**T.C.**

**İSTANBUL KÜLTÜR ÜNİVERSİTESİ**

**LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ**

# BASKILI DEVRE KARTLARI İÇİN DÜŞÜK MALİYETLİ OTOMATİK KUSUR TESPİT SİSTEMİNİN GELİŞTİRİLMESİ

**YÜKSEK LİSANS TEZİ**

**Ayhan ÇALIŞKAN**

**1201050006**

**Anabilim Dalı: Elektrik-Elektronik Mühendisliği**

**Programı: Elektrik-Elektronik Mühendisliği**

**Tez Danışmanı: Dr. Öğr. Üyesi Güray GÜRKAN**

**MART 2021**

**T.C.**

**İSTANBUL KÜLTÜR ÜNİVERSİTESİ**

**LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ**

# BASKILI DEVRE KARTLARI İÇİN DÜŞÜK MALİYETLİ OTOMATİK KUSUR TESPİT SİSTEMİNİN GELİŞTİRİLMESİ

**YÜKSEK LİSANS TEZİ**

**Ayhan ÇALIŞKAN**

**1201050006**

**Anabilim Dalı: Elektrik-Elektronik Mühendisliği**

**Programı: Elektrik-Elektronik Mühendisliği**

**Tez Danışmanı: Dr. Öğr. Üyesi Güray GÜRKAN**

**Jüri Üyeleri: Prof. Dr. Tülay YILDIRIM (Yıldız Teknik Ünv.)**

**Doç. Dr. Akhan AKBULUT**

**MART 2021**

## ÖZET

Bugün, ülkemizde elektronik baskılı devre kartı (İng. printed circuit board, PCB) üretimi ve elektronik kart dizgisi (devre elemanlarının otomatik olarak lehimlenmesi) yapan firmalara bakıldığında, üretilen kartların doğruluk testleri için oldukça farklı yöntemler uygulandığı görülmektedir. Kontrollerin operatör(ler) tarafından gözle yapıldığı yöntem baskılı devre kartı dizgisi yapan firmaların çoğunda kullanılan bir çözümdür. Bu çözüm için, kritik gözlemlerin yapılacağı her bir istasyona birden fazla operatör atanır. Ancak, bu şekilde işlem oldukça uzun sürmektedir. Ayrıca insan hatası dolayısı ile devre kusurlarının %100 doğrululukla ayırt edilemeyeceği bilinmektedir. Bu tez çalışmasında, ilk denemelerde alüminyum profillere dayalı hafif bir ana iskelet ve bilgisayarla kontrol edilebilecek hareketli kamera sistemi tasarlanmıştır. Ayrıca, ana iskelete ek olarak tasarlanan hareketli kamera tutucu, PCB kart sabitleyici vb. mekanik yapılar 3B baskı, torna ve frezeleme yöntemleri ile üretilmiştir. Sistemin yazılım tarafı Python programlama dili ile geliştirilmiştir. Ancak ortaya çıkan görüntü alım yetersizlikleri nedeni ile ikinci bir tasarıma gidilmiştir. Bu tasarımda ise hareketli kamera düzeneği yerine uzaktan tüm hedef kartı görüntüleyebilecek yeterlilikte lens-kamera donanımı seçilmiştir. Bu tasarım için de Raspberry-Pi 4 bir sistemde ve Python ortamında çalışan ikinci bir arayüz geliştirilmiştir. Tüm elektronik kontrol de yine Raspberry-Pi 4 ile sağlanmıştır. Alınan görüntülerden lehimsiz, eksik lehimli ya da hatalı lehimli kartları bulabilen bir makine öğrenme algoritması eğitilmiştir. Eğitim için, günümüzdeki yapay zekâ dünyasında gerçek zamanlı nesne takibi ve nesne tespiti uygulamalarında hızlı olması sebebiyle yaygın olarak kullanılan YOLOv4 algoritması kullanılmıştır. Ortaya çıkan sistemin, eğitilen lehimleme hataları tespit modeli ile seri bir üretim hattında uygulanabilirliği test edilmiştir. Yapay zekaya dayalı tasarlanan düşük maliyetli lehimleme hatası tespit cihazının üretim hattına katkısı ve sonuçları gözlemlenmiş ve sunulmuştur.

**ABSTRACT**

In today's modern world, when we look at the companies that manufacture electronic printed circuit boards and electronic card assemblies (automatic soldering of circuit elements) in our country, it is seen that quite different methods are used in the accuracy tests of the produced electronic circuit boards. The method by which the controls are made by the operators visually is a low-cost solution used by most of the companies that manufacture printed circuit board assembly. In addition, circuit defects and solder defects cannot be tested and controlled with 100% accuracy due to human error. In this thesis, a light main frame based on aluminum profiles and a computer-controlled mobile camera system were designed in the first trials. Additional mechanical parts such as sliding camera holder, PCB tray were realized using 3D printing and milling methods. The software of the system has been developed in Python programming language. However, due to the resulting image acquisition deficiencies, a second design was made. In this design, a lens-camera equipment sufficient to view the entire target printed circuit board from a distance was selected instead of the sliding camera assembly. Also, an alternative user interface is developed in Python environment running on a Raspberry-Pi 4. A machine learning algorithm has been trained that can detect the number and location of solderless or incompletely soldered printed circuit boards from captured images. YOLOv4 algorithm which is widely used in real-time object tracking and object detection applications is used for training. Systems's applicability in real life and on the production line are tested with the trained soldering defects detection model. The contribution and results of this low-cost test device to the production line have been observed and presented.

**TEŞEKKÜR**

Tez çalışmam sürecince değerli bilgisini, tecrübelerini ve manevi desteğini esirgemeyen değerli hocam Dr. Öğr. Üyesi Güray GÜRKAN'a teşekkürü borç bilirim.

Tez süresince aynı zamanda tam zamanlı çalışmakta olduğum PCI ELEKTRONİK (Avcılar, İstanbul) firma yönetimine, başta Ar-Ge yöneticim Altan BOZOĞLU olmak üzere, projenin gerçek hayatta uygulanabilirliği konusunda sunduğu imkân ve destekleri için teşekkürlerimi sunarım.

Bu çalışmaya İKÜ-BAP2002 kodlu proje ile maddi destek sağlayan İstanbul Kültür Üniversitesi'ne de teşekkür ederim.

Ayhan ÇALIŞKAN

Mart, 2021

**İÇİNDEKİLER**

## ŞEKİLLER LİSTESİ

## KISALTMALAR LİSTESİ

BDK: Baskılı Devre Kartı

PCB: Printed Circuit Board

ICT: In-Circuit Test

FCT: Functional Circuit Test

AOI: Automatic Optical Inspection

CNN: Convolutional Neural Network

YSA: Yapay Sinir Ağı

IOU: Intersection of Union

DP: Doğru Pozitif

YP: Yanlış Pozitif

DN: Doğru Negatif

YN: Yanlış Negatif

YOLO: You Only Look Once

ROC: Receiver Operating Characteristic

AUC: Area Under Curve

# 1. GİRİŞ

Günümüz dünyasında, elektronikteki son gelişmeler baskılı devre kartlarının (BDK) imalat ve montaj hızlarında bir artışa yol açmıştır. Buna ek olarak, üretilen baskılı devre kartlarının ve elektronik bileşenlerinin (örn. Dirençler, kondansatörler, transistörler vb.) boyutları çok daha küçük hale gelmektedir. Artan talep ve üretim hızı ile, üretilen baskılı devre kartlarının montajlarının güvenilirliği ve denetimi dolayısıyla önemli bir sorun haline geldi. Üretilen baskılı devre kartlarının kartların doğruluk testleri için oldukça farklı yöntemler uygulandığı görülmektedir. Baskılı devre kartlarının yerleşim ve tasarımları bilgisayar destekli tasarım araçları ile gerçekleştirilir. Bir BDK montajının fiziksel olarak gerçekleştirilmesi ilk olarak, bilgisayar üzerinde şeması tasarlanmış devrenin tasarlanmış yolların, lehim pedlerinin, deliklerinin bulunduğu yüzeydeki devre kartının boş imalatı ile başlar. BDK imalatını, elektronik devre elemanlarının yerleştirildiği ve BDK'ya lehimlendiği BDK montaj işlemi takip eder. BDK montaj işlemi, elektronik cihaz gerçekleştirmede en önemli adımlardan biridir. Eksik veya yanlış yerleştirilmiş elektronik bileşenler, eksik veya uygun olmayan lehimleme seviyesi gibi BDK montaj hataları, mevcut BDK üretim sürecinde sıklıkla gözlemlenir. Bir elektronik cihazın işlevsel güvenilirliğini sağlamak için, tüm lehimleme işlemleri tamamlanmış bir BDK kusurlara karşı dikkatlice kontrol edilmelidir. Elektronik devre elemanlarının gözle tespit edilemeyecek seviyede küçük boyuta gelmesinin ardından lehim pedlerinin küçük boyutları nedeniyle, BDK montaj kusurları insan gözü tarafından kontrol edilemez.

Üretilen kartların doğruluk testleri için oldukça farklı yöntemler uygulandığı görülmektedir. Kontrollerin operatör(ler) tarafından gözle yapıldığı düşük bütçeli çözümler oldukça yaygındır. Bu çözüm için, kritik gözlemlerin yapılacağı her bir istasyona birden fazla operatör atanır. Ancak, bu şekilde işlem oldukça uzun sürmektedir. Ayrıca insan hatası dolayısı ile devre kusurlarının %100 doğrululukla ayırt edilemeyeceği bilinmektedir. Alternatif olarak kusur algılama cihazlarının kullanılması ise, daha yüksek maliyetli bir çözümdür. Baskılı devre kartlarında otomatik bir kusur tespit cihazı

anlamında, BDK arızaları Devre İçi Test (*İng.*In-Circuit Test, ICT), Fonksiyon Devre Testi (*İng.*Function Circuit Test, FCT) ve Otomatik Optik Muayene (*İng.*Automatic Optical Inspection, AOI) sistemleri ile tespit edilebilir. Devre içi test (ICT) var olan en sağlam BDK testi türüdür. Yüksek fiyat bunu yansıtır. Çivi yatağı testi olarak da bilinen bir ICT, elektronik ve hareketli test probları yerleşik baskılı devre kartı üzerindeki bağımsız devreleri aktif eder ve çalıştırır. BDK'nın monte edilmiş devre elemanları ve bileşenleri elektriksel özelliklerini (örn. kısa devre, açık devre, kapasitans, endüktans vb.) ölçer. Bu test, BDK'nin tasarımına uyacak şekilde yerleştirilmiş test probların kullanılmasını içerir. Bu test için her baskılı devre kartı üzerine test pedleri konulması gerekmektedir. Problar, testi başlatmak için BDK'nı prob yatağına iter. BDK, devre içi test problarının devre ile bağlantı kurmasına izin veren önceden tasarlanmış iletkenlik sağlaması için test pedlerine temas eder. Sağlam kalmasını sağlamak için bağlantıya belirli bir miktar baskı uygulanr. Fonksiyonel devre testi (FCT) bir BDK'nın hedeflenen tüm işlevini kontrol etmektedir. Üretilen kartın normal şartlarda çalışacağı ortam test üniteleri ile simüle edilir ve kartın tüm fonksiyonları test edilmektedir. Bu testin sonucunda ise elektronik devre kartlarının üzerindeki tüm devre elemanlarının güvenlik testleri ve tüketici güvenliği açısından dolayı çalışmasına etkisi olmadığı gibi bu testlerde BDK kusur tespitleri gözden kaçırılabilmektedir.

Literatürde, lehimlenmemiş boş baskılı devre kartı ve lehimlenmiş devre kartları ile ilgili BDK kusur tespitler için Otomatik Optik Muayene (AOI) algoritmalarının geliştirilmesi üzerine birçok çalışma bulunmaktadır. Eski çalışmalar örnek bir referans BDK ve test edilecek kartın görüntü karşılaştırması ile görüntü işleme yöntemleri kullanırken son çalışmalar çoğunlukla yapay sinir ağlarının (YSA) kullanımına odaklanmıştır. Dave ve arkadaşları boş BDK kusurları iki gruba ayrılmıştır. ICT yöntemi yerine, şablon görüntü karşılaştırmasına dayalı kusurları tespit edebilen gerçek zamanlı BDK görüntüleme sistemi üzerinde çalıştılar [1]. Putera, Malge ve arkadaşları [2]–[4] ek katmanlı BDK için 14 kusur tipini listelemiş ve BDK kusurlarını 2 ana kategoride, yani işlevsel ve kozmetik kusurlar halinde sunmuştur. Görüntü parçalama için test görüntüsünü referans görüntü ile karşılaştırdılar ve morfolojik yöntemler kullanarak kusur tespiti yaptılar. Eşikleme, uzamsal ve gürültü filtreleri kullanarak referans ve hatalı BDK görüntüleri arasında

görüntü çıkarma algoritmasını ve piksel bazlı aritmetik işlemleri kullandılar [5]. Lehimlenmemiş boş bir BDK'daki hata tespiti için kenarları keskinleştirmek ve gelişmiş kusur tespiti için gürültüyü etkili bir şekilde filtrelemek için makine görüşüne dayalı bir görüntü iyileştirme algoritması önerilmiştir [6]. Doniwar ve arkadaşları [7], kullanıcının MATLAB (Mathworks Inc., ABD) kullanarak test görüntülerini girebileceği ve şablon görüntülerini karşılaştırabileceği bir grafik kullanıcı arayüzü (*İng.* graphical user interface GUI) geliştirdi. Kırmızı-yeşil-maviden (RGB) gri seviyeye dönüşüm, düşük geçişli filtreleme, kenar algılama ve eşikleme algoritmalarından sonra, iki görüntü arasındaki farkı ayrıcalıklı veya (x-veya) işlemiyle bulmayı başardılar. Nattarasu ve arkadaşları tarafından dört tip çıplak BDK kusurlarının (örn. açık devre, aşındırma hatası, delik kusurları ve iğne deliği) MATLAB tabanlı grafik kullanıcı arayüzü aracılığıyla sınıflandırıldığı uygun maliyetli bir yöntem sunulmuştur [8]. Canlı görüntü elde etmek için bir sistem tasarlandı. Test ve referans RGB görüntüleri, ton doygunluk değeri (*İng.* Hue-Saturation-Value, HSV) görüntülerine dönüştürüldü ve kusur tespiti, görüntü çıkarma yöntemiyle kolayca sağlandı [9]. 14 tür kusur tespit edildi ve referans yakalama yaklaşımı kullanarak mümkün olan sınıflarda sınıflandırıldı. Algoritmaları temel olarak görüntü kaydı, ön işleme, görüntü parçalama, hata tespiti ve kusur sınıflandırması olmak üzere 5 aşamadan oluşur [10]. Wang ve arkadaşları [11] bir BDK üzerindeki delikleri kontrol ettiler. Tsai ve arkadaşları [12], periyodik olmayan model görüntüsündeki küçük BDK kusurlarını tespit etmek için görüntü çıkarma yöntemi ile kusur tespitini kullandı ve Fourier görüntü yeniden yapılandırma (*İng.* Fourier Image Reconstruction Method) yöntemini önerdi. Heriansyah ve arkadaşları [13], lehimlenmemiş boş baskılı devre kartlarında gözlemlenebilen önceden tanımlanmış kusur modellerini sınıflandırmak için sinir ağı tabanlı bir yöntem önerdiler. Bir BDK görüntüsünün referans görüntüsüne matematiksel morfoloji ve pencereleme teknikleri kullanarak dört ana bölüme ayırmış ve BDK görüntü bölümlendirme algoritması geliştirmiştir. Hata modellerinin atanması, normalizasyonu ve sınıflandırılması, ikili morfolojik görüntü işleme ve Öğrenme Vektör Niceleme (*İng.* Learning Vector Quantization (LVQ) sinir ağı kullanılarak geliştirilmiştir.

Baskılı devre kartı üzerinde elektronik devre elemanı montajlarında lehim bağlantılarının incelenmesi ile ilgili ilk çalışmalardan biri olan Kim ve Cho [14], lehim bağlantılarının

kalitesini dairesel aydınlatma tekniği ve sinir ağı sınıflandırıcı kullanarak 5 sınıfa ayırmışlardır. Hu ve Wang [15], daha hızlı bölge tabanlı evrişimli sinir ağlarına (*İng.* Faster Region Based Convolutional Neural Network, Faster-RCNN) dayalı ağı değerlendirdiler ve BDK'daki küçük kusurları daha iyi tespit etmek için özellik çıkarma için Özellik Piramit Ağlarına (*İng.* Feature Pyramid Networks, FPN) sahip ResNet50 [16]'yi kullandılar. Zhang ve arkadaşları [17], büyük bir veri setine ihtiyaç duymadan derin ayırt edici özellikleri öğrenerek yüksek hassasiyetli çıplak BDK arıza tespitini geliştirdiler.

Ong ve arkadaşları [18], farklı açılı görüntüyü karşılaştırmış ve eğik görünümün dik görünümden daha iyi olduğunu göstermişlerdir çünkü eğik görünümler, ortogonal görünümden daha yararlı bilgiler içermektedir. Redmon ve arkadaşları [19] tarafından 2016 yılında YOLO-V1 algoritması tanıtılmıştır. Lin ve arkadaşları [20], bir bilgisayar anakartında yüzey montajlı kapasitör tespiti, kapasitör değerleri ve yönleri için YOLO (You only look once) algoritmasını kullandılar. YOLO, evrişimli sinir ağına (*İng.* Convolutional Neural Network, CNN) dayalı bir tür nesne algılama yöntemidir. 90 adet kapasitör görüntüsü toplayıp YOLO algoritması ile eğiterek 78 adet eğitim için test için ise 12 adet görüntü kullandılar. Kullandıkları görüntülerin boyutları 2048x2048 YOLO algoritması ile otomatik tekrar 416x416 boyutlandırarak sonuçları gösterdiler. Li ve arkadaşları [21], BDK'da komponent tespitinde YOLO-V3 algoritmasının kullanımını gösterdi. Kümeleme algoritmasını kullandılar. Huang ve arkadaşları [22], YOLO-V3 ağ modelinin hafif geliştirilmesiyle çözülen derin öğrenme ve algılama hızı üzerine hızlı tanıma yöntemi önermiştir. Tespit testinden önce süreyi kısaltmak için gürültü ekleme, bulanıklaştırma ve parlaklık seviyesi değişimi gibi görüntü işleme yöntemlerini kullandılar. Wei ve arkadaşları [23], geleneksel BDK hatası algılama yöntemlerine göre kusur algoritması için daha güçlü kararlılık sınıflandırmasına sahiptir. Dijital görüntü işlemeye dayalı bir sınıflandırma algoritması ve evrişimli sinir ağına dayalı kusurlu sınıflandırma algoritması denenmiştir.

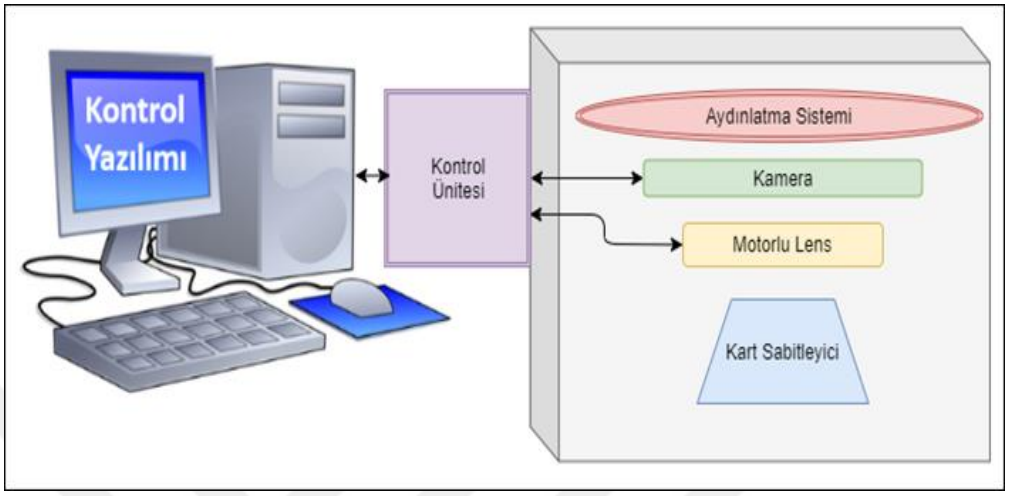
Sunulan bu tez çalışmasında literatür araştırmasına göre yapılan çalışmalara ek olarak gerçek hayata uygulanabilirliğine dikkat edilerek üretim hattında kullanılabilecek hata tespit cihazı geliştirilmiştir. Baskılı devre kartları üretiminde karşılaşılan lehimsiz devre

elemanı bacakları ve hatalı lehimlenmiş devre elemanlarının bacaklarının tespitleri için geliştirilen cihaza yönelik iki tasarıma gidilmiştir. Tasarlanan sistemlerden elde edilen sonuçların üretim hattına sağladığı avantajlar ve dezavantajlar elde edilmiştir. Avantajların yanı sıra elde edilen dezavantajların çözümleri için tasarım değişiklikleri ve işlem yükleri iyileştirilerek ve değiştirilerek elde edilen sonuçlar sayısal ve grafiksel olarak açıklanmıştır.

