir el-göz kalibrasyon prosedürü eklenebilir. Bu, ROI tabanlı ölçeklemeye olan bağımlılığı azaltarak daha sağlam bir koordinat sistemi dönüşümü sağlayabilir.
2. **Yapay Zekâ Modelinin İyileştirilmesi:**
    - *Daha Büyük Veri Seti ve Model Mimarileri:* Modelin segmentasyon sınırlarındaki hassasiyetini (mAP50-95) artırmak için, daha çeşitli ışık ve yüzey

koşullarını içeren daha büyük bir veri seti ile eğitim tekrarlanmalıdır. Ayrıca, çıkarım hızı üzerindeki etkisi de göz önünde bulundurularak, yolov8n-seg yerine yolov8s-seg veya yolov8m-seg gibi daha büyük model mimarileri denenebilir.

- *Çevrimiçi Veri Artırma:* Eğitim sırasında, her epoch'ta farklı artırma kombinasyonları uygulayan çevrimiçi (online) veri artırma teknikleri entegre edilerek modelin genelleme kabiliyeti daha da yükseltilebilir.

3. **Kontrol Stratejilerinin Geliştirilmesi:**
   - *Tam PID veya Kuvvet Kontrolü:* Mevcut P-kontrolcü yerine, kalıcı durum hatalarını sıfırlamak için İntegral (I) terimi ve sistemin tepkisini yumuşatmak için Türevsel (D) terimi eklenerek tam bir PID kontrolcüsü tasarlanabilir. Daha ileri bir adım olarak, Hall Effect sensörünün dolaylı ölçümü yerine, takım ile iş parçası arasına yerleştirilecek bir 6-eksenli Kuvvet/Tork (Force/Torque) sensörü entegre edilebilir. Bu, doğrudan kuvvet kontrolü yapılmasına olanak tanıyarak çok daha hassas ve akıllı bir yüzey işleme prosesi sunar.
4. **Otonom Yol Planlamanın Genişletilmesi:**
   - *Tam Yüzey Tarama:* Mevcut durumda takım, tespit edilen poligonun orta hattını takip etmektedir. Yazılım, sadece bir orta hat yerine, poligonun tüm iç alanını belirli bir bindirme (overlap) oranıyla tarayacak raster (tarama) veya zikzak takım yolları otomatik olarak üretecek şekilde geliştirilebilir. Bu, sistemin basit bir çizgi takip görevinden, tam bir otonom CAM (Bilgisayar Destekli İmalat) çözümüne evrilmesini sağlayacaktır.

6. KAYNAKLAR

Arnold, E., Al-Jarrah, O. Y., Dianati, M., & Fallah, S. (2019). A survey on 3D object detection methods for autonomous driving applications. IEEE Transactions on Intelligent Transportation Systems, 20(10), 3782–3795. doi: 10.1109/TITS.2019.2892405

Åström, K. J., & Hägglund, T. (1995). PID controllers: Theory, design, and tuning (2. bs.). Research Triangle Park, NC: ISA.

Bailey, D., & Wright, E. (2003). Practical SCADA for industry. Oxford, UK: Newnes.

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934. doi: https://doi.org/10.48550/arXiv.2004.10934

Bradski, G. (2000). The OpenCV library. Dr. Dobb's Journal of Software Tools.

Bruyninckx, H. (2001). Open robot control software: The OROCOS project. In Proceedings of the 2001 IEEE International Conference on Robotics and Automation (pp. 2523–2528). doi: 10.1109/ROBOT.2001.933002

Campos, C., Elvira, R., Gómez, J. J., Montiel, J. M. M., & Tardós, J. D. (2021). ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM. IEEE Transactions on Robotics, 37(6), 1874–1890. doi: 10.1109/TRO.2021.3075644

Chang, J.-R., & Chen, Y.-S. (2018). Pyramid stereo matching network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5410–5418). doi: https://doi.org/10.48550/arXiv.1803.08669

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. In Proceedings of the European Conference on Computer Vision (pp. 801–818). doi: https://doi.org/10.48550/arXiv.1802.02611

Corke, P. (2017). Robotics, vision and control (2. bs.). Cham, İsviçre: Springer. doi: 10.1007/978-3-319-54413-7

Craig, J. J. (2018). Introduction to robotics: Mechanics and control (4. bs.). Upper Saddle River, NJ: Pearson.

