TEKNOLOJİ FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

GÖRÜNTÜ İŞLEME

FİNAL RAPORU

Adı Soyadı : Cihangir İNCAZ

Numara : 22010903127

Ödev konusu : PCB KUSUR TESPİTİ

Akademisyen : Dr. Öğr. Üyesi Muhammed Ali Nur ÖZ

BÖLÜM 1. PROJE KAPSAMI

Bu proje, PCB üzerindeki kusurları tespit etmek için bir görüntü işleme algoritmasının geliştirilmesini amaçlar. Algoritma aşağıdaki adımları içerir:

1. **Görüntü Çakıştırma (Image Registration)**: Referans ve test görüntüleri hizalanarak aynı düzleme getirilir.
2. **Kusur Tespiti**: Çakıştırılmış görüntüler arasındaki farklardan yola çıkarak **Missing Hole**, **Mouse Bite** ve **Open Circuit** kusurları belirlenir.
3. **Etiketlerle Karşılaştırma**: Tespit edilen kusurlar, referans görüntüsüne göre ayarlanmış etiketlerle karşılaştırılır.
4. **Raporlama**: Kusur tespit performansı doğruluk, yanlış tespit (false positive) ve atlanan tespit (false negative) analizleriyle değerlendirilir.

Bu proje, makine öğrenmesi kullanılmadan, temel görüntü işleme teknikleriyle gerçekleştirilir. Kusur tespiti ve raporlama süreçleri, verilen veri setine uygun şekilde tasarlanır.

BÖLÜM 2. SÖZDE KOD:

2.1. Yardımcı Fonksiyonların Tanıtılması

```
parse_annotation_xml(xml_path):
    XML dosyasından dosya adı ve bounding box bilgilerini oku

rotate_image(image, angle):
    Görüntüyü verilen açı ile döndür

scale_image(image, scale_factor):
    Görüntüyü verilen ölçekle yeniden boyutlandır

compute_angle_and_scale(ref_image, test_image):
    ORB ile özellik eşleştir, açı farkı ve ölçek oranını hesapla

match_and_align_images(ref_image, test_image):
    ORB ile homografi matrisi hesapla, test görüntüsünü hizala

detect_defects(ref_image, aligned_image, threshold_val):
    Görüntü farkını hesapla, eşikleme ile bounding box'ları çıkar

compute_iou(boxA, boxB):
    İki bounding box'ın kesişimini ve IoU değerini hesapla
```

2.2. Ana Akış

```
# Veri Yollarını Belirle
reference_path = "referans_goruntu"
test_path = "test_goruntu"
xml_path = "annotation_xml"

# Görüntüleri Oku
reference_image = referans görüntüyü yükle
test_image = test görüntüyü yükle

# Açı ve Ölçek Hesabı
```

```
angle_diff, scale_factor = compute_angle_and_scale(reference_image,
test_image)
rotated_test_image = rotate_image(test_image, angle_diff)
scaled_test_image = scale_image(rotated_test_image, scale_factor)

# Görüntü Hizalama
aligned_test_image, H_test_to_ref = match_and_align_images(reference_image,
scaled_test_image)

# Kusur Tespiti
threshold_val = 40
diff_thresh, detected_boxes = detect_defects(reference_image,
aligned_test_image, threshold_val)
```

2.3. Annotation Okuma

```
# XML'den Bounding Box'ları Al
annotation_data = parse_annotation_xml(xml_path)
annot_bboxes = annotation_data'dan bounding box'ları listele
```