## 2. TASARLANAN İLK SİSTEM

Tez çalışmalarının ilk safhasında geliştirilen sistem hareketli kamera düzeneği ile hata tespiti yapılacak olan kart üzerinde gezdirilerek görüntüler alınıp tek bir görüntü elde etmek için birleştirilecek şekilde tasarlanmıştır. Baskılı devre kartlarında hata tespitini ayırt edebilmek için morfolojik dönüşümler, histogram eşitleme, ayrıcalıklı veyalama, erozyon, genişleme, kenar bulma, daire bulma ve fark alma gibi görüntü işleme teknikleri kullanılmıştır. Gri seviyelerde yapılan işlemlerin sonucunda bunların yanı sıra gürültü giderme işlemleri için ortalama alan filtreler ve eşikleme metotları ile hatalı bölge belirlenip renkli görüntüde sonuç gösterilmiştir. Sistem kontrolünü ve görüntü alımını sağlayacak olan yazılım iki kısımdan oluşmaktadır. Bunların ilki, otomatik kusur tespit sistemi üzerinde yerleştirilmiş olan gömülü sistem kartına ait yazılım, diğeri ise sistemin bağlı olduğu bilgisayarda kullanılacak yazılımdır. Donanım ve yazılım geliştirme olarak iki ana kısımdan oluşan bu sistemin blok diyagramı **Şekil 2.1**'de gösterilmektedir. Tasarlanan sistemde kullanıcı arayüzü programında her baskılı devre kartı için referans baskılı devre kartının görüntüsü tanımlanması gerekmektedir. Tanımlanacak devre kartı merkezlemesi manual olarak yapılması ve kaydedilmesi gerekir. Merkezleme işlemi manual olarak hareketli kamera sistemini kullanıcı arayüz programı üzerinden gerçek zamanlı görüntü ile kartın sağ üst köşesine getirir ve merkez noktası olarak kaydedilir. Bu kayıt işlemi ile sistem kaydedilen merkez noktasını o model için o köşeyi yatay ve dikey eksende sıfır (0, 0) noktası olarak kabul eder. Arayüz programında hata tespiti yapılacak referans görüntü her seçildiğinde sistem otomatik olarak daha önce seçilen modelin merkez noktasını bildiğinden otomatik sıfır noktasına gidecektir. Kayıt işlemi tamamlandıktan sonra kart ölçüleri arayüz programı üzerinden sisteme girilir ve tamamlanır. Bu işlem referans olarak kaydedilen baskılı devre kartının her modeli için bir defa gereklidir. Hareketli sistem seçilen her kartın merkezini bildiğinden dolayı kaç adım ilerlemesi gerektiğini milimetre'den adıma dönüştürerek mekanik ve yapısal özellikleri

kullanarak hesaplar. Böylece küçük boyuta sahip olan kartlar için sistem tüm alanı taramaz seçilen kartın ölçüsünü görüntü içerisine alacak kadar görüntü alıp ardı ardına toplar.

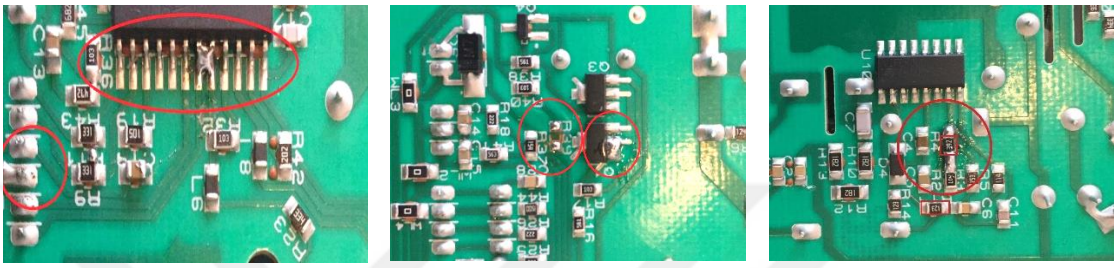



**Şekil 2.1: Sisteme ait blok şema**

Baskılı devre kartlarında oluşabilecek kusurlar, işlevsel ve kozmetik olarak iki gruba ayrılabilir. İşlevsel hatalar, baskılı devre kartının performansını ciddi şekilde etkileyebilir veya arızalanmasına neden olabilir. Kozmetik kusurlar kartın dış görünümünü etkiler, anormal ısı ve akım dağılımı nedeniyle performansını ancak uzun vadede tehlikeye atabilir.

Literatürde baskılı devre kartlarının incelenebilmesi için yapılmış çalışmalar, genellikle kozmetik kusurların algılanmasını/sınıflandırılmasını sağlayan görüntü işleme algoritmalarını kapsamaktadır. Wu ve arkadaşları [24], doğru üretilmiş bir devre kartından elde ettikleri şablon (referans) görüntü ile test edilecek kartlardan aldıkları görüntüleri, önceden belirledikleri kusur tiplerine göre kıyaslamışlardır. Kıyaslama, iki görüntüye ait matrislerin cebirsel farkın hesaplanmasının ardından kestirilebilen yedi farklı kusur şablonuna indirgenmiştir. Borba ve arkadaşları [25] ise herhangi bir referans görüntü kullanmadan, sadece kontrol edilen kartta oluşabilecek bakır bölgeler (devre yolları ve pedler) ile ilgili kusur kestirimi çalışmışlardır. Devre yollarında gerekenin %30'undan az olan genişlikler, ped bölgelerinde gerekenenin 2/3'ü kadar bakır

alanı olması ve iki eleman arasındaki mesafenin gerekenin %30'undan daha az olması gibi 3 ayrı kural tanımlamışlardır. Tsai ve ark [26], doku eşleştirme uygulamalarında oldukça kullanılan ancak ağır işlem gücü gerektiren "normalize çapraz korelasyon yöntemi"ne alternatif olarak 2B gri seviye referans ve test görüntülerden elde ettikleri 1B kantil (*İng. quantile*) verilerini karşılaştırmışlardır. Benzerlik ölçeği olarak Ki-kare test sonuçlarındaki anlamlılık temel alınmıştır.

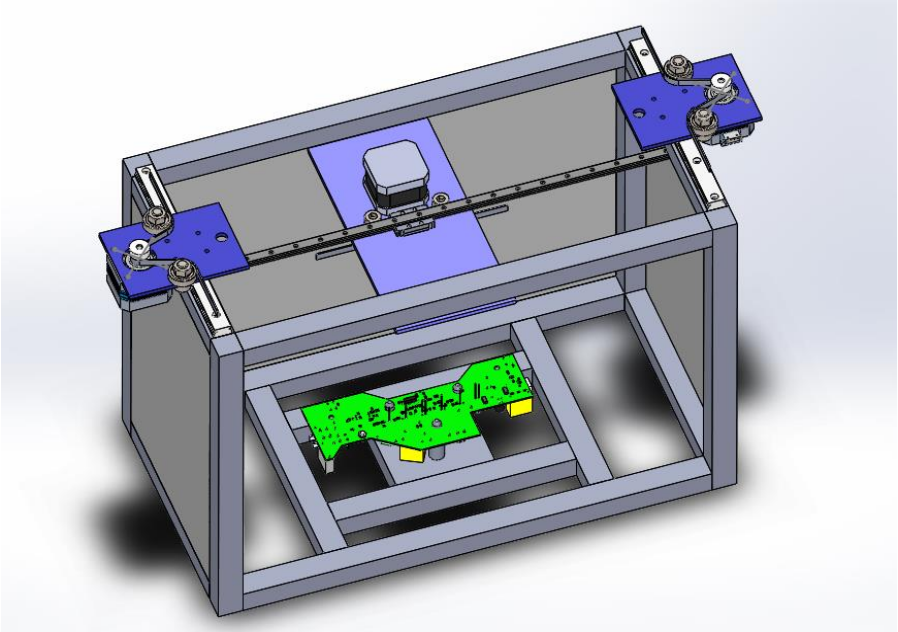


**Şekil 2.2: Kusurlu devre kartı örnekleri**

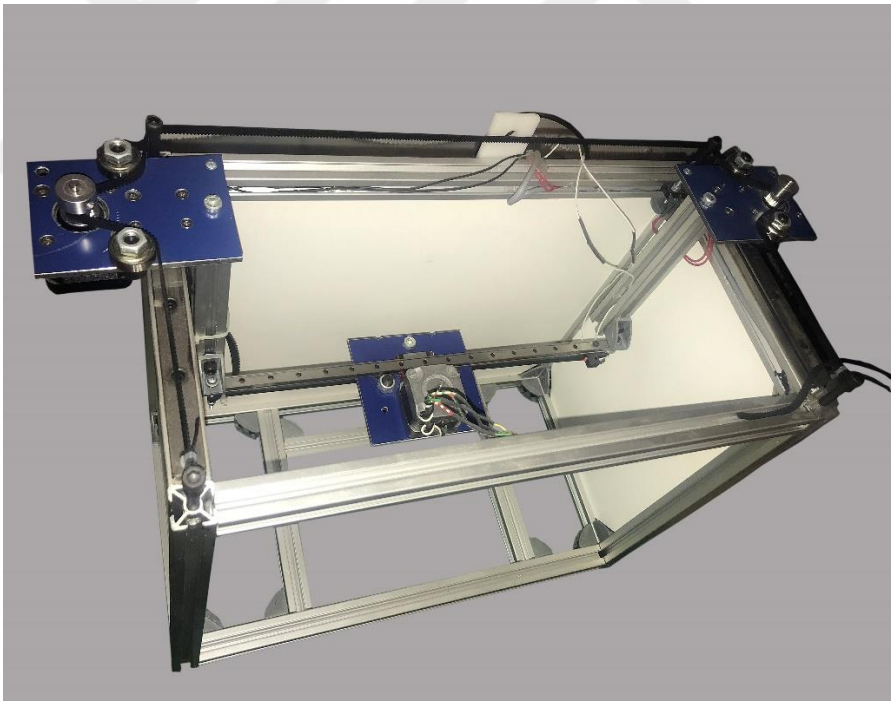
Singh ve arkadaşları [27] yaptıkları çalışmada eksik veya BDK'ya hatalı yerleştirilmiş elemanları tespit etmişlerdir. Kaur ve arkadaşları [28] bir hataların tespiti ve sınıflandırmada için referans bir görüntünün okunup saklandığı ve test edilen BDK görüntüsünün, kaydedilen görüntü ile karşılaştırıldığı bir yöntem önermişlerdir.

## 2.1 Mekanik Tasarım

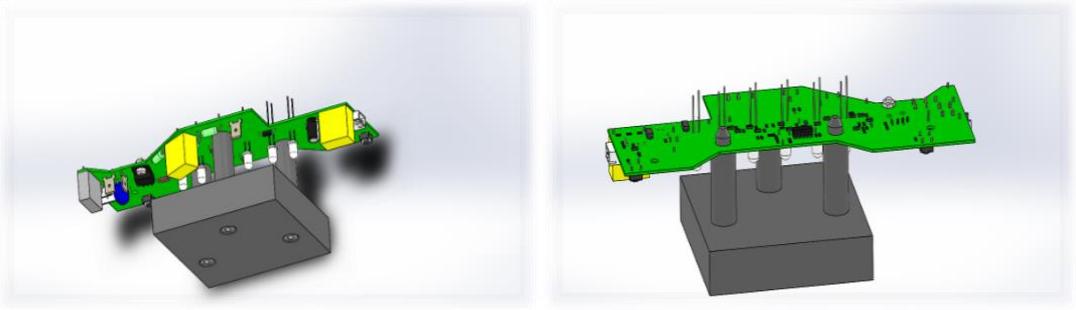
İlk tasarlanan sistemin görüntü alınmasından sorumlu donanım kısmı temel olarak üç kısımdan oluşmaktadır: 1) Cihaz İskeleti, 2) Aydınlatma Sistemi, 3) Hareketli Görüntüleme Sistemi. FreeCAD programı ile tasarım yapılmış olan cihaz iskeleti 25x25 alüminyum profil kullanılacak şekilde tasarlanmıştır (**Şekil 2.3**). Tasarım işlemi tamamlandıktan sonra adımlı motor, ray, GT2 kayış, trigger kasnak gibi malzemeler temin edilip gerekli ölçülerde elde edilmiştir. Çalıştığım firmada freezeleme, delik açma ve kesme işlemleri gibi işlemler uygulanarak sistem toparlanmıştır (**Şekil 2.4**). Hata tespiti yapılacak her model için kartı sabitleyecek bir merkezleme fikstürü yapılması gerekmektedir. İlk yapılan çalışmalarda sadece ED36 model süpürge kartı için fikstür yapılmış olup denemeler yapılmıştır (**Şekil 2.5**).



**Şekil 2.3: Sistemin FreeCAD programı üzerinde üç boyutlu görüntüsü**



**Şekil 2.4: Sistemin gerçeklenmesi**



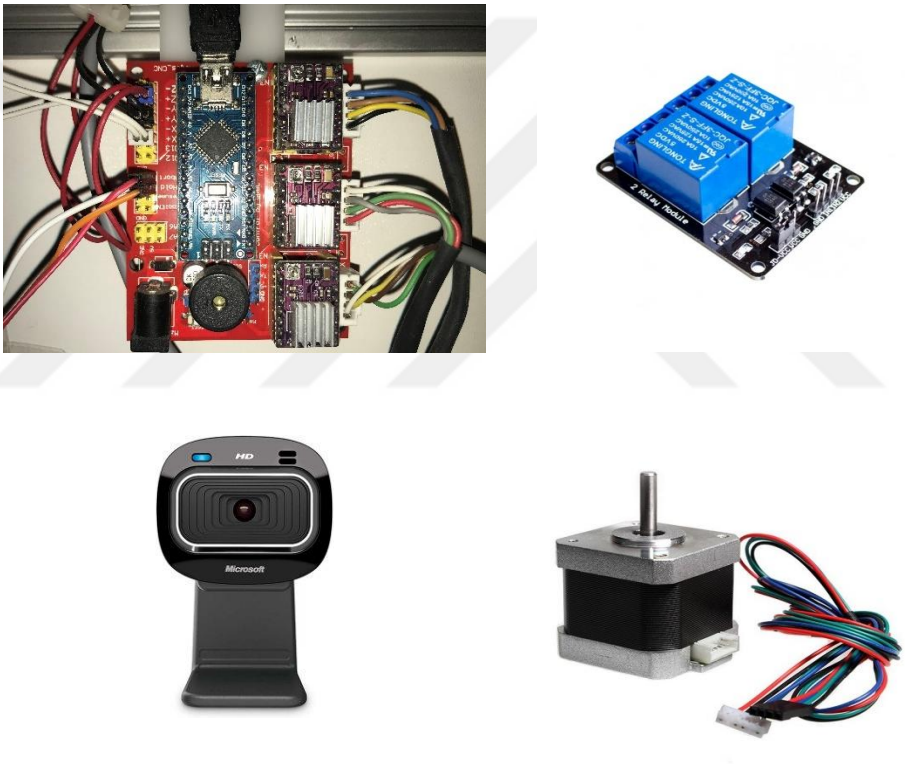
**Şekil 2.5: ED36 kodlu kart fikstürü**

Cihaza ait aydınlatma (açma/kısma/kapama) gibi fonksiyonların yazılımla kontrol edilebilmesi için temel bir gömülü sistem kullanılmıştır. Kullanıcının bilgisayarında bulanacak yazılımla seri port üzerinden haberleşecek olan sistem için Atmega328 yongası kullanılmıştır. Cihaz iskeleti tamamlandıktan sonra aydınlatma ünitesi tasarlanmıştır. Aydınlatma kalitesi ve yöntemi alınan görüntü kalitesini doğrudan etkilemektedir. Örneğin, ışık kaynaklarının, alınan elektronik devre kartı görüntüsünde görünmemesi gerekmektedir. Üretilen devre kartlarının parlak yüzeyi düşünüldüğünde, dolaylı ve yumuşatılmış yansıma tarzı ile aydınlatma yapılması gerekmektedir. Bu yüzden, sistemin içyapısını oluşturacak akrilik (pleksi-glas) tabakalar mat ve ışığı yumuşatan bir malzeme ile kaplanmıştır. Görüntü yakalama sistemi için USB çıkışlı yüksek çözünürlük veren sensör modülü kullanılmıştır. Görüntü sensör modülleri, pozlama süresi, beyaz dengesi, parlaklık, zıtlık gibi ayarların yazılımla kontrolüne olanak sağlamaktadır.

## 2.2 Elektronik Tasarım

Geliştirilen ilk sistemde kullanılan elektronik elemanlar kameranın hareketi için 3 adet adımlı motor (NEMA 17 EINSTRONIC-17HS1352), adımlı motorları sürebilmek için 3 adet çok eksenli motor sürücü kartı (DRV8825), aydınlatma sistemi kontrolü için 5 V röle kartı ve görüntü alımı için ise bir web kamerası (Microsoft LifeCam HD-3000) kullanılmıştır. Kamera 1280 x 720 çözünürlüğünde ve 16:9 formatında görüntü verebilmektedir. Seçtiğimiz adımlı motorların tam turu 200 adımdır. DRV8825 adımlı motor sürücüsünde mikroadım çözünürlükleri hassas bir şekilde ayarlanabilmektedir. 1/16 mikroadım çözünürlüğü seçilerek 200 adım 3200 adıma çıkartılmıştır. Görüntüde

1280 piksele denk gelen sutün da maksimum tarama yapıldığında 1280 piksel 6500 adıma denk gelmektedir. Bu adım başına 0.196 piksel hareket demektir. Kameranın görüş alanı (*İng.* Field of View) 20 cm yükseklikte yatay eksende 30 mm dikey eksende 17 mm'dir. Çevrimiçi GT2 tip kasnak hesaplayıcı programı ile kullanılan kayış ve kasnak özellikleri gibi parametre girişleri yapıldıktan sonra hesaplandığında 80 mm 400 adıma eşdeğerdir. Mikroadım çözünürlüğü kullanarak görüntüde elde edilecek kayıp en düşük orana indirgenmiştir. Yani 1/16 mikroadım çözünürlüğü seçildiğinde 40 mm 3200 adıma karşılık gelmektedir. Mekanik düzeneğin yapısı yatay eksende 150 mm, dikey eksende ise 80 mm'lik görüş alanına izin vermektedir.

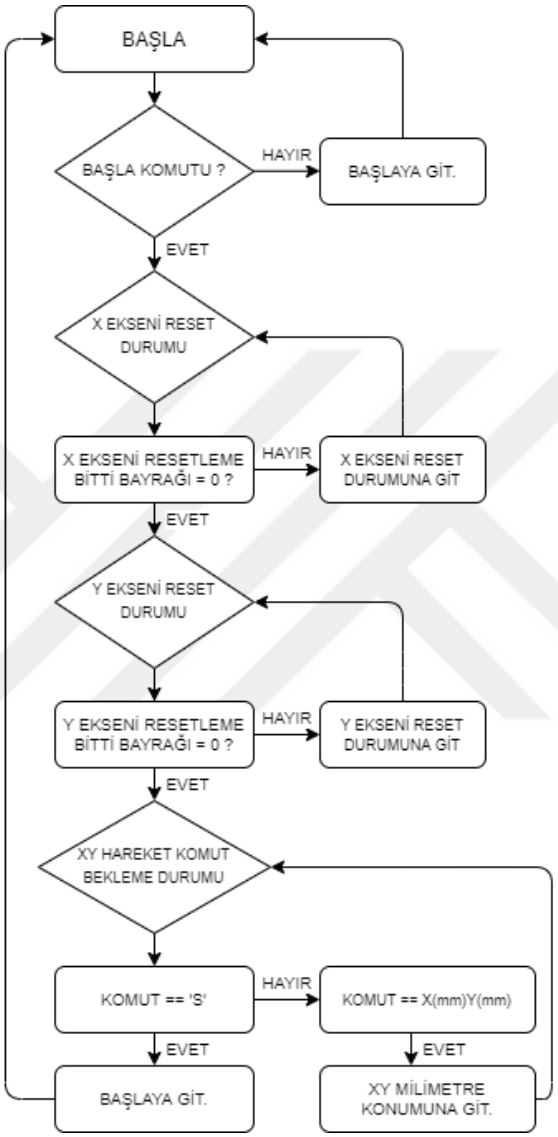



**Şekil 2.6: İlk tasarımda kullanılan elektronik donanımlar**

## 2.3 Geliştirilen Grafik Kullanıcı Arayüzü

Gömülü sistem kontrol ünitesi, C programlama dili ile programlanmıştır. Bilgisayarda bulunacak olan kullanıcı arayüzü yazılımı ise Python programlama dili ve Qt Designer

programları kullanılarak geliştirilmiştir. Görüntü sensör modülü ve aydınlatma sistemi yazılım ile kontrol edilmektedir. Ayrıca alınan görüntülerin kaydedilmesi ve görüntülerin

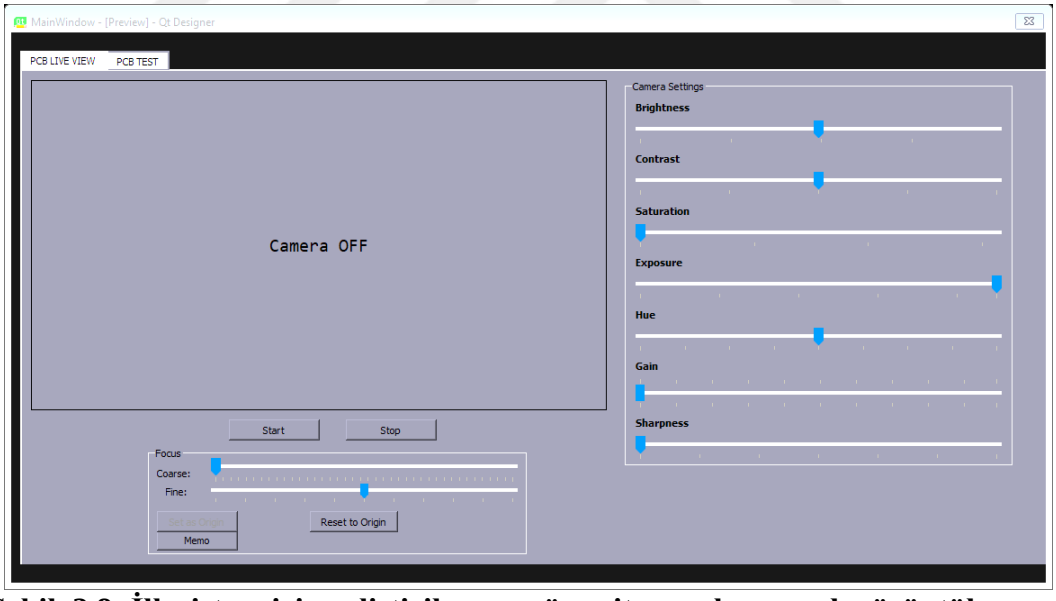



**Şekil 2.7: Atmega328 durum makinesi blok diyagramı**

işlenmesi gibi ileri işlemler Pycharm editörü kullanılarak OPENCV modülü eklenip Python programlama dili ile yapılmıştır. Kullanılan Atmega328 model mikroişlemci

donanımsal olarak seri haberleşme özelliğine sahip olduğundan bilgisayar üzerindeki kullanıcı arayüzü programı ile 9600 baud hızında haberleşecek şekilde baskılı devre kartı kusur tespiti tarama süreci tanımlanmıştır. Adımlı motorları kontrol edecek ve C programlama dili ile geliştirilen durum makinesi programının blok diyagramı **Şekil 2.7**'de gösterilmiştir.