Dong, X., Garratt, M. A., Anavatti, S. G., & Abbass, H. A. (2022). Towards real-time monocular depth estimation for robotics: A survey. IEEE Transactions on Intelligent Transportation Systems, 23(10), 16940–16961. doi: 10.1109/TITS.2022.3160741

Du, G., Wang, C., Li, F., & Lu, G. (2021). Vision-based robotic grasping from object localization, object pose estimation to grasp estimation: A review. Artificial Intelligence Review, 54, 1677–1734.

Fan, Z., Zhang, J., Zhang, Y., & Li, H. (2022). Deep learning on monocular object pose detection and tracking: A comprehensive overview. ACM Computing Surveys, 55(1), 1–40. doi: https://doi.org/10.1145/3524496

Forsyth, D. A., & Ponce, J. (2002). Computer vision: A modern approach. Upper Saddle River, NJ: Prentice Hall.

Fu, H., Gong, M., Wang, C., Batmanghelich, K., & Tao, D. (2018). Deep ordinal regression network for monocular depth estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 2002–2011). doi: https://doi.org/10.48550/arXiv.1806.02446

Gangal, A., et al. (2020). Complete scanning application using OpenCV (Teknik Rapor). doi: https://doi.org/10.48550/arXiv.2107.03700

Gonzalez, R. C., & Woods, R. E. (2018). Digital image processing (4. bs.). Upper Saddle River, NJ: Pearson.

Godard, C., Mac Aodha, O., Firman, M., & Brostow, G. J. (2019). Digging into self-supervised monocular depth estimation. In Proceedings of the IEEE International Conference on Computer Vision (pp. 3828–3838).

Groover, M. P., Weiss, M., Nagel, R. N., & Odrey, N. G. (1986). Industrial robotics: Technology, programming, and applications. New York, NY: McGraw-Hill.

Guizilini, V., Vasiljevic, I., Chen, D., Ambruș, R., & Gaidon, A. (2023). Towards zero-shot scale-aware monocular depth estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 9233–9243). doi: https://doi.org/10.48550/arXiv.2306.17253

Hartley, R., & Zisserman, A. (2004). Multiple view geometry in computer vision (2. bs.). Cambridge, UK: Cambridge University Press.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2020). Mask R-CNN. IEEE Transactions on Pattern Analysis and Machine Intelligence, 42(3), 386–397. doi: https://doi.org/10.1109/TPAMI.2018.2844175

Jocher, G., Chaurasia, A., & Qiu, J. (2023). YOLOv8: Real-time object detection (Ultralytics Preprint).

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. Journal of Basic Engineering, 82(1), 35–45. doi: https://doi.org/10.1115/1.3662552

Ke, X., et al. (2022). Review on robot-assisted polishing: Status and future trends. Robotics and Computer-Integrated Manufacturing, 80, 102482. doi: 10.1016/j.rcim.2022.102482

Kubáček, V. (2024). Utilization of 3D vision system for robotic sanding with force feedback control (Yüksek Lisans Tezi). Czech Technical University in Prague.

Laga, H., Jospin, L. V., Boussaid, F., & Bennamoun, M. (2022). A survey on deep learning techniques for stereo-based depth estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(4), 1738–1764. doi:10.1109/TPAMI.2020.3032602

Labbe, Y., Carpentier, J., Aubry, M., & Sivic, J. (2020). CosyPose: Consistent multi-view, multi-object 6D pose estimation. In Proceedings of the European Conference on Computer Vision (pp. 574–591). doi: https://doi.org/10.48550/arXiv.2008.08465

Lee, K. (1999). Principles of CAD/CAM/CAE systems. Reading, MA: Addison-Wesley.

Lee, S.-C., et al. (2021). A camera-based position correction system for autonomous production line inspection. Sensors, 21(13), 4380. doi: https://doi.org/10.3390/s21124071

Lipson, Z., Teed, Z., & Deng, J. (2021). RAFT-Stereo: Multilevel recurrent field transforms for stereo matching. In Proceedings of the International Conference on 3D Vision (pp. 218–227). doi: https://doi.org/10.48550/arXiv.2109.07547

McGinnis, S. (2019). GRBL: An open-source G-code parser for CNC motion control [Bilgisayar Yazılımı]. GitHub. https://github.com/gnea/grbl

Marr, D. (1982). Vision: A computational investigation into the human representation and processing of visual information. San Francisco, CA: W. H. Freeman.

Microsoft, Facebook, et al. (2017). Open Neural Network Exchange (ONNX). https://onnx.ai/

Montgomery, D. C., Peck, E. A., & Vining, G. G. (2012). Introduction to linear regression analysis (5. bs.). Hoboken, NJ: John Wiley & Sons.