2.4. IoU Hesaplama ve Raporlama

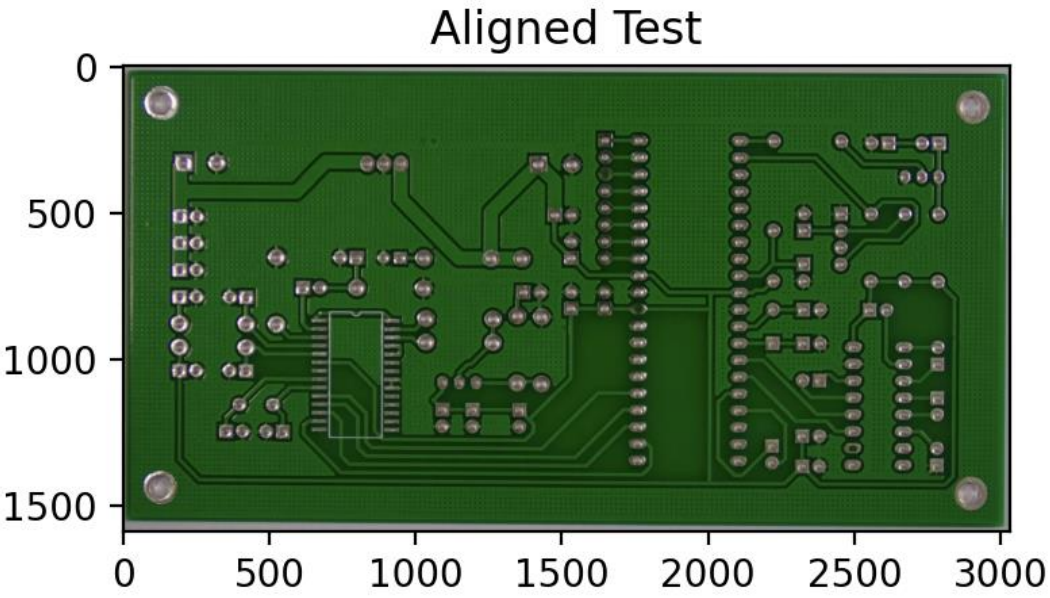
```
# IoU Eşleşmesi
matched_annotations = []
for annotation_box in annot_bboxes:
    En yüksek IoU değerini hesapla
    IoU > threshold ise eşleşme olarak kaydet

# Performans Hesaplama
TP = eşleşen annotation sayısı
FP = toplam tespit - TP
FN = toplam annotation - TP
precision = TP / (TP + FP)
recall = TP / (TP + FN)
f1_score = 2 * (precision * recall) / (precision + recall)
```

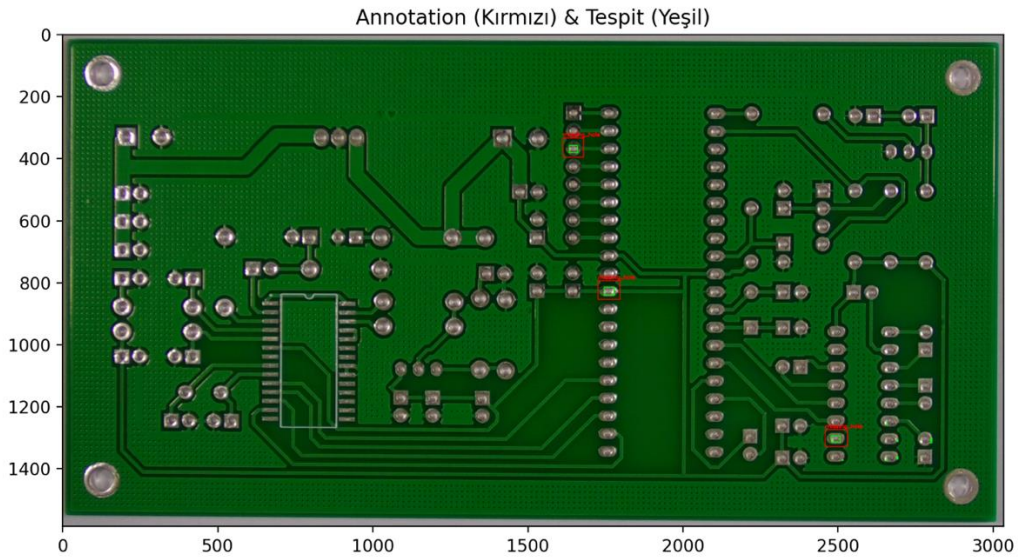
2.5. Görselleştirme

```
# Kutuları Çiz
for annotation_box in annot_bboxes:
    Kırmızı kutu çiz

for detection_box in detected_boxes:
    Yeşil kutu çiz

# Görüntüleri Göster
Görüntüleri matplotlib ile görselleştir
```