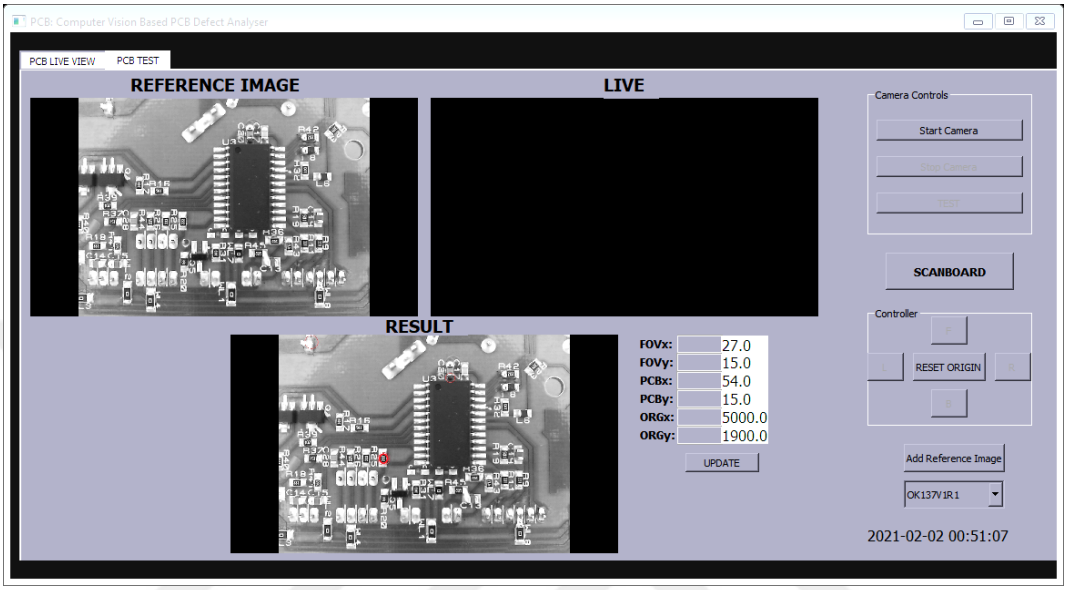
Görüntü işleme algoritmasının geliştirilmesinde literatürdeki farklılıklar, görüntü alım tarzından ziyade, kıyaslama algoritmalarından kaynaklanmaktadır. Özgün algoritma geliştirilirken, referans görüntüsü kullanılacak ve test devre kartı görüntüsü ile olan farklılıkların hızlı ve olabildiğince detaylı kestirilmesi hedeflenmiştir. Kusur tespit algoritması, MATLAB veya benzeri lisans gerektiren platformdan ziyade açık kaynak kodlu Python platformunda geliştirilmiştir. Tasarımı ve yazılımı yapılan kullanıcı arayüzü programı gerçek zamanlı görüntüleme yaparak model ile ilgili parametreleri ayarlamaya olanak sağlar (**Şekil 2.8**) ve ayarlanan bu parametreleri konfigürasyon dosyasına kaydeder.




**Şekil 2.8: İlk sistem için geliştirilen arayüze ait gerçek zamanlı görüntüleme ve kamera ayarları sayfası**

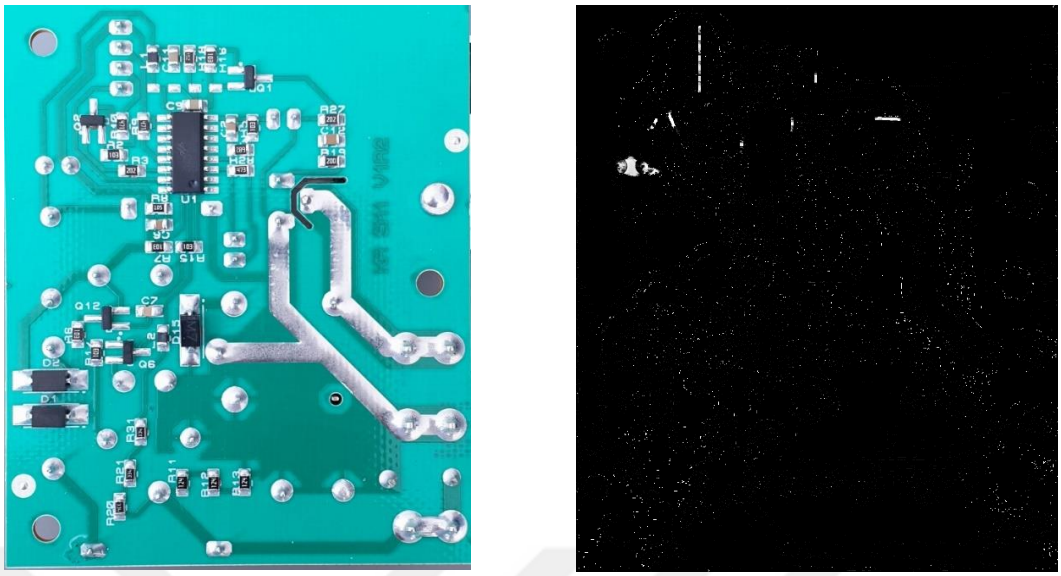
Program her çalıştırıldığında "config.ini" dosyası okunur, kaydedilen pozlama süreleri, parlaklık ve patlama gibi kamera ayarları yüklenir, programa kaydedilen referans görüntü ayarları (kart modeli adı, merkezleme koordinatı, x ve y ekseninde kart ölçüleri)

okunduktan sonra program açılır. Combobox kutusundan kart modeli seçildikten sonra referans görüntü otomatik olarak gri seviyede ekrana gelir (**Şekil 2.9**). Arayüzde görünmekte olan kart yazarın çalışmakta olduğu firmada üretim hattından alınmış ve test edilmiş kahve makinası haberleşme kartıdır.

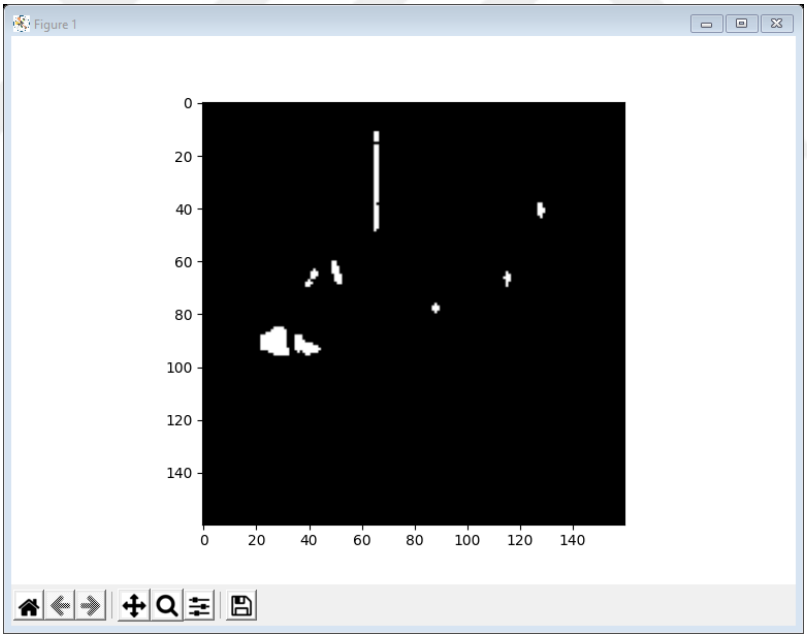



**Şekil 2.9: Test sayfası ve model seçimi**

Görüntüler alındıktan sonra OPENCV kütüphanesi kullanılarak gri seviyeye dönüştürülür ve referans görüntüye bağlı olarak histogram eşitleme yöntemi kullanılır. Bu yöntem kullanıldıktan sonra iki görüntü üst üste oturacak şekilde birbirleri arasında yatay eksende ve dikey eksen arası en fazla 15 piksel olacak şekilde kaydırma işlemi yapılarak mutlak minimum fark elde edilmiştir. Bu işlemlerden sonra OPENCV kütüphanesi ile eşikleme yöntemi ve sonrasında gürültü gidermek için 5 x 5 kernel ortalama alan filtreler kullanılarak iki görüntü arasındaki oluşan farklardan dolayı gürültüler giderilmiştir. Morfolojik görüntü metodu açma kullanılarak görüntüde hata tespiti belirgin hale getirilir. Kırmızı daire içerisine alınarak kullanıcıya hata tespiti gösterilir (**Şekil 2.9**).



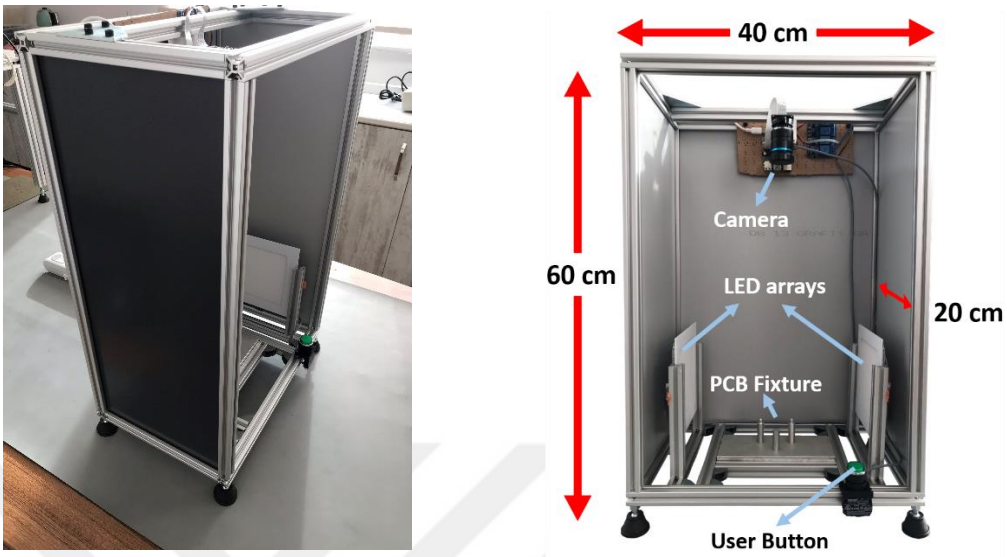
**Şekil 2.10: Referans görüntü (sol) ve farklardan oluşan ayrıcalık veyalama işlemi uygulanmış test sonucu görüntü (sağ)**




**Şekil 2.11: Ortalama alan filtre ve genişleme (dilation) yöntemi uygulandıktan sonra elde edilen farklar**

Yapılan ilk sistemde baskılı devre kartlarında referans görüntüye bağlı olarak kart üzerindeki farklılıklar kullanılan görüntü işleme teknikleri ve eşikleme yöntemleri ile sonuçlandırılmıştır. Gürültü giderme işlemlerinde daha önce de belirtildiği gibi 5x5 kernel ortalama alan filtreler kullanılmıştır. Yapılan çalışma sonucunda sistemin baskılı devre kartı üzerindeki hata tespitlerini yani kısa devreleri ve hatalı komponentleri tespit ettiği gözlemlenmiştir. Sistemde kullanılan GT2 tip 2 mm genişlikli zamanlama kayışı ve 8 dişli alüminyum kasnak ile çok yavaş tarama yapılabilmektedir. Bir adet 120 mm x 80 mm ölçüsünde baskılı devre kartını taramak ve sonuç almak otuz saniyeyi geçmektedir. Bu sebeple sistemin doğrulunun yanısıra yavaş olması sonuçları olumsuz etkilemiştir. Üretim hattında test edilecek kartlar test edilmeye başladığında sistemin kart üzerinde parça parça fotoğraf çekip görüntü toplaması ve hepsine ayrı ayrı bakıp sonuç vermesinin üretimi yavaşlattığı gözlenmiştir. Bu sebepten, makine öğrenmesi yöntemine geçilerek evrişimli sinir ağı altyapısına dayanan YOLOv4 algoritması kullanılıp yüksek kaliteli (4056 x 3040 çözünürlüklü) kamera ile tek bir seferde baskılı devre kartının tüm görüntüsü elde edilip görüntüyü parçalar haline bölüp işlem yapılması kart üzerindeki detayları daha doğru, hızlı ve etkili bir sonuç elde edileceği düşünülmüştür. İkinci sistemde daha çok detay ve inceleme gerektiren yüzey montajlı devre elemanların açık olup olmadığı gibi lehim hataları tanımlanıp görüntü işleme yöntemlerinin yanı sıra yapay zekâ kullanılacak şekilde tasarlanıp geçilmiştir. Bu yönteme geçildiğinde adımlı motor, adımlı motor sürücü kartları ve Atmega328 yongası ortadan kalkarak maliyet minimum seviyeye indirgenmiş olup daha efektif bir mekanik ve elektronik yöntem sunulmuştur.

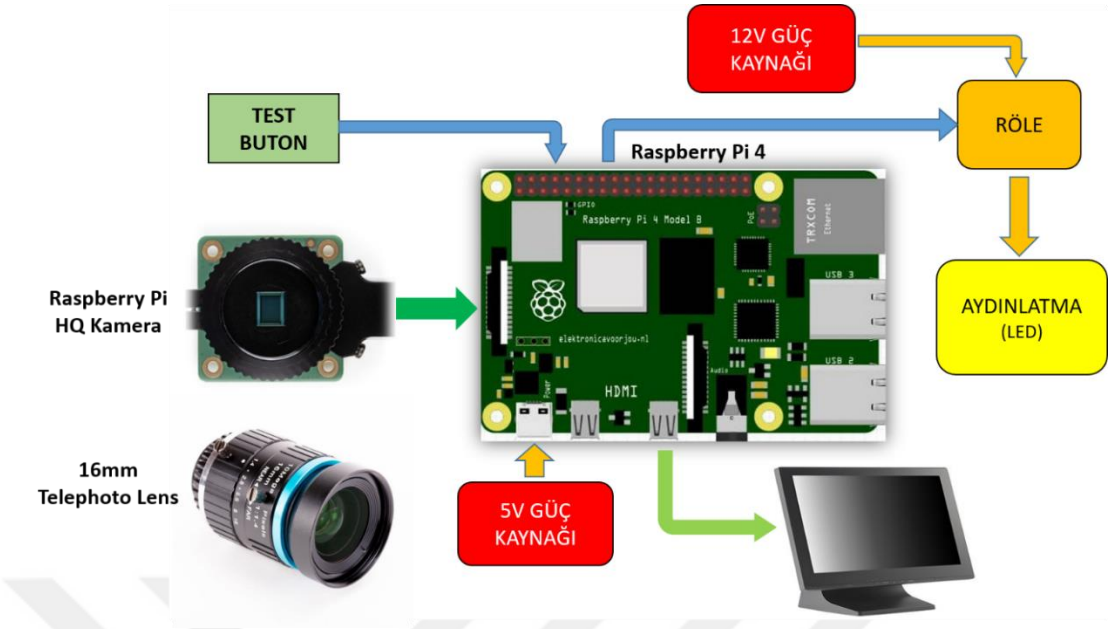
# 3. TASARLANAN İYİLEŞTİRİLMİŞ SİSTEM




**Şekil 3.1: İskelet görünümü**

İskelet oluşturulurken 18 parça sigma profil kullanılmıştır. Atölyede kesme makinası ve frezeleme makinası kullanılarak delik delme işlemleri ve kesme işlemleri uygulanmıştır. Bu işlemlerden sonra iskelet Genişlik x Yükseklik x Derinlik olacak şekilde 40mm x 60mm x 20mm ölçülerinde köşe bağlantıları ile toplanmıştır. Kompozit paneller dışarıdan gelen ışık farklılıklarını ve değişikliklerini engellemek için kesilip monte edilmiştir. Led paneller sigma profiller ile eşit ışık şiddetini sağlamak için sağ ve sol tarafa merkez olacak şekilde montajlanmıştır (**Şekil 3.1)**. Baskılı devre kartları lehim hataları tespiti için tasarlanan cihaza Full HD görüntü sağlayabilen bir monitör monte edilmiştir. Monte edilen bu monitör bu cihaz ile çalışacak operatörün hatalı kartları detaylı bir şekilde görüp hataları düzeltebilmesi için ihtiyaç duyulmuştur. Cihazın veri seti için görüntü toplayabilecek ve eğitim sonrasında tahmin ve test edebilecek son hali **Şekil 3.2'**de gösterilmiştir.

**Şekil 3.2: Cihaz görüntüsünün son hali**

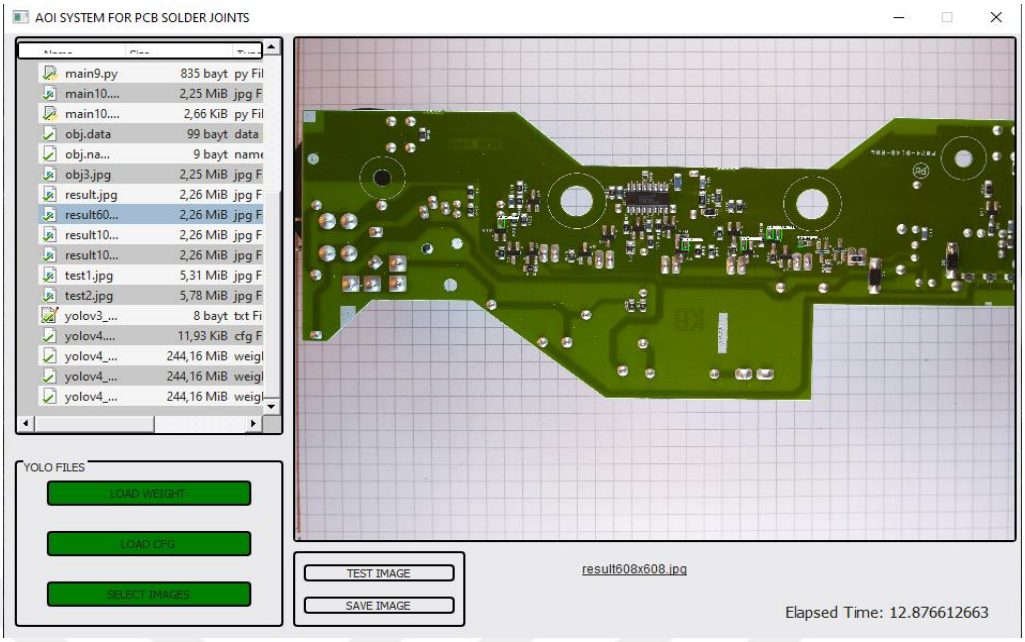



**Şekil 3.3: Elektronik bağlantı şeması**

Tüm görüntüler endüstriyel bir kamera ve Raspberry Pi 4 4GB bilgisayar aracılığıyla elde edilmektedir. Kamera kartında 1.55 µm x 1.55 µm piksel boyutunda 12.3 Megapiksel Sony IMX477 1 / 2.3" CMOS sensör bulunmaktadır. Bu sensör bir C/CS montaj adaptörü ile donatılmıştır. Görüntü sensörüne 16 mm odak uzunluğuna ve manuel odaklamaya sahip bir lens (PT3611614M10MP) monte edilmiştir. Kullanıcı etkileşimini sağlamak için kamera ve aydınlatma rölelerini kontrol etmek için bir Python kodu geliştirilmiştir. Kullanıcı, görüntü yakalama sürecini başlatmak için düğmeye basmalıdır. Düğme, Raspberry Pi cihazının GPIO-23 bacağına bağlıdır. Düğmeye basıldıktan sonra, GPIO-18 üzerindeki röle çıkışı yüksek seviyeye ayarlanır ve led panelleri yanar. 2 saniyelik led yanma süresinden sonra baskılı devre kartı görüntüsü picamera modülü ile yakalanır ve OPENCV modülü aracılığıyla 4056 x 3040 piksel (JPG uzantılı) görüntü dosyası olarak kaydedilir (**Şekil 3.4**).

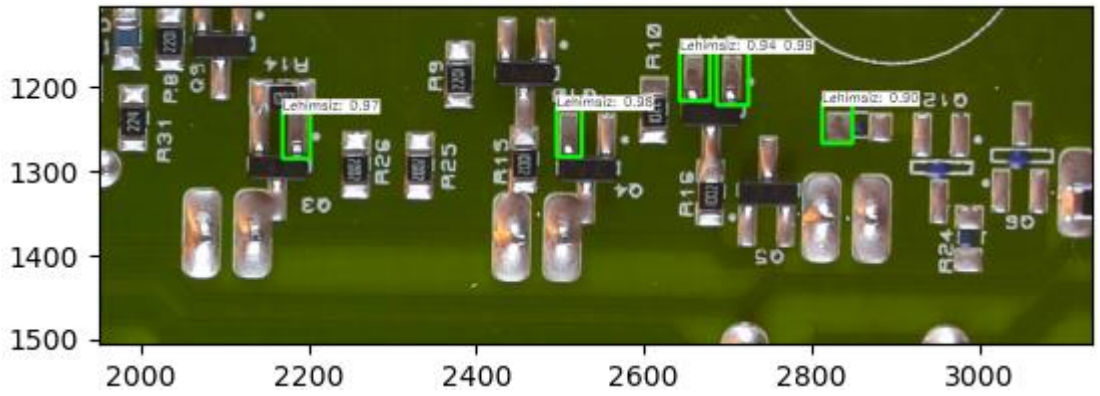
**Şekil 3.4: Görüntüleme için seçilen donanımlar**

Sunulan bu tez projesinde iki ayrı kullanıcı arayüzü geliştirilmiştir. İlk olarak, proje süresi ve eğitimin öğrenmesi sürecinde yapılan eğitim denemelerini kolayca test edebilmek için hızlıca görüntülerin tek tek veya klasör halinde yüklenebildiği kullanıcı arayüzü geliştirilmiştir. Bu kullanıcı arayüzü programına tüm YOLO versiyonlarının eğitim sonucunda elde edilen ağırlıkları ve konfigürasyon dosyaları yüklenebilmektedir. Kullanılacak konfigürasyon dosyaları ve ağırlıkları yüklendikten sonra test işlemi sonucu tahminde bulunulan test süresini ve sonuçlarını gösteren bu arayüz tasarımı kullanılmıştır. Bu programda MATPLOTLIB kütüphanesi kullanılarak gösterilen görüntü üzerinde detaylı gözlemler ve değerlendirmeler yapılmıştır (**Şekil 3.5**).




**Şekil 3.5: Tahmin denemeleri için hazırlanan arayüz görüntüsü**

Tahmin olasılıkları ve sonuçları detaylı şekilde incelenerek sistemin gözden kaçırdığı elektronik devre elemanlarının lehimsiz pedlerinin hataları incelenmiştir.