Muñoz, R. A. (2019). Vision-based grasping strategy for the determination of stable grasping points (Yüksek Lisans Tezi). Universidad Politécnica de Madrid, Madrid, İspanya.

Nicholson, J. I. (2022). Deep learning techniques to estimate 3D position in stereoscopic imagery (Yüksek Lisans Tezi). Air Force Institute of Technology (AFIT), Wright-Patterson AFB, OH.

Nof, S. Y. (Ed.). (1999). Handbook of industrial robotics. New York, NY: John Wiley.

Nordh, J., & Vikén, M. (2021). Self-supervised stereo depth estimation (Yüksek Lisans Tezi). Chalmers University of Technology, Gothenburg, İsveç. uri: https://hdl.handle.net/20.500.12380/303652

Pan, S. J., & Yang, Q. (2009). A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering, 22(10), 1345–1359.

Ranftl, R., Bochkovskiy, A., & Koltun, V. (2021). Vision transformers for dense prediction. In Proceedings of the IEEE International Conference on Computer Vision (pp. 12179–12188). doi: https://doi.org/10.48550/arXiv.2103.13413

Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., & Koltun, V. (2022). Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. IEEE Transactions on

Pattern Analysis and Machine Intelligence, 44(3), 1623–1637. doi: 10.1109/TPAMI.2020.3019967

Sansoni, G., Trebeschi, M., & Docchio, F. (2009). State-of-the-art and applications of 3D imaging sensors in industry, cultural heritage, medicine, and criminal justice. Sensors, 9(1), 568–601. doi: https://doi.org/10.3390/s90100568

Schell A. (2020). Vision-based guidance for robot assisted endovascular interventions (Yüksek Lisans Tezi). University of Twente, Enschede.

Sheridan, T. B. (1992). Telerobotics, automation, and human supervisory control. Cambridge, MA: MIT Press.

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. Journal of Big Data, 6(1), 1–48. doi: https://doi.org/10.1186/s40537-019-0197-0

Siciliano, B., & Khatib, O. (Eds.). (2016). Springer handbook of robotics (2. bs.). Berlin, Heidelberg: Springer. doi: https://doi.org/10.1007/978-3-319-32552-1_1

Singh, A., et al. (2022). A survey on vision-guided robotic systems with intelligent learning. Cogent Engineering, 9(1), 2050020. doi: https://doi.org/10.1080/23311916.2022.2050020

Smith, S. T., & Chetwynd, D. G. (1995). Foundations of ultraprecision mechanism design. New York, NY: Gordon & Breach. doi: https://doi.org/10.1201/9781315272603

Spong, M. W., Hutchinson, S., & Vidyasagar, M. (2006). Robot modeling and control. Hoboken, NJ: John Wiley & Sons.

Sun, J., Wang, Z., et al. (2022). OnePose: One-shot object pose estimation without CAD models. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 6825–6834). doi: https://doi.org/10.48550/arXiv.2205.12257

Szeliski, R. (2022). Computer vision: Algorithms and applications (2. bs.). Cham, İsviçre: Springer.

Taşel, F. S. (2008). 3D reconstruction of a scene using stereo images (Yayınlanmamış Tez). uri: http://hdl.handle.net/20.500.12416/63

Ulrich, M., & Hillemann, M. (2023). Uncertainty-aware hand–eye calibration. IEEE Transactions on Robotics, 40(3), 573–591. doi: 10.1109/TRO.2023.3330609

Varma, G. (2020). Unsupervised monocular depth estimation: Learning to generalize (Yüksek Lisans Tezi). University of Illinois at Urbana-Champaign, Urbana, IL. uri: http://hdl.handle.net/2142/108020

Vahid, F., & Givargis, T. (1999). Embedded system design: A unified hardware/software introduction. New York, NY: John Wiley & Sons.

Xing, H., Lai, X., Chu, Y., Zhang, R., Yu, H., Wang, X., Wang, X. (2022). Depth camera-based algorithm for visual recognition and automated polishing of small ship components. Proceedings of SPIE, 12261, 122613R. doi: https://doi.org/10.1117/12.2640982

Zhang, F., et al. (2019). GA-Net: Guided aggregation net for end-to-end stereo matching. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 185–194). doi: https://doi.org/10.48550/arXiv.1904.06587

Zhang, Z. (2002). A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(11), 1330–1334). doi: 10.1109/34.888718