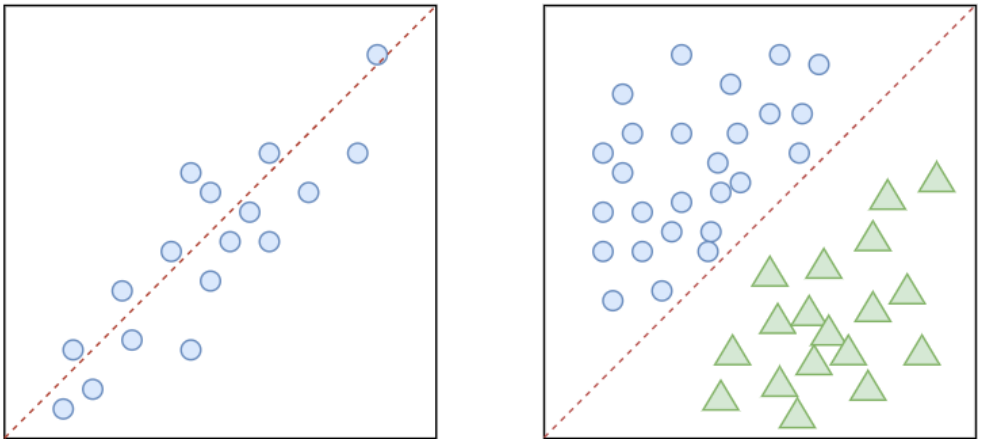

**Şekil 3.6: Örnek tahmin sonuçları**

# 4. MAKİNE ÖĞRENMESİ

Makine öğrenmesi, bilgisayarlara bilgi öğrenmeye imkân vermemiz yani onlara bir bilgiyi öğretmemiz ve onlara öğrendikleri bu bilgiyi kullanarak elde ettikleri veriler doğrultusunda, bilgisayarların yeni gelen verilere yönelik bir tahmin veya çıkarım yapmasıdır. Bunu yaparken de oluşturduğumuz modeli eğittiğimiz için yeni gelen verileri analiz ederek kategorilere ayırmasıdır. Makine öğrenmesi, çok çeşitli kavramlarla ilgilenmektedir. Yaygın olarak kullanılan makine öğrenmesi tekniklerinden en çok bilinenleri gözetimli öğrenme ve gözetimsiz öğrenmedir.

## 4.1 Gözetimli (denetimli) Öğrenme

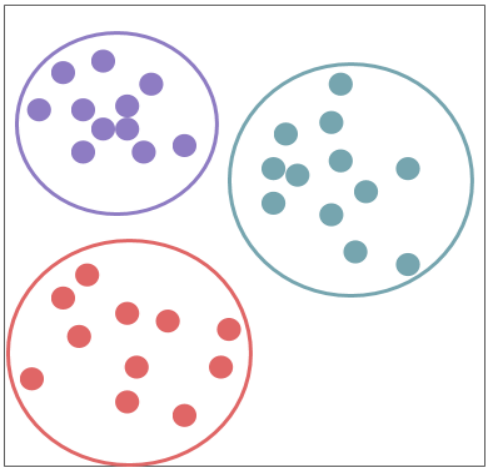
Gözetimli (denetimli) öğrenme, sınıfın özelliklerine sahip olup etiketlenmiş verilerin bulunduğu ve hedef değişkeninin verildiği öğrenme konseptidir. Bir öğrenme algoritması elde etmek için sınıflandırılmış (ve etiketli) veriler kullanılır. Etiketli verilerden değerleri tahmin etmeyi öğrenir. Örneğin, kuşlar ve balıklar. Bu öğrenme türünde sınıf sayısı belli bir eğitilmiş model bulunmaktadır. Burada eğitilen modelin doğruluğu doğru etiketlenmiş veriler demektir ve bu teknikte en önemli adım doğru veri etiketlemedir. Kendi içerisinde sınıflandırma ve gerileme (*İng.* Regression) olarak bölümlere ayrılır. Bir sınıflandırma problem olarak "sarı" veya "yeşil" gibi çıktı değişkenlerinin bir bölümde olmasına örnek verilebilir. Tek dezavantajı eğitim süresi büyük hesaplar sebebiyle uzun süreler almaktadır. Avantajı ise gerçek hayatta hesap problemlerinin çözümünde yardımcı olarak kullanılır. Gerileme problemi ise çıktı verilerinin gerçek verilerden oluştuğu yani Euro ve Dolar gibi değişkenlere denir. Sonuç olarak, girdi verisi ve bu veriye karşılık gelen çıktı değişkenleri gerileme ve sınıflandırma tahminleri olabilir.



**Şekil 4.1: Gerileme ve sınıflandırma yöntemleri**

## 4.2 Gözetimsiz (denetimsiz) Öğrenme

Makine öğrenimindeki diğer bir teknik gözetimsiz (denetimsiz) öğrenmedir. Bu öğrenme tekniğinde çıkarımlar giriş verilerinden yapılır. Giriş verilerinin özelliklerine göre kümeleme (*İng.* Clustering) yöntemi ile belirli gruplar oluşturan öğrenme biçimidir. Burada etiketlenmiş ve sınıflandırılmış bir veri yoktur. Girdi verisindeki bilgiler üzerindeki ilişkileri gruplandırıp kullanan tekniktir. Gözetimsiz öğrenme algoritması etiket olmayan bir veri kümesinden öğrenir ve verileri otomatik ayırıp gruplandırabilir.



**Şekil 4.2: Kümeleme yöntemi**

## 4.3 Evrişimli Sinir Ağları

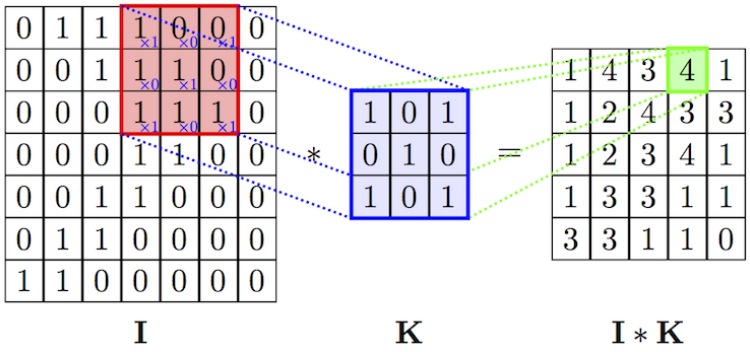
Evrişimli sinir ağları yaygın bir şekilde kullanılan görüntü ve vidyo işleme için geliştirilen bir derin öğrenme ağıdır. Görüntü tanıma, nesne sınıflandırma, yüz tanıma için oluşturulacak modeller için başta gelen kategorilerden birisi olarak makine görüntülemesinde kullanılmaktadır. Bir giriş görüntüsünü alabilen ve görüntüdeki çeşitli yönlere, öğrenebilir ağırlıklara ve önyargılara sahiptir. Nesnelere önem atayabilen ve birini diğerinden ayırt edebilen derin öğrenme algoritmasıdır. Yapay nöron katmanlarndan oluşur. Yapay nöronlar, biyolojik benzerlerinin kabaca genellersek taklidi, çoklu girdilerin ağırlıklı toplamını hesaplayan ve bir aktivasyon değeri veren matematiksel fonksiyonlardır. Bilgisayarlar girdi görüntüsünü piksel dizisi olarak görür ve görüntü çözünürlüğüne bağlıdır. Görüntü çözünürlüğü ile ilişkili olarak y x g x b (yükseklik x genişlik x boyut) değerlerini bilmektedir. Ağırlıklar tarafından her nöronun davranışı tanımlanır. Evrişimli sinir ağları piksel değeri girdileriyle işleme alındığında yapay nöronlar görsel özellikleri seçmektedir. Bir konvolüsyon katmanına görüntü girildiği zaman katmanların her biri birden fazla olabilecek şekilde etkinleştirme haritası oluşturur. Etkinleştirme haritaları, görüntünün ilgili özelliklerini vurgular. Nöronların her biri girdi olarak bir piksel yaması alır ve renk değerleri ağırlıklarla çarparak ve toplayarak aktivasyon fonksiyonu ile çalıştırır.

“ConvNets” olarak da adlandırılan modern evrişimli sinir ağları üzerine ilk başarılı sonuç veren uygulamalar 90’ların başında yapılmıştır. Yann LeCun ve arkadaşları tarafından belge tanıma uygulamasına uygulanan gradyan temelli öğrenme başlıklı makalede, basit özelliklerin yanısıra daha karmaşık özellikleri bir araya getiren bir evrişimli sinir ağı modelinin el yazısı karakter tanıma için başarı ile kullanılabileceği tanıtılmıştır [29]. 2012 yılında AlexNet [30] adlı bir evrişimli sinir ağı ImageNet (son teknoloji performans etiketleme) resim yarışmasında çok büyük ilgi görmüştür. ImageNet yaklaşık olarak 22000 kategoriye ait 15 milyondan fazla etiketli yüksek çözünürlüklü görüntüden oluşan bir veri setidir. AlexNet’in giriş boyutu sabittir ve görüntüleri 256 x 256 çözünürlüğüne yeniden ölçeklendirir.

## 4.4 Evrişim Katmanı

Evrişim katmanı evrişimli sinir ağlarının yapısını oluşturan ilk yapısıdır. Girdi olarak girilen görüntüler ilk evrişim katmanına girmektedir. Bu katman görüntüdeki yatay, dikey ve çapraz kenarlar gibi temel özellikleri algılamakla sorumludur. Evrişim işleminin görüntü girdisi 2 boyutlu bir matriste 2 boyutlu 3 x 3 kernel (**K**) filtrelesinin gerçekleştirildiği örnek **Şekil 4.3**'de gösterilmiştir.

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) \qquad (1)$$



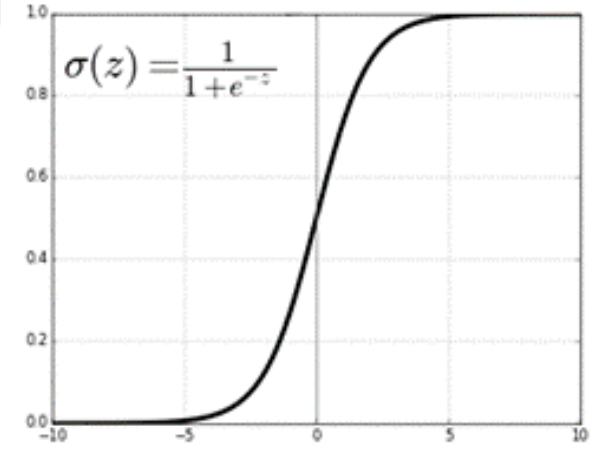

**Şekil 4.3: Evrişim işlemi**

Örnek görselde görüldüğü üzere girdi olarak kullanılan görüntünün sol üst köşesine yerleştirilen bir 2 boyutlu kernel **K** filtresi bulunur ve bu filtre tüm görüntünün üzerinde gezdirilerek evrişim işlemi eşitlik (1)'deki formül ile hesaplanır [31]. Her basamak sonunda elde edilen sonuç sol üst köşeden sırasıyla görseldeki gibi 2 boyutlu matris oluşacak şekilde işlemler sırayla yapılır. Bu hesap sonucunda elde edilen matrise özellik veya aktivasyon haritası (*İng.* Feature Map) denir.

2 boyutlu uzayda (S) görüntü girdisi **I** olarak tanımlanırsa 2 boyutlu **K** filtresi anlamına gelir. Formülde kullanılan **i** ve j parametreleri işlem sonucunda elde edilecek özellik haritasındaki konumlarını ifade eder.

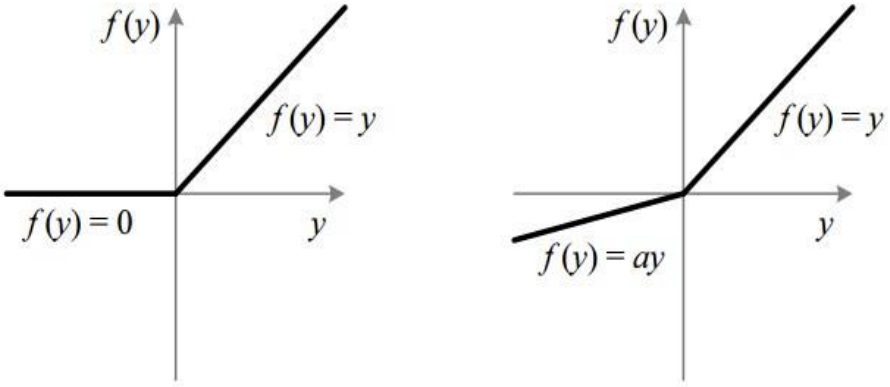
## 4.5 Aktivasyon fonksiyonları

Aktivasyon fonksiyonu bir sinir ağının çıktısını belirleyen matematiksel denklemlerdir. Aktivasyon fonksiyonunun amacı, bir nöronun çıktısına doğrusal olmama durumu katmaktır. İşlev, ağdaki her bir nörona bağlanır ve her bir nöron girişinin modelin önyargısı ile alakalı olup olmadığına bağlı olarak etkinleştirilip etkinleştirilmeyeceğini belirler. Girişten gelen bilgiler ağırlıklar ile çarpılıp sinir hücresine aktarılmaktadır. Bu hücrede çıkışa aktarılmadan önce aktivasyon fonksiyonu işlemi gerçekleştirilmektedir. Geriye yayılım algoritması kullanılırken yapılan türev işleminde ağırlıklar güncellenir. Dolayısıyla, ağırlıkların güncellenmesi öğrenmeyi sağlamaktadır. Bu işlemlerin doğru gerçekleşebilmesi için aktivasyon fonksiyonları kullanılmaktadır. Aktivasyon fonksiyonu olmayan bir sinir ağı, sadece doğrusal bir regresyon modelidir. Aktivasyon işlevi, girdiye doğrusal olmayan dönüşümü gerçekleştirerek daha karmaşık görevleri öğrenip gerçekleştirmesini sağlar. En çok bilinen lineer olmayan aktivasyon fonksiyonları Sigmoid, Düzleştirilmiş doğrusal birim katmanı (ReLU, *İng.* Rectifier Linear Unitlayer), Sızıntı ReLU ve Softmax aktivasyon fonksiyonlarıdır.




**Şekil 4.4: Sigmoid fonksiyonu**

Sigmoid fonksiyonu değeri 0 ile 1 arasına sıkışmaktadır. Bu sebeple bir olayın olma olasılığını bulan modellerde yaygın kullanılmaktadır. Bu fonksiyon $f(x)=1/(1+e^{-x})$ şeklinde tanımlanır.




**Şekil 4.5: ReLU (sol) ve Sızıntı ReLU (sağ) fonksiyonları**

Evrişimli sinir ağlarında ve derin öğrenmede, çoğunlukla ReLU fonksiyonunun hesaplama yükünün az olması sebebiyle çok katmanlı ağlarda bu fonksiyon kullanılmaktadır. Doğrusal olmayan fonksiyondur. Y pozitifse çıktı y'dir. Aksi takdirde çıkış 0'dır.

$$\alpha(x)=\begin{cases}\max(0,x) & ,x\geq 0\\ 0 & ,x<0\end{cases} \qquad (2)$$

Sızıntı ReLu fonksiyonunda ise ReLU'daki gibi 0 olan kısımda türevininde sıfır olmasından dolayı öğrenme orada gerçekleşmiyordu. Bu durumda "leaky" denilen

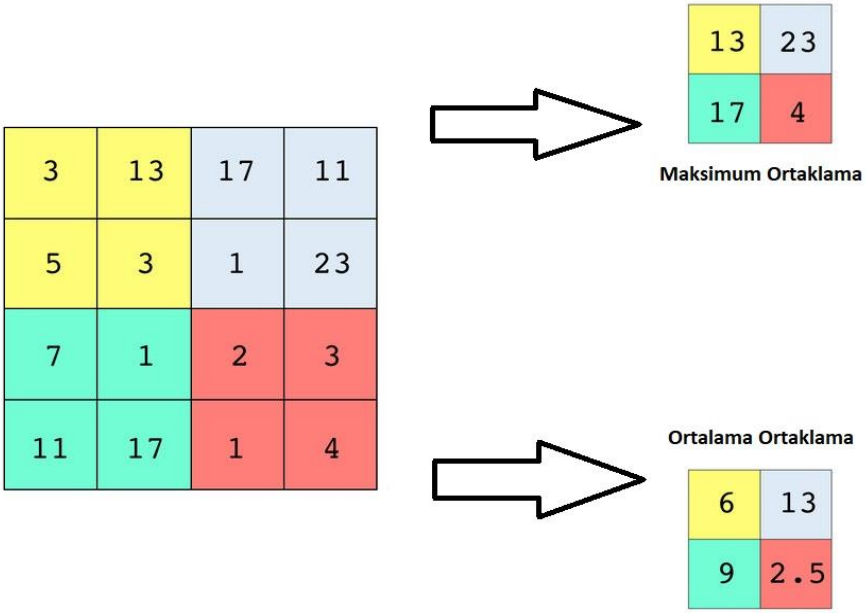
sızıntı sonsuza kadar eğimli bir şekilde gidebilmektedir. Böylece sıfıra yakın olsada o kısımlarda öğrenme yapılabilmektedir.

$$f(x) = \begin{cases} 0.01x & ,x < 0 \\ 0 & ,x \geq 0 \end{cases} \qquad (3)$$

Softmax fonksiyonu genellikle birden fazla sınıfı işlemeye çalışırken kullanılır. Her girdinin sınıfını tanımlama olasılıklarını elde etmeye çalıştığımız sınıflandırıcının çıktı katmanında kullanılır. Bu işlem her bir sınıf için çıktıları 0 ile 1 arasında sıkıştırır ve bu aralıkta olasılık hesabı döndürmektedir. Çoklu sınıflandırma sorunlarında kurulan modellerin çıktı katmanında kullanıldığı görülür.

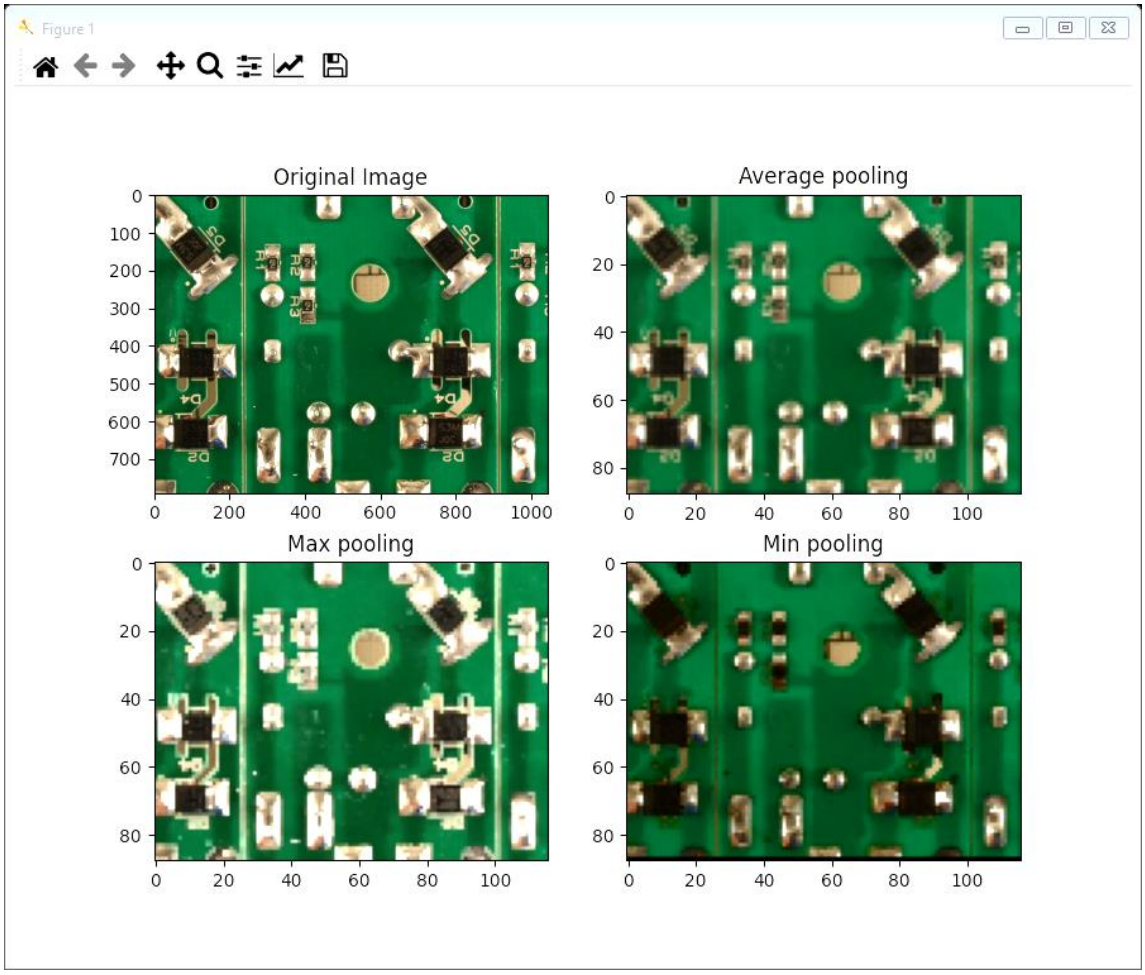
### 4.6 Ortaklama veya Havuzlama Katmanı (Pooling Layer)

Ortaklama katmanına diğer bir adıyla alt örnekleme (*İng.* downsampling) katmanı olarakta adlandırılır. Bu işlem tipik olarak evrişim katmanından sonra uygulanır Ağırlık sayılarını azaltır ve uygunluğunu kontrol ederek işlem yükünü azaltırken verileri minimum kayıp oluşturacak şekilde verir. Varyansı ve hesaplama karmaşıklığını azaltmak için sinir ağlarında kullanılır. Özellik haritasındaki özelliklerin sayısını azaltan bir tekniktir. İşlem, bir özellik haritası boyunca 2 boyutlu filtre uygulamayı içerir ve bunu yapmak, haritanın bir bölgesinde bulunan özellikleri özetlemektir. Bu ortaklama katmanında 4 tip ana tür havuzlama yöntemi vardır. Bu yöntemler maksimum havuzlama, minimum havuzlama, ortalama havuzlama ve küresel havuzlama tipleridir. Genel olarak kullanılan ortaklama katmanları maksimum ve ortalama tipleridir. Ortalama havuzlama yöntemi görüntüyü düzleştirir ve bu nedenle bu havuzlama yöntemi kullanıldığı zaman keskin özellikler tanımlanamayabilir. Maksimum havuzlama yöntemi ise görüntüden daha parlak pikselleri seçer. Görüntünün arka planı karanlık olduğunda ve görüntünün yalnızca daha açık pikselleri ile ilgilendiğimizde bu yöntem daha kullanışlı ve uygundur.




**Şekil 4.6: Maksimum ve ortalama ortaklama**

Aşağıdaki şekilde görüldüğü üzere 3 farklı havuzlama yönteminin orijinal girdi görüntüsüne uygulandıktan sonra birbirilerine göre farkları gösterilmektedir.




**Şekil 4.7: Ortaklama fonksiyonları Python uygulama örneği**

Bu sonuçlardan yola çıkarak herhangi bir uygulamada ortaklama katmanında maksimum ortalama, minimum ortaklama veya ortalama ortaklama yönteminden birisinin kullanılması doğrudur demek yanlış olmaktadır. Görüntüde aranacak ve ilgilenilecek detaylara göre karar verilmelidir. Örneğin koyu renkli arkaplanda açık renk detaylar ile ilgileniliyorsa maksimum ortaklama açık pikselleri seçer. Bu katmanın amacı anlaşıldığı gibi özellik haritalarının boyutlarını azaltarak hesaplama miktarını ve süresini azaltmak için kullanılmaktadır. Havuzlama katmanı kullanılarak öğrenilecek parametre sayısını ve ağda gerçekleştirilen hesaplama miktarı azalır. Özetle bir evrişim katmanı tarafından elde edilen özellik haritasının bir bölgesinde bulunan özellikleri özetler.

## 4.7 Tam Bağlantı Katmanı

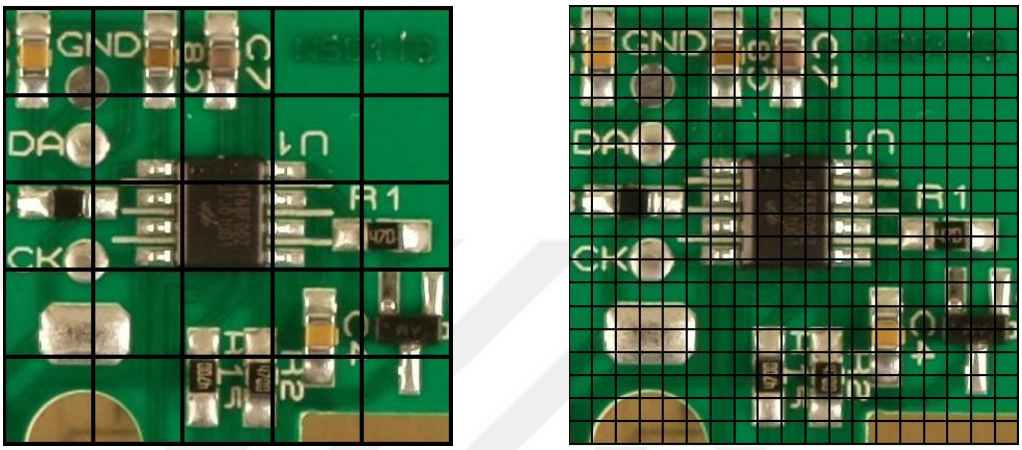
Bu katmana gelene kadar verilerden hesaplanan ağırlıklar birleştirilir ve nöronlara en uygun olacak ağırlık bulunur. Olasılıksal olarak sınıflandırma softmax fonksiyonu ile yapılır. Bu katmanda çıkış matrisi 1 x 1 olmaktadır. Gerçek sınıflandırmaya yapan son katmandır. Her etiket için olan olasılıkları verir. Tam bağlantı katmanı, tüm girdilerin tüm nöronlara bağlı olduğu bir sistem üzerinde işler.

## 4.8 YOLO-V4 Algoritması

Yolo algoritması ilk olarak 2015 yılında Joseph Redmon tarafından "You Only Look Once: Unified, Real-Time Object Detection" adlı makale ile tanıtıldı [19]. Bu algoritma günümüzde gerçek zamanlı nesne tanıma ve takibinde sık olarak kullanılan algoritmadır. Tercih edilmesinin temel olarak iki sebebi vardır. Bu sebeplerden biri, diğer evrişimsel sinir ağları kullanan algoritmalardan daha hızlı olmasıdır. Diğeri ise, yine bu algoritmalara göre daha başarılı sonuçlar vermesidir. Bahsedilen diğer algoritmalar sırasıyla gelişen Bölge tabanlı evrişimli sinir ağları (*İng.* R-CNN), hızlı bölge tabanlı evrişimli sinir ağları (*İng.* Fast R-CNN), daha hızlı bölge tabanlı evrişmli sinir ağları (*İng.* Faster R-CNN)'dir. Yolo algoritmasını bu algoritmalardan ayıran en büyük özelliğide görüntü üzerinde tek bir sefer işlem yapmasıdır. Örneğin bölge tabanlı nesne tespit algoritmaları nesne bulunduğu alanları tespit belirledikten sonra o bölgeler için ayrı evrişimli sinir ağı sınıflandırıcıları yürütmektedir. Bu durumda iki işlem sebebi ile işlem yükü arttığından saniye başına düşen kare sayımızın düşmesine ve sistemin yavaş çalışmasına sebep oluyor.

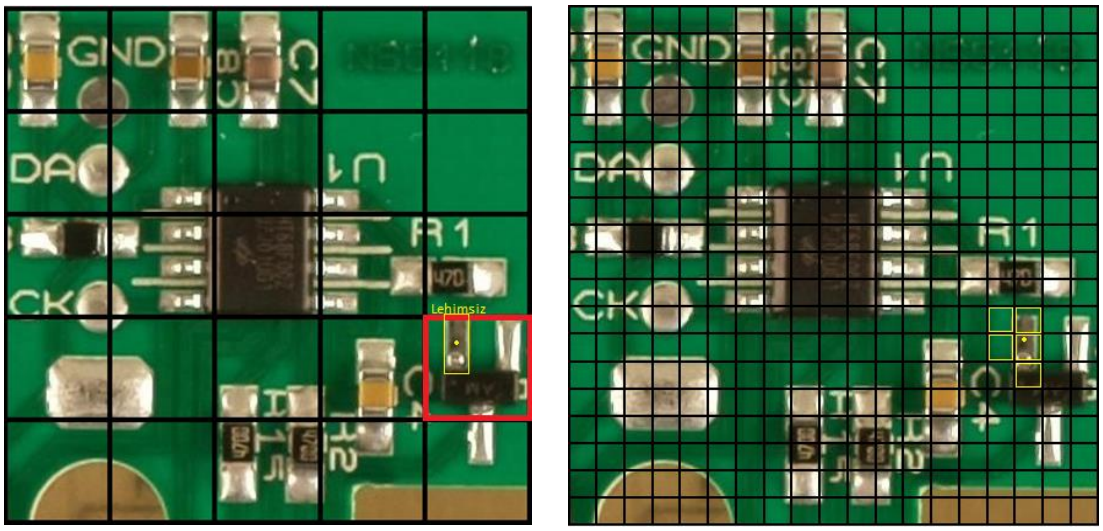
YOLO algoritması "Sadece Bir Kez Bak" ismiyle adından da anlaşıldığı gibi nesne tespiti yapabilmek için bir defa işlem uygulamaktadır. Algoritma görüntüde bulunmakta olan nesneleri ve koordinatlarını aynı anda hesaplamaktadır. Bahsettiğimiz algoritmalar girdi olarak verilen görüntülerde bölge önerileri fonksiyonunu kullanmaktadır. Yani, görüntüdeki nesnelerin olabilme ihtimalinin olduğu yerler bulunur ve bulunduktan sonra bu yerlere bölge önerileri (*İng.* region proposals) ismi verilir. Hangi bölgede nesne bulunup bulunmadığına bakar ve hangi sınıftan olduğuna karar verir. YOLO algoritmasında görüntü bir defa evrişimli sinir ağına sokulduktan sonra çıktı olarak tahmin

edilen nesnenin hangi sınıftan olduğu ve sınırlayıcı kutuların konumları elde edilir. İşlem sırasıyla, algoritma girdi olarak verilen giriş görüntüsünü N x N hücre ızgaralarına bölmektedir (**Şekil 4.8**). Girdi görüntüsünü eşit hücrelere böldükten sonra her hücreyi sinir ağına sokar ve algoritmanın ağdaki adımlarını uygular. Kullanılan eğitim için 19 x 19 ızgara seçilmiştir.



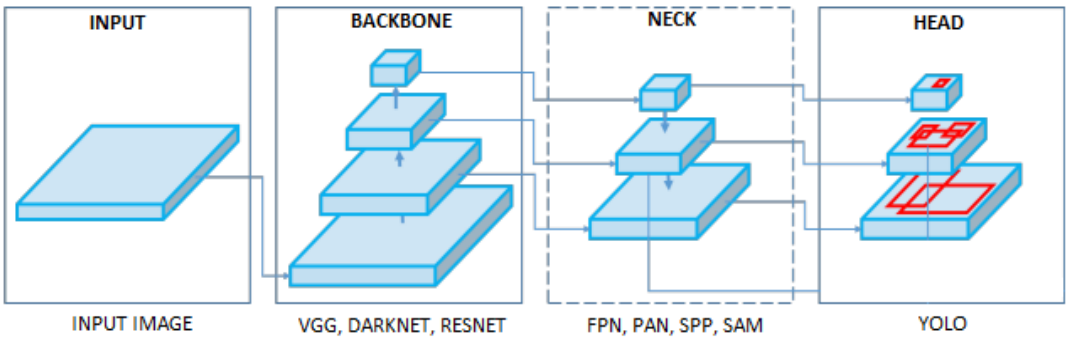

**Şekil 4.8: 5 x 5 ve 19 x 19 hücreleme**

Bölünen her hücrede eğitilen nesnelerin varlığı yokluğu araştırılır. Her bir ızgara hücresi kendi bölgesinden sorumludur. Tahmin edilen nesnenin merkezi hangi hücrede bulunursa o hücre o nesnenin öznitelik bilgilerinden (x, y, w, h, güven skoru) ve nesneyi tespit etmekten sorumludur. Her ızgara hücresi bulunan nesneleri içine alan sınırlayıcı kutuları çizer (**Şekil 4.9**). Bu sınırlayıcı kutuları çizdikten sonra her çizilen kutu için güven skoru ve her nesne için nesnenin bulunma olasılığı hesaplanır.




**Şekil 4.9: Sınırlayıcı kutu gösterimi (sarı) ve sınıflandırma**

YOLO algoritmasının dördüncü versiyonu 23 Nisan 2020 yılında yayınlanmıştır [32]. YOLOv4 algoritması omurga ağı, yukarı örnekleme ve yolo katmanlarından oluşmaktadır.




**Şekil 4.10: YOLOv4 Genel yapısı**

Bu versiyonu önceki versiyonlardan ayıran iki özelliği vardır. Tek bir ekran kartı ile hızlı training yapılabilmesidir. Diğer bir özelliği ise bazı özel tekniklerin eklendiği hız, başarı iyileştirme ve veri arttırma teknikleridir. Omurga ağı ve kafa ağı arasında bulunan boyun (*İng.* Neck) katmanının içerdiği hediye çanta (*İng.* Bag Of Freebies) ve özel çanta (*İng.* Bag Of Specials) teknikleridir. Bu teknikler bilgi çıkarımları için, yüksek doğruluk oranları elde etmek için kullanılmıştır. Bu katmanda YOLOv3 versiyonuna göre CutMix

ve Mosaic fonksiyonları ile veri arttırma, rastgele görüntü kırpma, gölgeleme, görüntü bulanıklaştırma fonksiyonları gibi özellikler içermektedir. Tasarlanan hedef tespit modelinin farklı ortamlarda elde edilen görüntülere karşı daha sağlam olması için giriş görüntüsünün değişkenliği arttırılabilir hale gelmiştir. Konfigürasyon dosyasında görüntüde bozulmalar için aşağıdaki parametreler ile sisteme daha uygun hale getirilebilmektedir.

- Aydınlatma bozulmaları: parlaklık, zıtlık, hue, doyma, gürültü ekleme
- Geometrik bozulmaları: rasgele yakınlaştırma, kesme, döndürme

YOLOv4 algoritması omurga (İng. backbone) aşamasında özellik çıkarıcı olarak Oxford Üniversitesi'nden "Very Deep Convolutional Networks for Large-Scale Image Recognition" adlı dergide yayınlanan makalede ve ILSVRC yarışmasının ikincisi olan Simonyan ile arkadaşı Zisserman tarafından geliştirilen VGGNet [33] adında bir CNN mimarisi ağına benzeyen evrişimsel sinir ağlarının öğrenme kapasitesini iyileştirmek için önerilen 28 Kasım 2019'da yayınlanmış CSPDarknet-53 [34] omurgasını kullanır.

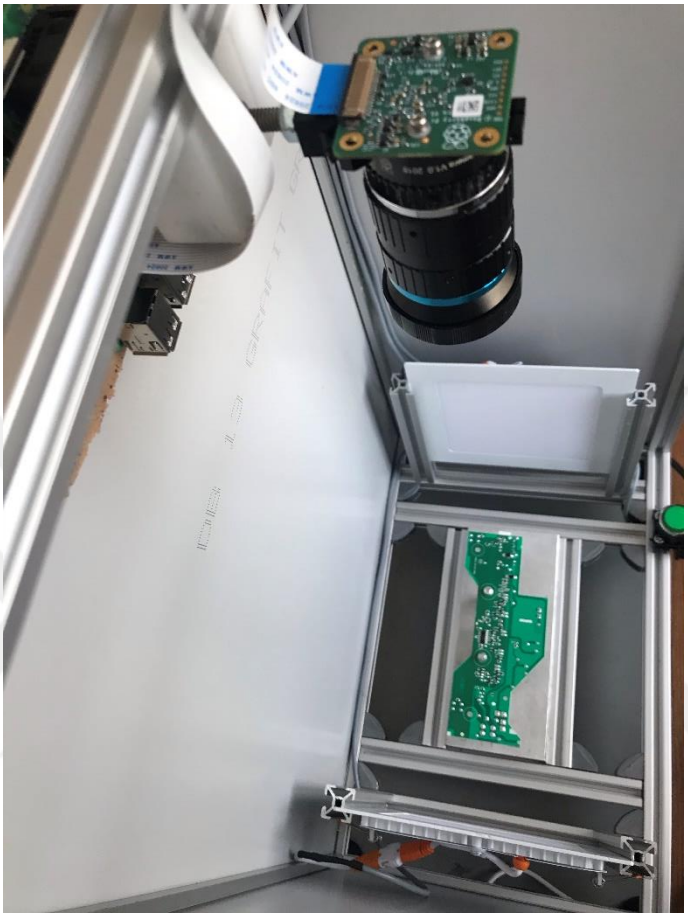
| | Type | Filters | Size | Output |
|---|---|---|---|---|
| | Convolutional | 32 | 3 × 3 | 256 × 256 |
| | Convolutional | 64 | 3 × 3 / 2 | 128 × 128 |
| 1× | Convolutional | 32 | 1 × 1 | |
| | Convolutional | 64 | 3 × 3 | |
| | Residual | | | 128 × 128 |
| | Convolutional | 128 | 3 × 3 / 2 | 64 × 64 |
| 2× | Convolutional | 64 | 1 × 1 | |
| | Convolutional | 128 | 3 × 3 | |
| | Residual | | | 64 × 64 |
| | Convolutional | 256 | 3 × 3 / 2 | 32 × 32 |
| 8× | Convolutional | 128 | 1 × 1 | |
| | Convolutional | 256 | 3 × 3 | |
| | Residual | | | 32 × 32 |
| | Convolutional | 512 | 3 × 3 / 2 | 16 × 16 |
| 8× | Convolutional | 256 | 1 × 1 | |
| | Convolutional | 512 | 3 × 3 | |
| | Residual | | | 16 × 16 |
| | Convolutional | 1024 | 3 × 3 / 2 | 8 × 8 |
| 4× | Convolutional | 512 | 1 × 1 | |
| | Convolutional | 1024 | 3 × 3 | |
| | Residual | | | 8 × 8 |
| | Avgpool | | Global | |
| | Connected | | 1000 | |
| | Softmax | | | |

**Şekil 4.11: Darknet-53 yapısı [34]**

Bu omurga ağı, girdi katmanın uygulanan giriş görüntüsünden özellik haritasını elde eder ve temel bileşenler olarak artan blokları (*İng.* residual blocks) kullanır. Her artan blok 3x3 ve 1x1 evrişimli katman çiftinden oluşur. Toplam evrişimli katman sayısı 53 olduğundan Daknet-53 adındaki 53 konvolüsyon katmanlarının sayısını belirtir. Son özellik haritasındaki uzamsal çözünürlük giriş görüntüsünden 32 kat daha küçüktür. YOLO algoritması Darknet omurgası kullanılarak eğitim yapılabilmesi için bazı parametrelere ve dosyalara ihtiyaç duyar. Konfigürasyon (.cfg) dosyası yolo algoritmasının yapısını içeren en önemli ve algoritmanın başladığı ilk dosya diyebiliriz. Bu dosya ağın hızını, başarısını ve işlem yükünü etkileyecek bilgilere sahiptir. Eğitilip oluşturulacak her model için ve eğitimde kullanılacak bilgisayarın özellikleri ile ekran kartı kapasitesi için parametrelerin ayarlanması gerekmektedir. Algoritmanın başlatılabilmesi için darknet kurulumu yapılan bilgisayarda kurulumun gerçekleştirildiği dosyada isimler ve veri uzantılı dosyalara bakar. Names uzantılı dosyada etiketlenen sınıf isimleri bulunmaktadır. Veri uzantılı dosyada ise yapılacak eğitimde etiketlenen sınıf sayısı, eğitim ve testte kullanılacak görüntülerin Yolo formatındaki txt dosyalarının yolları gösterilir.

# 5. GÖRÜNTÜ ALIM VE SINIFLANDIRMA

## 5.1 Veri Seti Toplama

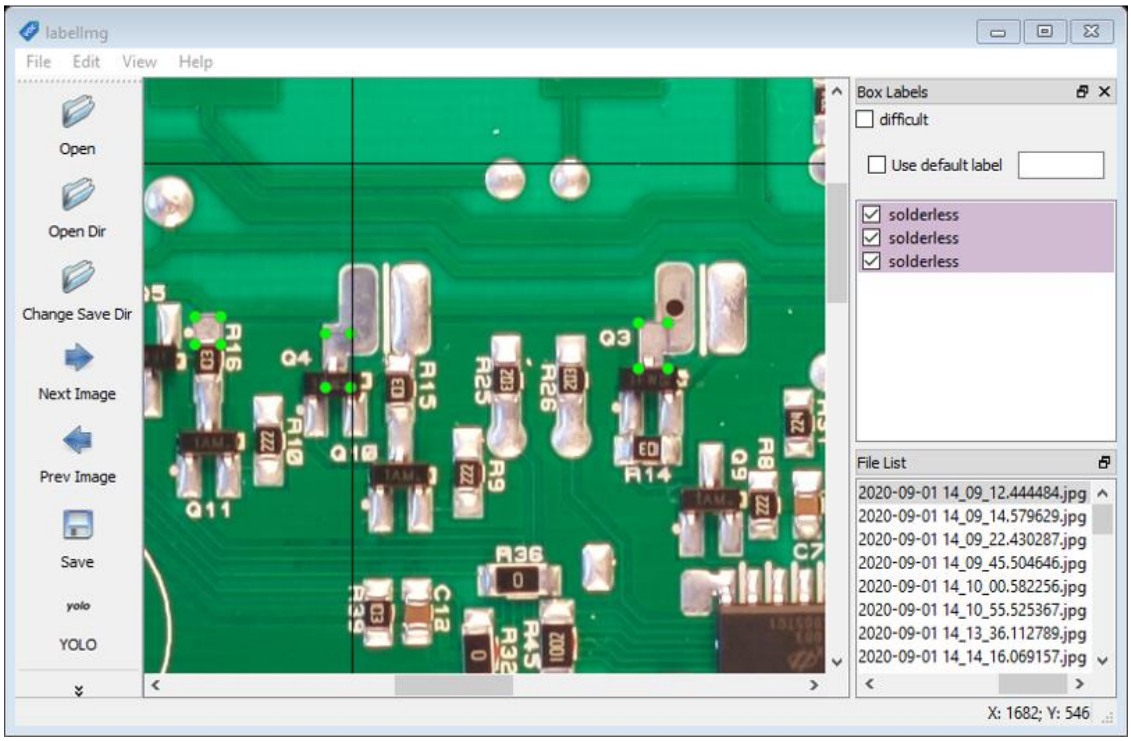


**Şekil 5.1: Eğitim için seçilen BDK'nın cihaz içine yerleşimi ve kamera donanımı**

Yapılan bu sistemde çalıştığım firmanın piyasada bulunan altı tip çeşitli elektrikli süpürge kartının ve kahve makinası güç kartları yaklaşık 800 adet yüzey montajlı dizgi işlemi tamamlanmış baskılı devre kartı görüntüsü kullanılmıştır. Kamera – nesne mesafesi, yönelim ve aydınlatma gibi edinim parametresi varyasyonlarını ortadan kaldırmak için alüminyum profil tabanlı bir kafes geliştirilmiştir (**Şekil 5.1**). Elde edilen baskılı devre kartı görüntüleri YOLOv4 derin öğrenme algoritmasının hem eğitim hem de test aşamalarında kullanılmaktadır. Kamera – nesne mesafesi görüntü tüm kartı kapsayacak şekilde odak ayarlanarak 40 cm'ye sabitlenmiştir. Aydınlatma için kafesin her iki tarafına dikdörtgen beyaz led panelleri yerleştirilmiştir.

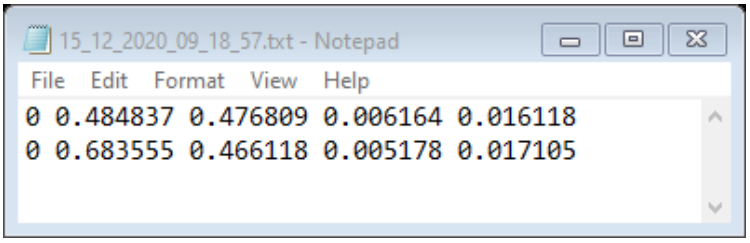
## 5.2 Veri Seti Etiketleme

Veri seti etiketleme işlemleri LABELIMG programı kullanılarak gerçekleştirilmiştir. LABELIMG Python programlama dili kullanılarak ve arayüzü Qt Designer kullanılarak oluşturulmuş bir programdır. LABELIMG programı ImageNet tarafından kullanılan format olan PASCAL VOC formatında XML dosyaları olarak kaydedebilir. Bunların yanısıra etiketlerini txt uzantılı kaydederek YOLO formatınıda desteklemektedir. **Şekil 5.2**' de görüldüğü üzere YOLO formatı kullanılmıştır.



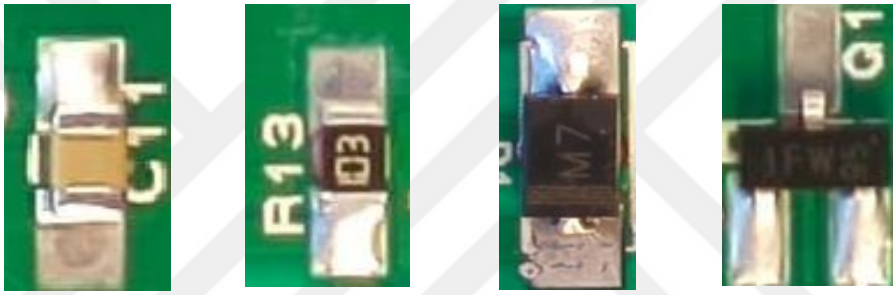

**Şekil 5.2: Labelimg örnek veri seti etiketleme görüntüsü**

YOLO formatı kullanıldığı zaman program çıktı olarak her görüntüye ait bir etiket.txt dosyası üretmektedir ve verisetinde kullanılan tüm sınıfların isimlerinin bulunduğu bir classes.txt dosyası üretir. Görüntülere ait txt dosyalarının içinde sırasıyla etiket_kimlik_no, x_merkez_normali, y_merkez_normali, genişlik_normali ve yükseklik normali değerleri yazmaktadır (**Şekil 5.3**).




**Şekil 5.3: Etiketleme bilgilerini içeren txt dosyası içeriği**

Veri seti oluşturulurken etiketlenen bazı komponentlerin görüntüleri aşağıdaki şekilde gösterilmektedir. Yaklaşık olarak pedlerin çözünürlükleri en küçük 30 x 30 en büyük 50 x 80 piksel olarak maksimum çözünürlükte etiketlenmiştir.




**Şekil 5.4: Etiketlenen komponentlerin lehimsiz ped örnekleri**

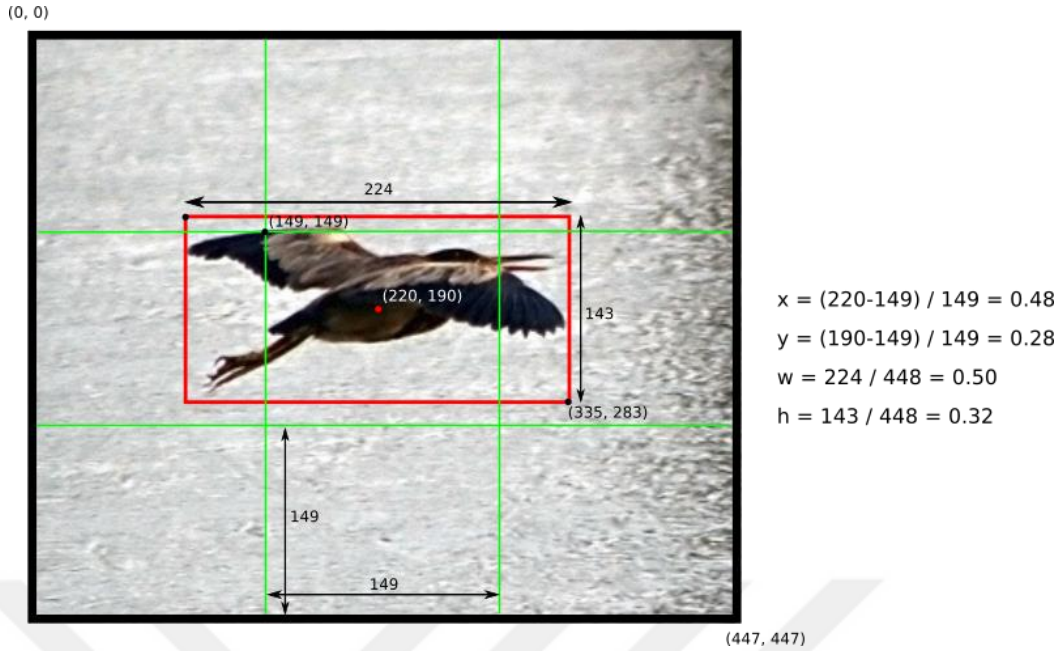
Etiket kimlik numarası classes.txt dosyasındaki tanımlanan sınıfların dizin (*İng.* index) numarasıdır. İlk etiket numarası sıfır ile başlar ve artan tam bir sayı olarak devam eder. Etiket dosyasındaki konum öznitelikleri [0,1] arasında olacak şekilde normalleştirilmiştir (**Şekil 5.5**).

X_MERKEZ_NORMALİ = X_MERKEZ_MUTLAK/GENİŞLİK

Y_MERKEZ_NORMALİ = Y_MERKEZ_MUTLAK/YÜKSEKLİK

GENİŞLİK_NORMALİ = ETİKET_MUTLAK_GENİŞLİĞİ/GÖRÜNTÜ_GENİŞLİĞİ

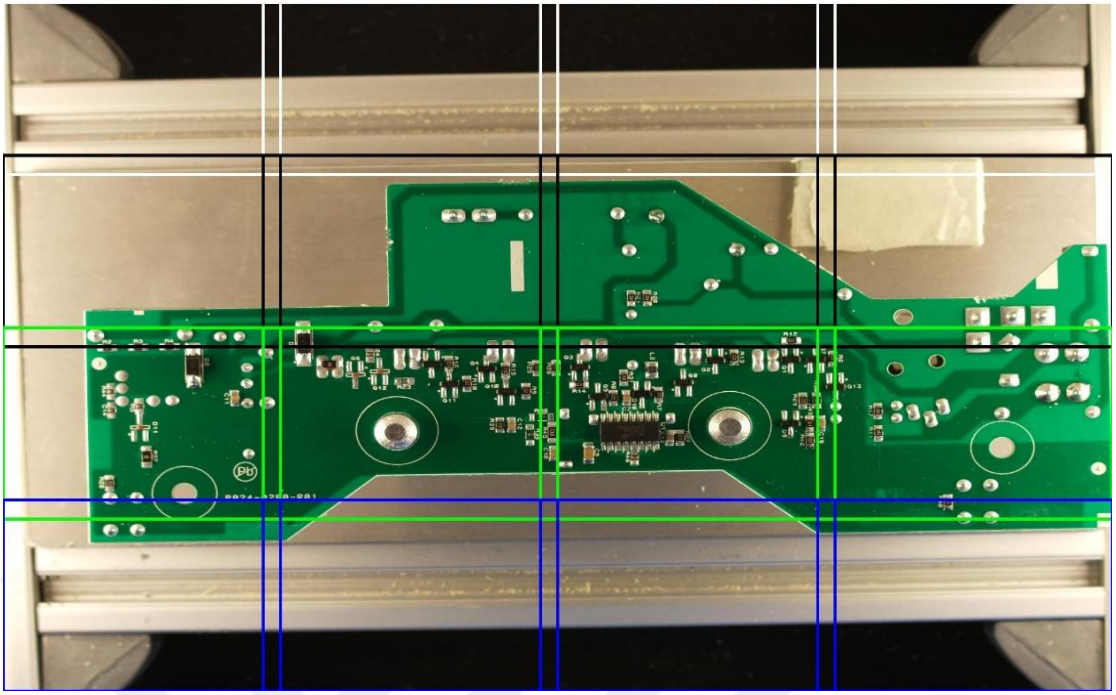
YÜKSEKLİK_NORMALİ = ETİKET_MUTLAK_YÜKSEKLİĞİ/GÖRÜNTÜ_YÜKSELİĞİ




**Şekil 5.5: Koordinat normalleştirme**

## 5.3 Veri Seti Çoğaltma ve Bölümleme

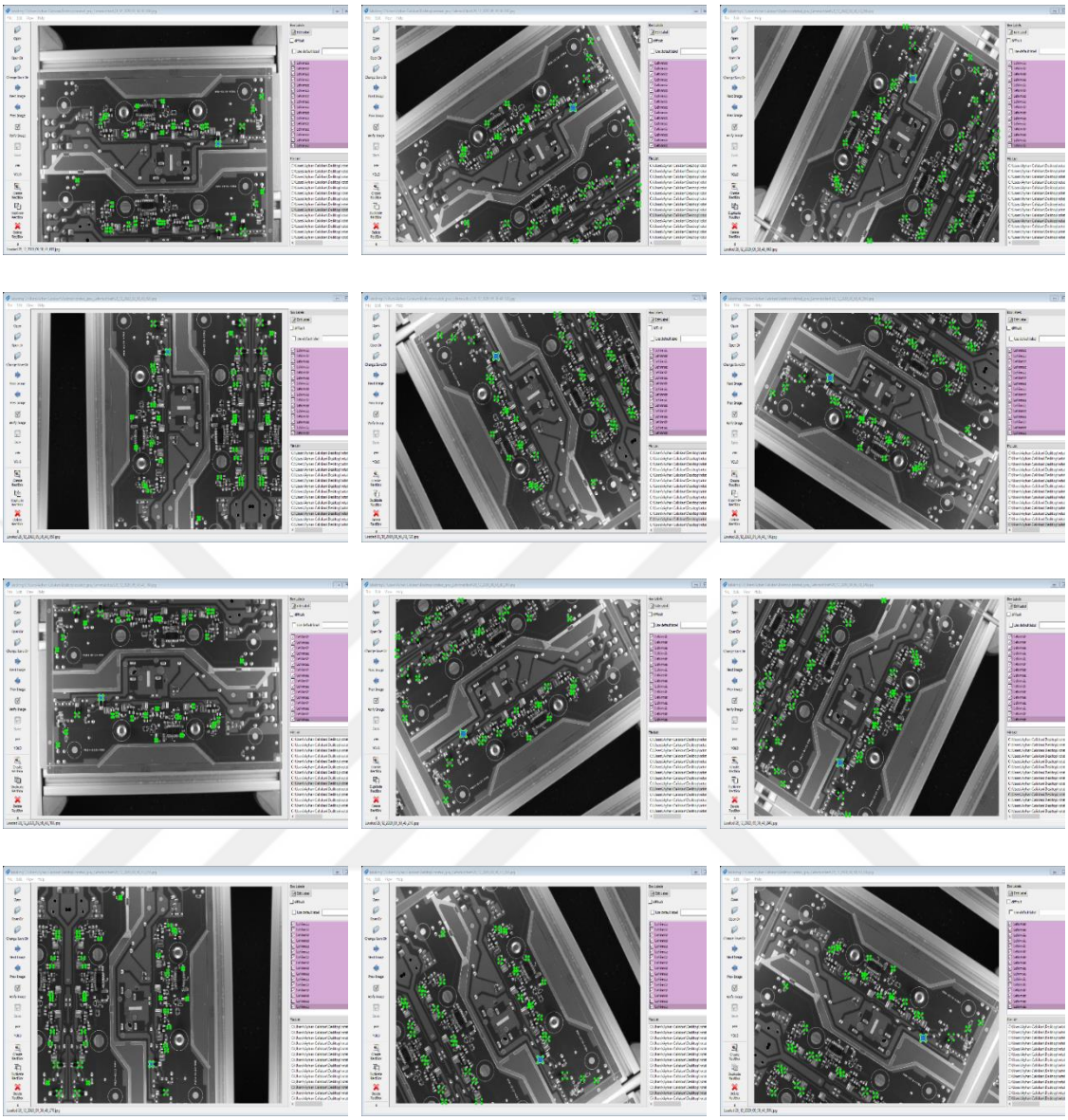
Yapılan çalışmanın kilit noktası olan veri setini çoğaltma ve bölümleme işlemleri için Python programlama dili ile yazılan bir senaryo (*İng.* Script) programı hazırlanmıştır. Hazırlanan bu senaryo programı, 4056x3040 çözünürlüğüne sahip görüntüyü 16 parçaya böler. Görüntü bölme işlemi yapıldıktan sonra her parça 1014x760 çözünürlüğe denk gelmektedir ve bu parçalar ayrı görüntüler olarak kaydedilmektedir. Bu işlem sonrasında her 5K'lık görüntüden 16 parça görüntü elde edilmektedir Burada kritik nokta bizim nesne takibi yapacağımız sınıfların piksel boyutları maksimum 30x30 ölçülerinde olduğundan görüntü bölümleme işlemi sırasında 30 piksel hem yatay hem dikey eksende taşma gerçekleştirilmiştir. **Şekil 5.6**'da görüldüğü gibi taşma (*İng.* Overlap) işlemi sonucunda görüntü bölme işlemi gerçekleştirilirken lehimsiz olarak tanımlanan sınıfın etiketleri görüntünün bölündüğü noktaya denk gelmesi halinde oluşacak kayıplar bu şekilde ortadan kaldırılmıştır.



**Şekil 5.6: Görüntü bölümleme ve taşma işlemi**

Bölümleme işlemi sonucunda elde edilen görüntü adedi 800*12 ‘den 9600 adettir. Elde edilen bu görüntülerin içinde etiketleme işlemi bulunan görüntü sayısı neredeyse yarısı kadardır. Eğitim esnasında gereksiz işlem yükü ve hata oluşturacak boş görüntüler Python programlama dili ile yazılmış bir yazılım ile silinmiştir. Görüntüler etiket.txt dosyasında herhangi bir etiketleme işleminin bulunmamasına bakılarak silindikten sonra kalan görüntü sayısı 3650 adettir.

Bir senaryo yazılımı kullanılarak bu görüntülere ait öznitelik bilgilerini kaybetmeden txt dosyalarındaki koordinatlar hesaplanarak her görüntüye 30 derece eklenecek şekilde veri arttırma işlemi uygulanmıştır (**Şekil 5.7**). Bu işlem sonunda gerçek bir üretim hattında kullanılan baskılı devre kartlarında yüzey montajlı kondansatör, diyot, direnç ve transistör gibi elektronik komponentlerin açık devre lehim padleri ile ilgili ciddi bir veri seti elde edilmiştir. Her görüntü başına 11 adet görüntü eklenmiştir. Eğitime hazır olan görüntü sayısı 40150 ve eğitime hazır olan etiketlenmiş lehimsiz pad sayısı 76000’dir.

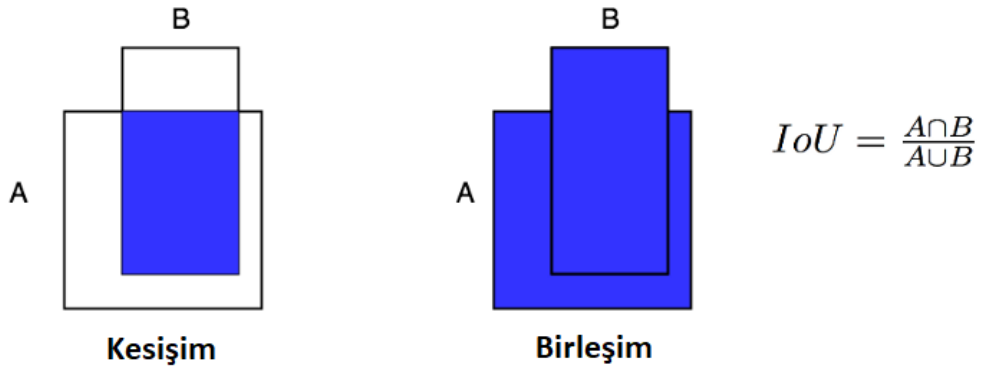


**Şekil 5.7: Görüntünün 0 dereceden 360 dereceye 30'ar derecelik adımlarla döndürülmesi ile yapılan veri arttırma işlemi**

## 5.4 Kesişme Bölgeleri (Intersection over Union)

Bir sınıf veya nesne tespit edici sisteme kesişme bölgeleri uygulayabilmek için bazı ihtiyaçlar duyulmaktadır. Bu ihtiyaçlar nesnemizin görüntüde nerede olduğuna dair gerçek sınırlayıcı kutulardır (*İng.* Grounded-truth bounding box). Bir nesne birçok kutuda bulunabilmektedir ve bu durumda nesnenin örtüşmesi demektir. Nesne algılama

algoritmasının doğruluğunu ölçebilmek için kullanılan bu değerlendirme iki sınırlayıcı kutu arasında oluşan örtüşmenin ölçüsünün sonucudur.

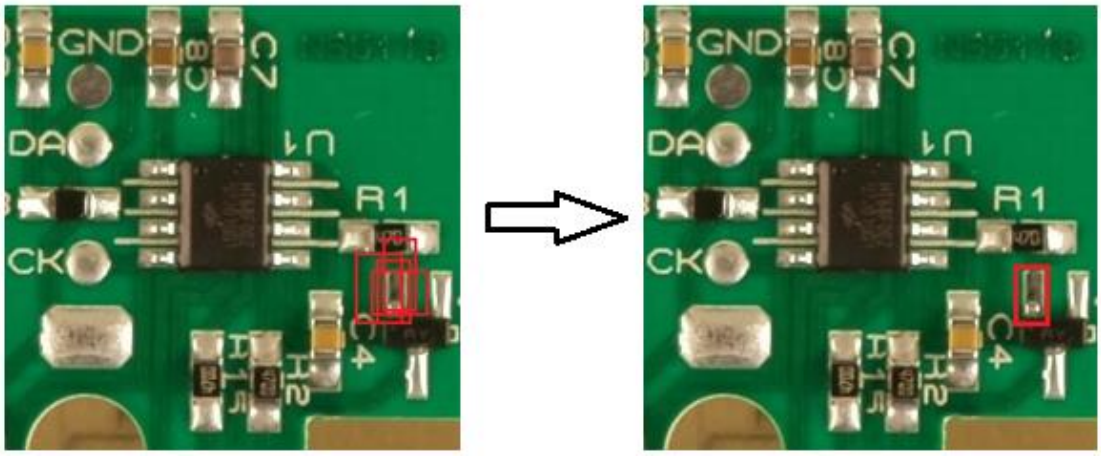



**Şekil 5.8: Kesişme-Birleşim oran hesabının gösterimi**

Nesne algılama modelleri, farklı nesneleri tahmin etmek ve sınırlayıcı kutu tahminleri yapabilmek için bağlantı kutularını kullanmaktadır. Bağlantı kutuları, sabit yükseklik ve genişlikte konfigürasyon dosyasında bulunan önceden tanımlanmış kutuladır. Tespit edilen herhangi bir nesnenin önceden tanımlanmış kutulardan en az birinin içine rahatça sığma için sınır sayıda bağlantı kutusu kullanmaktadır. Yükseklik ve genişlik değerlerinin sonsuz olamaması için önceden tanımlanmış değerlerle sınırlanmasıdır. Bağlantı kutuları ölçüleri her eğitime göre ve takip edilecek nesneye göre farklı seçilmektedir. Sonuç olarak, farklı nesnelerin farklı yükseklik ve genişlik oranlarına düşmesidir.

## 5.5 Maksimum Olmayan Bastırma (Non-max Suppression)

Bir nesne birden çok kez tespit edilebilir. Son sonuç için temizleme işlemi gerekmektedir. En yüksek olasılığa sahip sınırlayıcı kutu alınabilmesi için bu teknik kullanılmaktadır. Güven skoru düşük olan tahminleri hiç değerlendirmeye almadan sınırlayıcı kutuları eleme yöntemidir. Tahmin esnasında maksimum olmayan bastırma tekniği kullanılarak tahmin edilen nesne için birden fazla olan sınırlayıcı kutuların güven skoru en yüksek olan bırakılabilir. Birden fazla algılama sorununu ortadan kaldırmak için kullanılır.




**Şekil 5.9: Maksimum olmayan bastırma etkisi**

Maksimum olmayan bastırma yöntemi birleşim bölü kesişim tanımını kullanır. IoU tanımlandıktan sonra maksimum olmayan bastırma işlenecek kutu kalmayana kadar çalışır. En yüksek tahmin olasılığına sahip kutu seçilir ve seçildikten sonra en yüksek tahmin olasılığına sahip kutu ile yüksek IoU değerine sahip tüm kutuları kaldırır. Seçilen kutu alınır.

## 5.6 Model Eğitimi için Kullanılan Donanım

Çalışmada baskılı devre kartlarında komponentlerin lehimsiz pedlerinin tespiti için eğitilecek modelin eğitim ve test işlemleri NVIDIA GeForce RTX3070 8GB ekran kartı kullanılarak çalışılmıştır. Model eğitimi 6000 adım (epoch) olarak ayarlanmıştır. YOLOv4 modelindeki öğrenme oranı ve hiperparametleri aynı şekilde kullanılmıştır. Modelin eğitimi 95 saatte tamamlanmıştır.

## 5.7 Model Eğitimi Yöntemleri

Karışıklık matrisi, genellikle bir sınıflandırma modelinin gerçek değerlerinin bilindiği bir dizi test verisi üzerindeki performansını tanımlamak için kullanılır. Bu ölçümler dört parametre kullanılarak hesaplanır. Bu parametreler doğru pozitif, doğru negatif, yanlış pozitif, yanlış negatif'tir. Gerçek değerlerin öngörülen sınıfla çeliştiği noktada yanlış pozitifler ve yanlış negatifler ortaya çıkmaktadır.

**DP (Doğru pozitif)** = Doğru tahmin edilen pozitif değerlerdir. Gerçek sınıfın değerinin evet olduğu ve öngörülen sınıfın değerinin de evet olduğu anlamına gelmektedir.

**DN (Doğru negatif)** = Doğru tahmin edilen negatif değerlerdir. Gerçek sınıfın değeri hayır olduğu ve öngörülen sınıfın değerininde hayır olduğu anlamındadır.

**YN (Yanlış negatif)** = Gerçek sınıf evet olduğu durumda öngörülen sınıf değeri hayır ise yanlış negatif anlamına gelir.

**YP (Yanlış pozitif)** = Gerçek sınıf değeri hayır olduğunda ve öngörülen sınıf evet olduğunda yanlış pozitiftir.

Hassasiyet (*İng.* Precision), tahminlerinizin ne kadar doğru olduğunu ölçer. Doğru tahmin edilen pozitif gözlemlerin toplam tahmin edilen pozitif gözlemlere oranıdır.

$$Hassasiyet = \frac{DP}{DP + YP}$$

Geri çağırma (*İng.* Recall), tüm pozitifleri nekadar iyi bulduğunu ölçer. Yani, doğru tahmin edilen olumlu gözlemlerin olması gereken tüm doğru pozitiflere oranına bakılır.

$$Geri\ çağırma = \frac{DP}{DP + YN}$$

F1 skoru (*İng.* F1-Score), hassasiyet ve geri çağırma arasındaki dengeyi ölçer. F1 değeri yüksek olduğunda, bu hem hassasiyet hem de geri çağırmanın yüksek olduğunu gösterir. Daha düşük F1 skoru hassasiyet ve geri çağırma arasında daha büyük bir dengesizlik anlamına gelmektedir. Hassasiyet ve geriçağırmanın ağırlıklı ortalamasıdır. Bu puan hem yanlış pozitifleri hem de yanlış negatifleri hesaba katmaktadır.

$$F1\ değeri = 2 * \frac{Hassasiyet * Geriçağırma}{Hassasiyet + Geriçağırma}$$

Ortalama hassasiyet (*İng. Average Precision*) nesne tespiti doğruluğunu ölçmek için kullanılan bir metriktir. Farklı geri çağırma değerlerini hesaba katan ortalama hassasiyettir. Ortalama hassasiyet gerçek sınırlayıcı kutuyu tespit edilen kutuyla

karşılaştırır ve sonuç olarak bir puan verir. Bu verilen puan ne kadar yüksek ise eğitilen modelde sistemin test sonuçları ve tahminleri o kadar doğru olmaktadır. Kayıp fonksiyonu ve ortalama hassasiyet değerleri eğitim sonunda grafiksel olarak incelenmiş ve gösterilmiştir. Güven skoru, geçerli hücerede sınırlayıcı kutuların içerdiği nesneden ne kadar emin olduğunu bildirir. Çıkış vektöründe pc ile tanımlanır. 0 kesinlikle ise yok 1 ise kesinlikle var anlamına gelir. Eğitilen sistemde kaç adet sınıf var ise o kadar tahmin değeri olur. 2016 yılında yayınlanan makalede YOLO kutu güven skorunu P(nesne)*(IoU) olarak tanımlar.

Performans ölçümü için sınıflandırma modelinin performansının değerlendirilmesinde Hassasiyet-Geri Çağırma ve ROC eğrisinden yararlanılmıştır. Aynı zamanda AUC olarakta bilinen ROC eğrisi, farklı eşik değerlerinde sınıflandırma modelinin performansını ölçmek için kullanılan bir performans metriğidir.

# 6. TESTLER

Sunulan bu tez çalışmasında baskılı devre kartları için düşük maliyetli lehim hatası tespiti yapabilen iki sistem tasarlanmıştır. Tasarlanan sistemlerden birincisi, ekler bölümünde belirtildiği gibi hareketli bir kamera düzeneği tasarlanarak baskılı devre kartı üzerinde gezdirilerek görüntü alınmasıdır. Görüntüsü çekilecek ve hata tespiti yapılacak olan baskılı devre kartının tüm görüntüsünü topladığı görüntüleri birleştirerek hata tespiti yapmayı hedeflemiştir. Tasarlanan bu sistemde Python ve Qt kullanılarak operatör için kullanıcı arayüzü geliştirilmiştir. Geliştirilen sistem ile Atmega328 işlemcisi programlanarak seri haberleşme protokolü aracılığıyla kamera sistemi yatay ve dikey eksende hareket ettirilmiştir. Sistemde OPENCV kütüphanesi kullanılarak görüntü işleme yöntemleri ve teknikleri kullanılmıştır. Elde edilen sistemin sonucunda hareket düzeneği ile tek bir baskılı devre kartının tamamını parçalar halinde görüntü alarak toplamak yaklaşık olarak 30 saniyeden fazla sürdüğü hesaplanmıştır. Bu işlem süresi sonunda sistemin görüntü toplamasının yavaş olması sebebi ile gerçek hayatta baskılı devre kartı üreten firmada kullanıldığında üretimi yavaşlattığı tespit edilmiştir. İkinci dezavantaj olarak ise temin edilen sistemin maliyeti oldukça yüksek çıkmıştır. Sistemde alüminyum profiller haricinde raylı araba sistemleri, adımlı motorlar, rulmanlar, motor sürücüler ve Atmega328 işlemcisi, güç kaynağı ve sistemi kapatabilmek için plakalar kullanılmıştır. Ana sebep olarak sistemin yavaş olması yüzünden ikinci bir sistem tasarlanması planlanmıştır.

İkinci tasarlanan sistemde öncelikli olarak hızlı sonuçlar ve düşük maliyet hedeflenmiştir. İkinci tasarımda sistemin tüm kartın görüntüsünün yanı sıra üretim için panelleme işlemi gerçekleştirilmiş 5 adet kartın tek bir seferde görüntü alma işlemi alınıp doğru ve hızlı işlemler gerçekleştirilebilmesi için çalışılmıştır. Bu sistemin önceliklerinden olan hızlı sonuç alma adımları için görüntü üzerinde gezdirilip fotoğraf çekme işlemi yerine tek bir

seferde göürüntü alma planı yapılmıştır. Yapılan bu planlamada yüksek çözünürlüğe sahip çekilen fotoğrafın üzerinde görüntü bölme ve parçalama işlemleri yaparak en büyük problem olan komponentlerin lehimsiz olan bacaklarının tespit edilmesi hedeflenmiştir. İkinci cihaz tasarlandıktan sonra hata tespitlerini yapabilmesi için makine öğrenmesi kullanılmasına karar verilmiştir. Bu çalışmada evrişimli sinir ağları altyapısına sahip olan YOLOv4 algoritması ile kendi verisetimiz oluşturularak iki tip eğitim yapılmıştır. Bu eğitimler arasındaki doğruluk ve ortalama hassasiyet değerleri karşılaştırılmıştır. Eğitim sonuçlarının hassasiyet ve kayıp fonksiyonu değerleri grafiksel olarak **Şekil 6.1** ve **Şekil 6.2'**de gösterilmiştir. Birinci eğitimde görüntüler eğitime verilmeden önce veri seti oluşturulurken gri seviyeye çevrilimiştir. Görüntüler gri seviye görüntüye dönüştürüldükten sonra veri seti sınıfı etiketleme işlemi elde edilen görüntüler ile yapılmıştır. Görüntülerde farklı açılarla görüntü döndürme gibi veri arttırma işlemleri kullanılmamıştır. Görüntüyü bölme ve parçalama işlemleri yapılmamıştır. Yani, 4056x3040 çözünürlüğündeki görüntü YOLO algoritmasına direk olarak verilmiştir. Algoritma bu çözünürlüğü konfigürasyon dosyasında ayarlanan yükseklik ve genişlik olarak 608x608 çözünürlüğüne çevirmektedir. Bu çözünürlüğe çevirmek 4k görüntüde 80x100 piksellik detayların yani bize lehimsiz sınıfı verilerinin kaybolması anlamına gelmektedir. Bu verilerin kaybolduğu eğitim sonuçlarında elde edilen mAP sonuçlarında gözlemlenmiştir (**Şekil 6.1**).

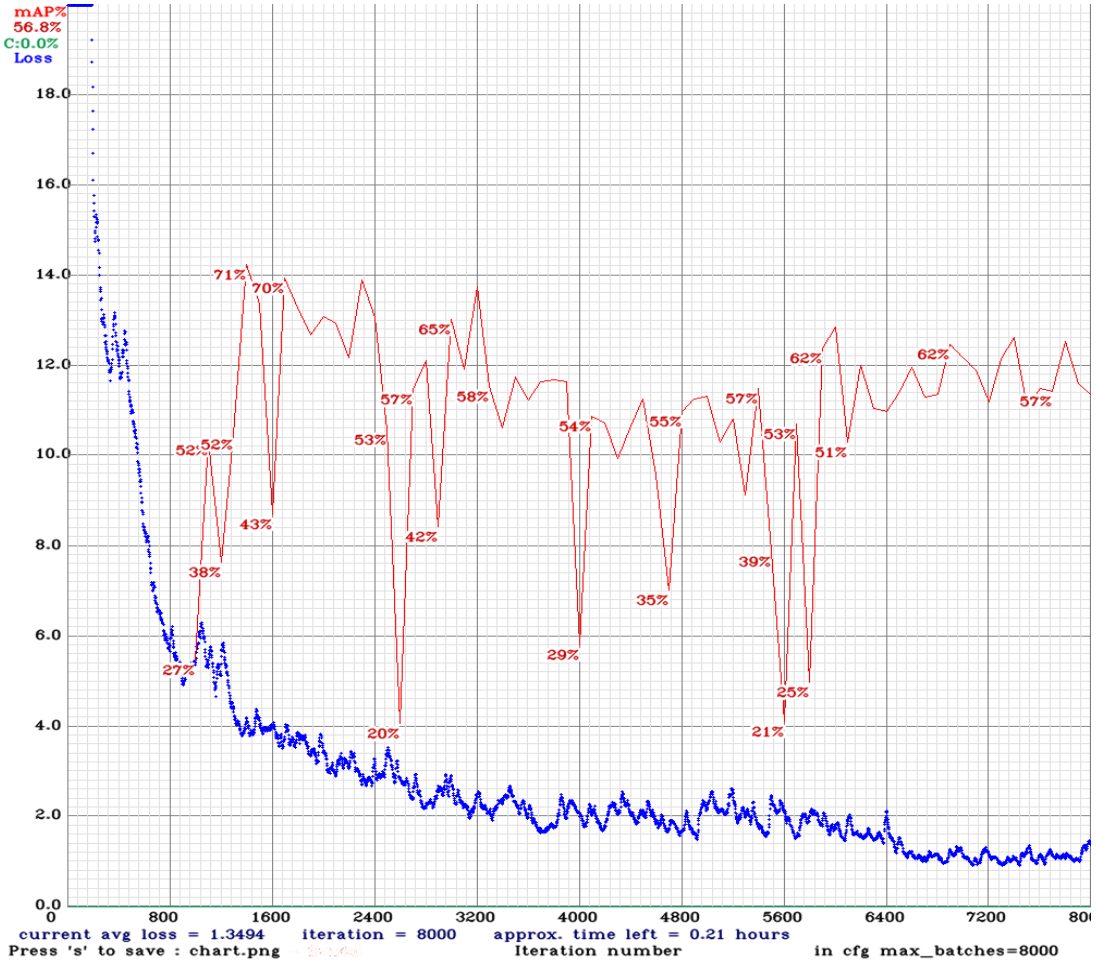
YOLO algoritması ekran kartının tek bir seferde işleyebileceği görüntü sayısının ayarlanmasına izin vermektedir fakat tek bir seferde işlem yapılacak görüntü sayısı ne kadar yüksek ise çözünürlük o kadar düşük olmalıdır. Bu çözünürlüğe çevirmesinin sebebi işlem yükü fazlalığı sebebiyle kullanılan ekran kartının rastgele erişimli hafıza (*İng.* Random Access Memory) belleğinin yetersiz kalmasıdır. Eğitim 8000 dönüm yapılmıştır. Eğitim süresi yaklaşık olarak 44 saat sürmüştür.

Yapılan ilk eğitim sonucunda elde edilen modelin tahmin ve test sonuçlarından elde edilen önemli bilgiler aşağıdaki gibidir;

1. Eğitime başlamadan önce girdi görüntüsü olarak uygulanan görüntünün çözünürlüğü ne ise tahmin edilecek görüntünün yani testte kullanılacak

görüntünün çözünürlüğü aynı olması gerektiği sonucuna varılmıştır. Çözünürlüğün fazla olması durumunda kullanılan ekran kartının (NVIDIA Geforce RTX3070) belleğinin yetersizliği sebebi ile YOLO algoritmasının eğitim için işlem yapabildiği en büyük çözünürlüğün 608x608 olduğu tespit edilmiştir. Çözünürlük küçültme işlemi gerçekleştirildiğinde görüntüde önemli bilgi kaybı olduğundan doğruluk yüzdesi (%40) çok düşük olduğu gözlemlenmiştir.

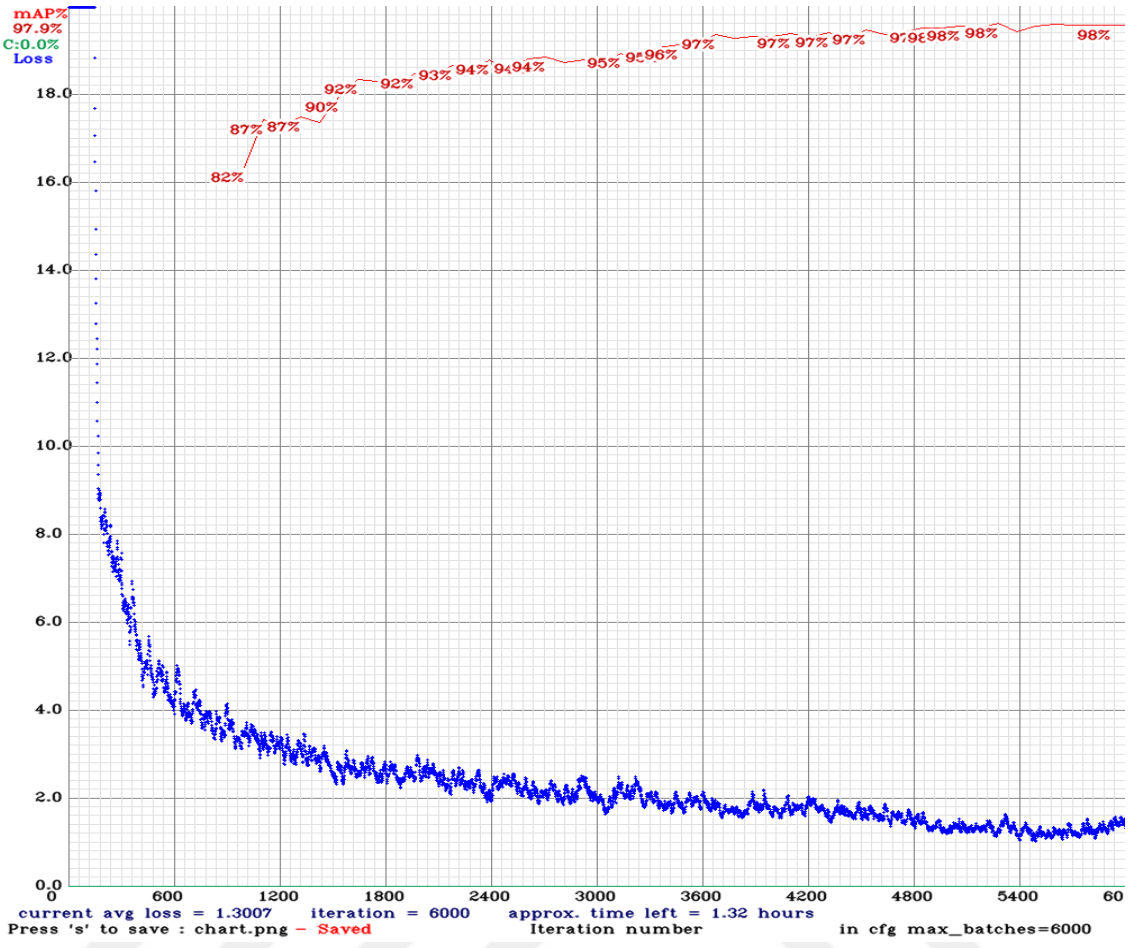
2. Birinci maddeye bağlı olarak yaptığımız lehimsiz sınıfı için kestirilmek istenen nesnelerimizin 4056 x 3040 görüntüde 80 x 100 piksel olması ve kaybedilen veri sebebiyle mAP değerlerinin çok düşük olduğu tespit edilmiştir.
3. Eğitilen modelde gri seviye görüntülerin kullanılmasından dolayı, lehimsiz sınıfına ait olan nesneler görüntüde olmamasına rağmen baskılı devre kart kenarlarının düz yüzeyle kesiştiği noktaların da ışık yansımaları sebebiyle, lehimsiz olarak kestirildiği tespit edilmiştir.
4. Lehimsiz sınıfına ait nesnelerin tespit edildiği bir devre kartının, 180 derece veya 90 derece çevrildiğinde, aynı nesnelerin artık tahmin edilemez hale geldiği tespit edilmiştir.
5. 8000 dönüm (*İng.* iteration) yapılan işlem adımlarının 6400 dönümden sonra kararlı hale geldiği gözlemlenmiştir. Ezberleme ve fazla uyum gösterme (*İng. overfitting*) gerçekleşmemesi için en yüksek dönüm sayısı 6000 olarak belirlenmiştir.




**Şekil 6.1: Gri seviye görüntü kullanımı ile eğitilen model için Darknet-53 ile elde edilen dönüm- kayıp fonksiyon (mavi) ve dönüm-mAP (kırmızı) değerleri**

Kayıp fonksiyonu tasarlanan modelin hata oranını aynı zamanda başarımını ölçen fonksiyondur. Bu fonksiyon, hata hesaplama işini bir optimizasyon problemine dönüştürerek yaptığı için optimizasyon terminolojisinde kullanılan "objective function", "cost function" isimleriyle de tanımlanmaktadır. Kayıp fonksiyonu temelde modelin yaptığı tahminin, gerçek değerden (İng. Ground truth) ne kadar farklı olduğunu hesaplamaktadır. Bu nedenle iyi tahmin eden bir model oluşturulamazsa, gerçek değer ile tahmin edilen değer arasındaki fark yüksek olur ve dolayısıyla kayıp değeri yüksek olur. Daha iyi bir model oluşturulmuşsa kayıp değeri daha az olacaktır. Hatasız kestirim durumunda ise kayıp değeri 0 olacaktır. İlk eğitilen modelin eğitim esnasında kayıp fonksiyonu ve mAP sonuçlarından elde edilen değerlere bağlı veri seti ve işlem adımları

ile ilgili bazı hatalar tespit edilmiştir. Görüntüyü test ve tahmin etmeden önce aşağıdaki işlemlerin yapılması gerektiği sonucuna varılmıştır. Böylece ikinci eğitim süreci ve tahmin testleri öncesi işlemler yapılmıştır. YOLOv4 algoritmasında konvolüsyon katmanı girdisi yatay ve dikey eksende eşit çözünürlüğe ihtiyaç duyduğundan 1024x760 olacak şekilde 16 parçaya bölünüp ardından YOLOv4 algoritmasının görüntü üzerinde detay ve veri kaybına uğratacak çözünürlük düşürme işlemlerinin önüne geçilip her parça için algoritma çalıştırılmıştır. Yapılan çalışmada tespit edilen sınıfların yükseklik genişlik ölçüleri, yatay ve dikey eksendeki konumları dahil olmak üzere hepsi her parça için hesaplanıp tekrar 4056x3040 çözünürlük üzerinde çizdirilmiştir. Bu modelin eğitiminde mAP değerlerinin yüksek ve kayıp değerlerinin düşük çıkmasının en önemli sebebi parçalama yönteminin uygulanması olduğu tespit edilmiştir. 4056x3040 çözünürlüklü bir görüntüde 80x100 çözünürlüğe sahip nesneler ararken görüntü hücreler şeklinde parçalandıktan sonra 1024x760 çözünürlüğünde 80x100 aynı piksel değerlerine sahip nesnelerin tespiti ve tahmini daha doğru sonuçlara yol açmıştır. Hazırlanan veri setinde renkli görüntüler kullanılarak sistemin gri seviye görüntülere göre baskı devre kartlarının renkli olması sebebiyle daha kararlı olduğu sonuçlandırılmıştır. Elde edilen veri setinde görüntüden elde edilen 16 parçanın her biri 30 derecelik açılar ile döndürülerek her parçadan 360 dereceyi tamamlayacak şekilde kendisi dahil 12 adet görüntü elde edilmiştir. Bu işlemler sonrası 11 gigabyte boyutunda yüksek boyutlu bir veri seti elde edilmiştir. Eğitilecek modelde veri setinde bulunan görüntülerin ve örnek etiketlerin çok olması sistemin kararlı ve yüksek oranlarda nesne takibi yapabildiğine sebep olduğu gözlemlenmiştir. Bu gözlemlerin sonunda yapılan veri seti çalışmasında yaklaşık olarak 55000 adet lehimsiz sınıfı etiketleme işlemi gerçekleştirilmiştir.




**Şekil 6.2: Renkli görüntü ve bölütleme kullanımı ile eğitilen model için Darknet-53 ile elde edilen dönüm - kayıp fonksiyonu (mavi) ve dönüm - mAP (kırmızı) değerleri**

Yapılan ikinci eğitim sonunda mAP değerlerinin sisteme verilen test görüntüleri sonrasında ciddi bir şekilde %98 seviyelerine geldiği tespit edilmiştir. 400 adet imgeden oluşan doğrulama setinin Darknet ağı üzerinden komut istemi ile kesişme bölgeleri eşik değerleri 0.1% ile 100% arasında yüzde 10 arttırılarak elde edilen sonuçlar **Şekil 6.3'de** gösterilmiştir.

| Seçilen IoU Eşik Değeri | Doğruluk | Ortalama Hassasiyet | Geri Çağırma | Avg IoU | F1 Skor | DP | YP | YN |
|---|---|---|---|---|---|---|---|---|
| **0.1%** | 0.82% | 96.59% | 0.97% | 61.22 % | 0.89% | 771 | 174 | 20 |
| **0.2%** | 0.82% | 96.51% | 0.97% | 61.42 % | 0.89% | 771 | 174 | 20 |
| **0.3%** | 0.81% | 96.37% | 0.97% | 61.39 % | 0.89% | 770 | 175 | 21 |
| **0.4%** | 0.81% | 95.46% | 0.96% | 61.16 % | 0.88% | 763 | 182 | 28 |
| **0.5%** | 0.78% | 91.01% | 0.93% | 59.92 % | 0.85% | 738 | 207 | 53 |
| **0.6%** | 0.72% | 81.01% | 0.86% | 56.56 % | 0.78% | 681 | 264 | 110 |
| **0.7%** | 0.57% | 52.87% | 0.68% | 46.75 % | 0.62% | 539 | 406 | 252 |
| **0.8%** | 0.33% | 18.19% | 0.36% | 28.53 % | 0.36% | 310 | 635 | 481 |
| **0.9%** | 0.08% | 14.39% | 0.10% | 7.52 % | 0.11% | 77 | 868 | 714 |

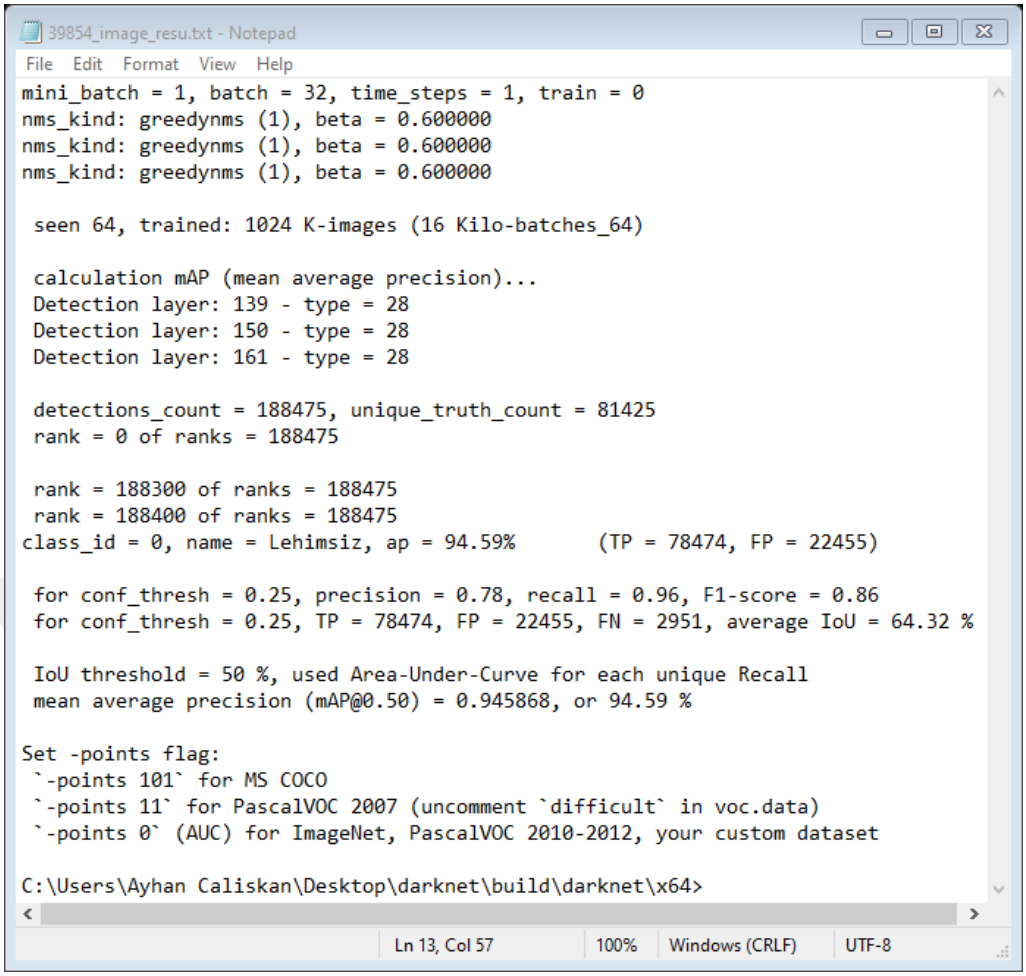
**Şekil 6.3: Oluşturulan doğrulama setinin Darknet ağı ile elde edilen sonuçları**

Cihaz olarak ortaya çıkartılan baskılı devre kartı lehim hatası tespit cihazının gerçek bir baskılı devre kartı üretim hattında test edebilme fırsatı veren firmada cihazın ve modelin doğruluğunu sağlayıp test sonuçlarını elde edebilmek için cihaz üretim hattında otomatik lehimleme istasyonu sonrası çalıştırılmaya başlamıştır (**Şekil 6.4**).

**Şekil 6.4: Üretim hattında cihaz görüntüsü ve testi**

**Şekil 6.5: Cihazın üretim hattında karışık kartlar ile testi**

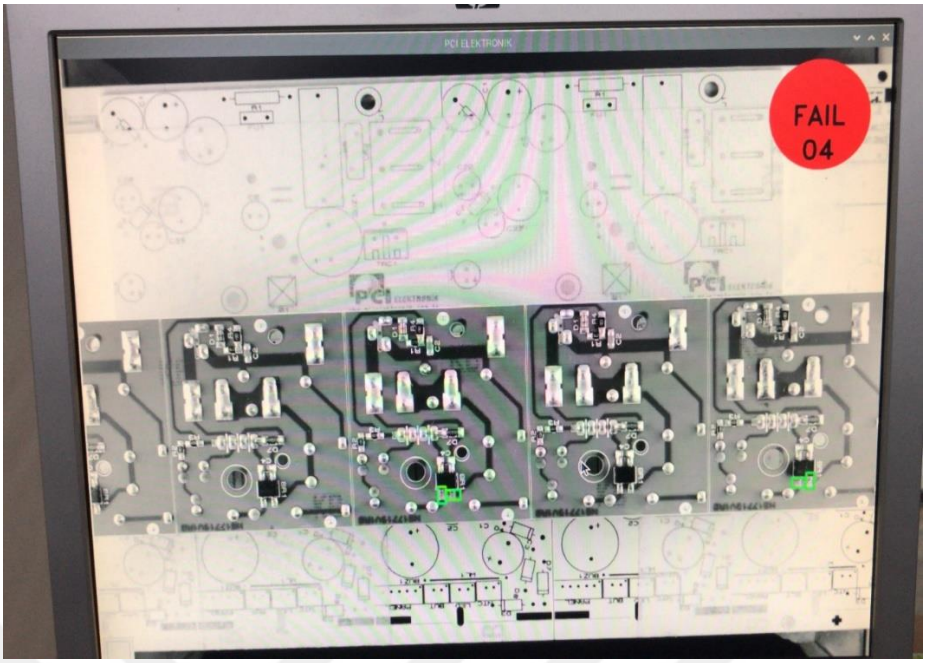
Hata tespit cihazının üretimde 2 haftalık test süreci ve başarı takibi operatör sorumluları tarafından yapılmıştır. Test edilen tüm kartların görüntüleri ve lehimsiz sınıfı hata tespiti sonuçları gün bazında klasörlenerek bilgisayara kaydedilmiştir. Kaydedilen bu görüntüler tek tek incelenerek cihazın güvenilirlik oranı hesaplanmıştır. Üretim hattında yaklaşık olarak günlük 1500 kart geçerken hergün sürekli olarak 500 adet panellenmiş kart testi yapılabilmiştir. Yapılan testlerde kart farklı açılarda döndürüldüğünde ve çevirildiğinde görüntüler tahmin yapılarak cihaz test edilmiştir. Üretim hattında gerçek hayattan elde edilen 30.000 görüntüden oluşan test setinin tüm görüntüleri toplanıp tümüne komut istemi ile map komutu uygulanarak mAP@0.50 sonuçları elde edilmiştir (**Şekil 6.6**).




**Şekil 6.6: Üretim hattından toplanan tüm test görüntülerinin Darknet-map komutu sonuçları**

Hata tespit cihazının kayıp olmadan lehimsiz sınıfını kartın açısı ve yönü farketmeksizin doğru tespit ettiği görülmüştür (**Şekil 6.7, Şekil 6.8**). Böylece eğitim yapılmadan önce aynı girdi görüntülerinin ve veri etiketlerinin farklı açılarda sisteme verilmesi yöntemi önerilmiştir.


**Şekil 6.7: Test aşaması tahmin sonucu: Karta herhangi bir rotasyon uygulanmamış ve lehimsiz bölgeler doğru tespit edilmiş.**




**Şekil 6.8: Test aşaması tahmin sonucu: Karta 180º rotasyon uygulanmış ve lehimsiz bölgeler doğru tespit edilmiş.**

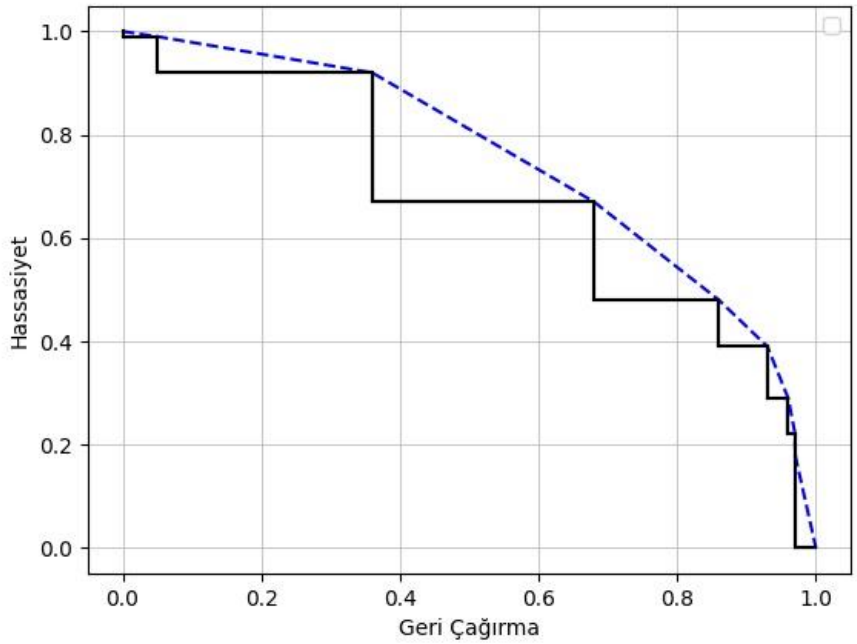

**Şekil 6.9: Yüksek hatalı kart için test aşaması tahmin sonucu. 77 adet lehimsiz bacak doğru olarak kestirilmiş.**

Bu testler sonucunda cihazın lehimsiz sınıfı tespit etme başarısı %92 olarak hesaplanmıştır. Tahmin etme oranları %98 seviyelerinde iken başarı oranının %92 seviyelerinde çıkmasının sebebi araştırılmıştır. Yapılan araştırma sonucunda aydınlatma sisteminin sağ ve sol taraflarda olması ve komponentlerin yakınlıkları sebebi ile birbirine uyguladığı gölgelemeden dolayı sistemi az miktar da olsa hatasız lehim olmasına rağmen fazladan lehimsiz sınıfı tespit etmesine neden olduğu görülmüştür (**Şekil 6.10**). Bu problemin iyileştirilmesi için sabit parlaklık ve doygunluk gibi parametrelerin el ile ayarlanmasının hatayı azalttığı gözlemlenmiştir.

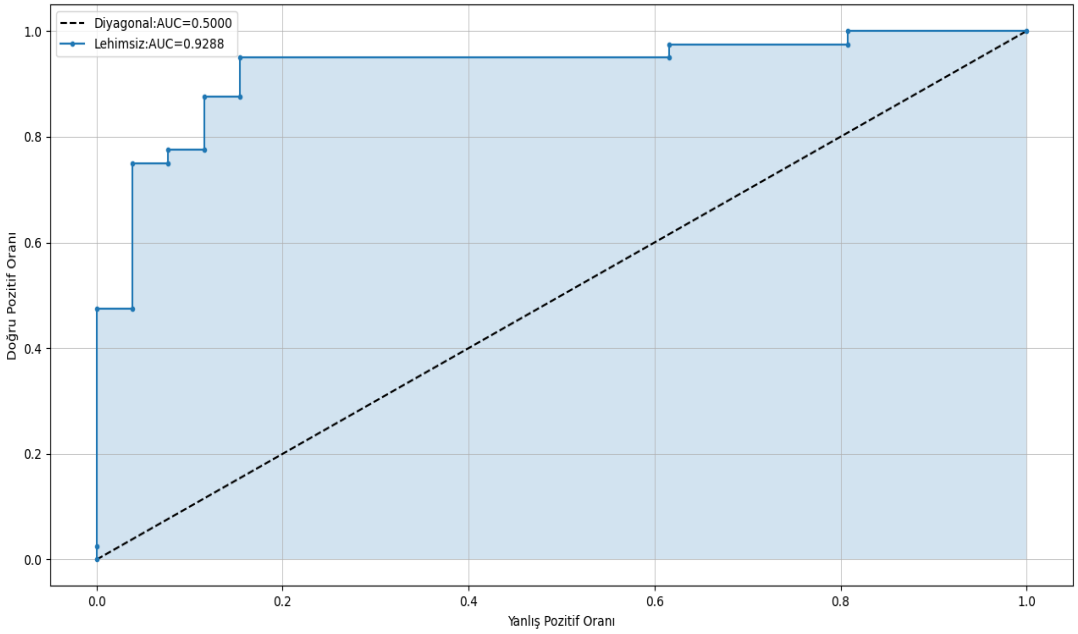

**Şekil 6.10: Fazla lehimsiz sınıfı tespiti örneği**

# 7. TARTIŞMA VE SONUÇ

Tasarlanan sistemde YOLO-v4 algoritması kullanılarak baskılı devre kartlarının üzerindeki devre elemanlarının pinlerinin lehimleme problemlerinin tespiti için Lehimsiz sınıfı oluşturulup yapılan eğitim ile baskılı devre kartları üzerindeki devre elamanlarının tüm pinlerinin açık devre durumunda olup olmadığını tespit edebilen model ile tespiti hedeflenmiştir. Yazarın çalıştığı firmada Darknet ağı ile YOLO-v4 algoritması kullanılarak 8000 dönüm olarak ayarlanan eğitim adımı işlemi, 96 saatlik süren işlem süresi sonunda YOLO modeli elde edilmiştir. Eğitim için kullanılan YOLO-v4 algoritmasının verdiği bu model sadece Lehimsiz sınıfı için yaklaşık 55000 adet Lehimsiz sınıfı etiketine sahiptir. Eğitimde yazarın çalıştığı firmanın üretim hattında bulunan 11 farklı model kart üzerinden etiketleme işlemi yapılmıştır. Doğrulama ve performans adımları için eğitimde kullanılan görüntülerin içermediği 400 adet görüntüyü içeren doğrulama seti oluşturulmuştur. Yapılan eğitimde Lehimsiz sınıfı için oluşturulan doğrulama seti kullanılarak Darknet ağından alınan sonuçlar ile Hassasiyet-Geri Çağırma ve ROC eğrileri çizdirilmiştir. Kullanılan algoritmanın AUC puanı 92.88% ile en iyi sonucunu vermiştir. Kullanılan bu görüntüler ile sınıflandırma modelinin gerçek değerlerinin bilindiği bir dizi test verisi üzerindeki performansını tanımlamak için DP, YP, YN, DN değerleri Darknet ağı üzerinden doğrulama sonuçları ile elde edilmiştir. Ona ait Hassasiyet- Geri Çağırma ve ROC eğrisi **Şekil 7.1 ve Şekil 7.2'de** gösterilmiştir. Önerilen bilgisayar destekli system, baskılı devre kartları üzerinde hata tanısı ve izlem sürecinde üretim hattında hızını ve doğruluk oranını arttırmada destek sağlayabilecektir.




**Şekil 7.1: Hassasiyet-Geri Çağırma Eğrisi**




**Şekil 7.2: ROC Eğrisi**

Literatürde makine öğrenmesi ve yapay sinir ağları kullanılarak yapılan çalışmalara bakıldığında YOLO algoritması kullanılarak yapılan bu çalışmaya benzer elektronik devre kartlarında devre elemanlarının açık devre ve kısa deve üzerinde çalışılan bir uygulama görülmemiş olması tez çalışmasının özgünlüğünü önemli ölçüde ön plana koymaktadır.

Sadece "Lehimsiz" sınıfı kullanılarak gerçekleştirilen eğitime alternatif olarak "Kısa devre" sınıfının da tanıtılması ile başarım yüzdesinin arttırılması gelecek çalışma planını oluşturmaktadır.

Yapılan sistemin test sırasında ortamda bulunan ışık kaynaklarından olumsuz etkilendiği gözlemlenmiştir. Sonraki çalışmalarda sistemin kapalı bir ortam içerisinde olacak şekilde üretim hattı üzerinde sabitlenmesi önerilmektedir.

**KAYNAKÇA**


[1] N. Dave, V. Tambade, B. Pandhare, and S. Saurav, "PCB Defect Detection Using Image Processing And Embedded System," *Int. Res. J. Eng. Technol.*, 2016, [Online]. Available: www.irjet.net.

[2] M. P.S. and N. R. S., "PCB Defect Detection, Classification and Localization using Mathematical Morphology and Image Processing Tools," *Int. J. Comput. Appl.*, vol. 87, no. 9, pp. 40–45, 2014, doi: 10.5120/15240-3782.

[3] S. H. I. Putera, S. F. Dzafaruddin, and M. Mohamad, "MATLAB based defect detection and classification of printed circuit board," *2012 2nd Int. Conf. Digit. Inf. Commun. Technol. its Appl. DICTAP 2012*, pp. 115–119, 2012, doi: 10.1109/DICTAP.2012.6215366.

[4] S. H. Indera Putera and Z. Ibrahim, "Printed circuit board defect detection using mathematical morphology and MATLAB image processing tools," in *ICETC 2010 - 2010 2nd International Conference on Education Technology and Computer*, 2010, doi: 10.1109/ICETC.2010.5530052.

[5] S. Kaushik and J. Ashraf, "AUTOMATIC AUTOM ATIC PCB DEFECT DETECTION USING," vol. 1, no. 5, 2012.

[6] M. Kumar, M. Kumar, and G. Kumar, "PCB Image Enhancement Using Machine Vision For Effective Defect Detection," no. 3, pp. 50–53, 2014.

[7] A. Doniwar, A. Dadhe, and A. Baiswara, "PCB Faults Detection Using Image Processing," pp. 4–8, 2017.

[8] V. Nattarasu, *An Image Based PCB Fault Detection And Its Classification*. 2017.

[9] P. Hankare, V. Patil, D. Solanki, S. Tikke, and S. Vishwakarma, "Circuit Board Defect Detection Using Image," pp. 3–6, 2017.

[10] V. Chaudhary, I. R. Dave, and K. P. Upla, "Automatic visual inspection of printed circuit board for defect detection and classification," *Proc. 2017 Int. Conf. Wirel. Commun. Signal Process. Networking, WiSPNET 2017*, vol. 2018-Janua, pp. 732–737, 2018, doi: 10.1109/WiSPNET.2017.8299858.

[11] W. C. Wang, S. L. Chen, L. B. Chen, and W. J. Chang, "A Machine Vision Based Automatic Optical Inspection System for Measuring Drilling Quality of Printed Circuit Boards," *IEEE Access*, vol. 5, no. November, pp. 10817–10833, 2017, doi: 10.1109/ACCESS.2016.2631658.

[12] D.-M. Tsai and C.-K. Huang, "Defect Detection in Electronic Surfaces Using Template-Based Fourier Image Reconstruction," *IEEE Trans. Components, Packag. Manuf. Technol.*, vol. 9, no. 1, pp. 163–172, Jan. 2019, doi: 10.1109/TCPMT.2018.2873744.

[13] R. Heriansyah, S. A. R. Al-attas, and M. M. Ahmad Zabidi, "Neural Network Paradigm for Classification of Defects on PCB," *J. Teknol.*, vol. 39, no. 1, 2003, doi: 10.11113/jt.v39.465.

[14] J. H. Kim and H. S. Cho, "Neural network-based inspection of solder joints using a circular illumination," *Image Vis. Comput.*, vol. 13, no. 6, pp. 479–490, 1995, doi: 10.1016/0262-8856(95)94381-9.

[15] B. Hu and J. Wang, "Detection of PCB Surface Defects With Improved Faster-RCNN and Feature Pyramid Network," *IEEE Access*, vol. 8, pp. 108335–108345,

2020, doi: 10.1109/ACCESS.2020.3001349.
[16] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.
[17] C. Zhang, W. Shi, X. Li, H. Zhang, and H. Liu, “Improved bare PCB defect detection approach based on deep feature learning,” *J. Eng.*, vol. 2018, no. 16, pp. 1415–1420, Nov. 2018, doi: 10.1049/joe.2018.8275.
[18] T. Y. Ong, Z. Samad, and M. M. Ratnam, “Solder joint inspection with multi-angle imaging and an artificial neural network,” *Int. J. Adv. Manuf. Technol.*, vol. 38, no. 5–6, pp. 455–462, 2008, doi: 10.1007/s00170-007-1117-6.
[19] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 779–788, 2016, doi: 10.1109/CVPR.2016.91.
[20] Y. L. Lin, Y. M. Chiang, and H. C. Hsu, “Capacitor Detection in PCB Using YOLO Algorithm,” *2018 Int. Conf. Syst. Sci. Eng. ICSSE 2018*, pp. 4–7, 2018, doi: 10.1109/ICSSE.2018.8520170.
[21] J. Li, J. Gu, Z. Huang, and J. Wen, “Application research of improved YOLO V3 algorithm in PCB electronic component detection,” *Appl. Sci.*, vol. 9, no. 18, 2019, doi: 10.3390/app9183750.
[22] R. Huang, J. Gu, X. Sun, Y. Hou, and S. Uddin, “A rapid recognition method for electronic components based on the improved YOLO-V3 network,” *Electron.*, vol. 8, no. 8, 2019, doi: 10.3390/electronics8080825.
[23] P. Wei, C. Liu, M. Liu, Y. Gao, and H. Liu, “CNN-based reference comparison method for classifying bare PCB defects,” *J. Eng.*, vol. 2018, no. 16, pp. 1528–1533, Nov. 2018, doi: 10.1049/joe.2018.8271.
[24] H. S. Cho, “Automated inspection of printed circuit boards,” *Intell. Syst. Technol. Appl. Six Vol. Set*, vol. 28, no. 95, pp. V-293-V–325, 2002, doi: 10.1201/9781420040814.ch12e.
[25] J. F. Borba and J. Facon, “Printed circuit board automated inspection system,” *Midwest Symp. Circuits Syst.*, vol. 1, pp. 69–72, 1995, doi: 10.1109/mwscas.1995.504380.
[26] D. M. Tsai and C. H. Yang, “A quantile-quantile plot based pattern matching for defect detection,” *Pattern Recognit. Lett.*, vol. 26, no. 13, pp. 1948–1962, 2005, doi: 10.1016/j.patrec.2005.02.002.
[27] S. Singh, “Image Processing Based Automatic Visual Inspection System for PCBs,” *IOSR J. Eng.*, vol. 02, no. 06, pp. 1451–1455, 2012, doi: 10.9790/3021-026114511455.
[28] B. Kaur, G. Kaur, and A. Kaur, “Detection and classification of Printed circuit board defects using image subtraction method,” *2014 Recent Adv. Eng. Comput. Sci. RAECS 2014*, pp. 6–8, 2014, doi: 10.1109/RAECS.2014.6799537.
[29] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-Based Learning Applied to Document Recognition,” *proc. IEEE*, 1998, [Online]. Available: http://ieeexplore.ieee.org/document/726791/#full-text-section.
[30] T. F. Gonzalez, “Handbook of approximation algorithms and metaheuristics,” *Handb. Approx. Algorithms Metaheuristics*, pp. 1–1432, 2007, doi: 10.1201/9781420010749.

[31] I. Namatēvs, “Deep Convolutional Neural Networks: Structure, Feature Extraction and Training,” *Inf. Technol. Manag. Sci.*, vol. 20, no. 1, pp. 40–47, 2018, doi: 10.1515/itms-2017-0007.

[32] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” 2020, [Online]. Available: http://arxiv.org/abs/2004.10934.

[33] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–14, 2015.

[34] C. Y. Wang, H. Y. Mark Liao, Y. H. Wu, P. Y. Chen, J. W. Hsieh, and I. H. Yeh, “CSPNet: A new backbone that can enhance learning capability of CNN,” *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work.*, vol. 2020-June, pp. 1571–1580, 2020, doi: 10.1109/CVPRW50498.2020.00203.