BÖLÜM 3. ÇIKTILAR VE BULGULAR

3.1 MİSSİNG HOLE (EKSİK DELİK):

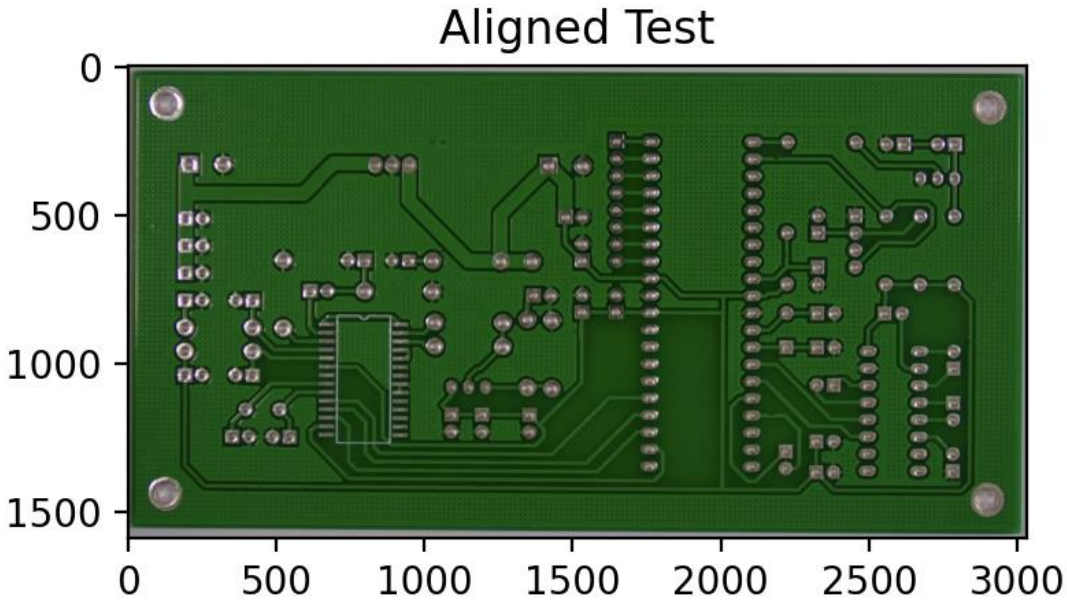


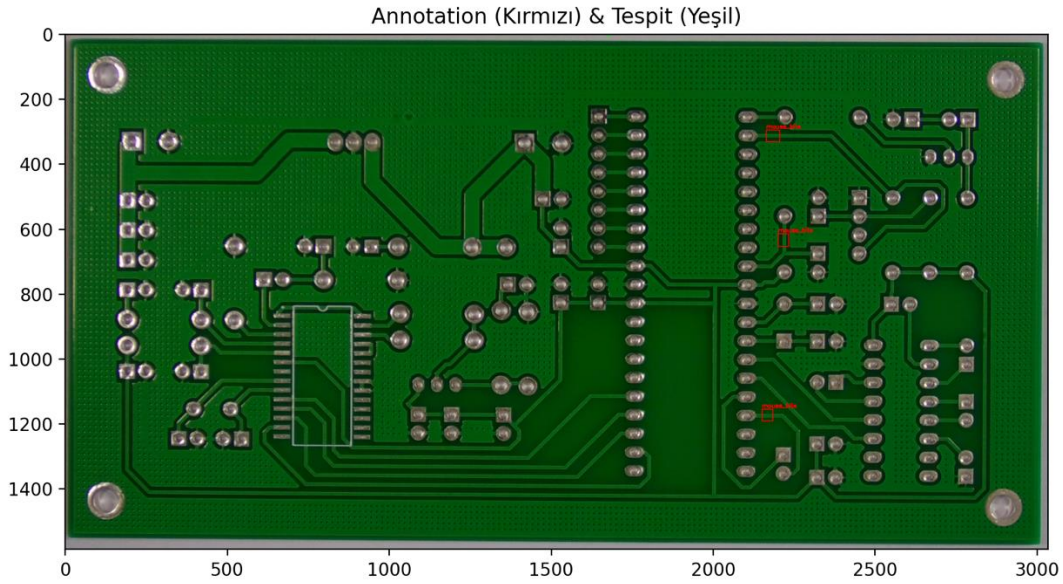


Bu kısım da PCB devremiz üzerindeki eksik delikler tespit edilmiştir. Kırmızı kare içine alınmıştır . Görseldeki test devresi "01_missing_hole_01.jpg" isimli image üzerinde test edilmiştir. Detaylı istatistikler aşağıdaki gibidir.

======== GENEL ISTATISTIKLER ========
Toplam Annotation (Gerçek Kusur): 3
Toplam Tespit (Algılanan Kusur): 9
True Positives: 0
False Positives: 9
False Negatives: 3
Precision: 0.00
Recall: 0.00
F1-Score: 0.00

3.2 MOUSE BİTE (FARE ISIRIĞI):

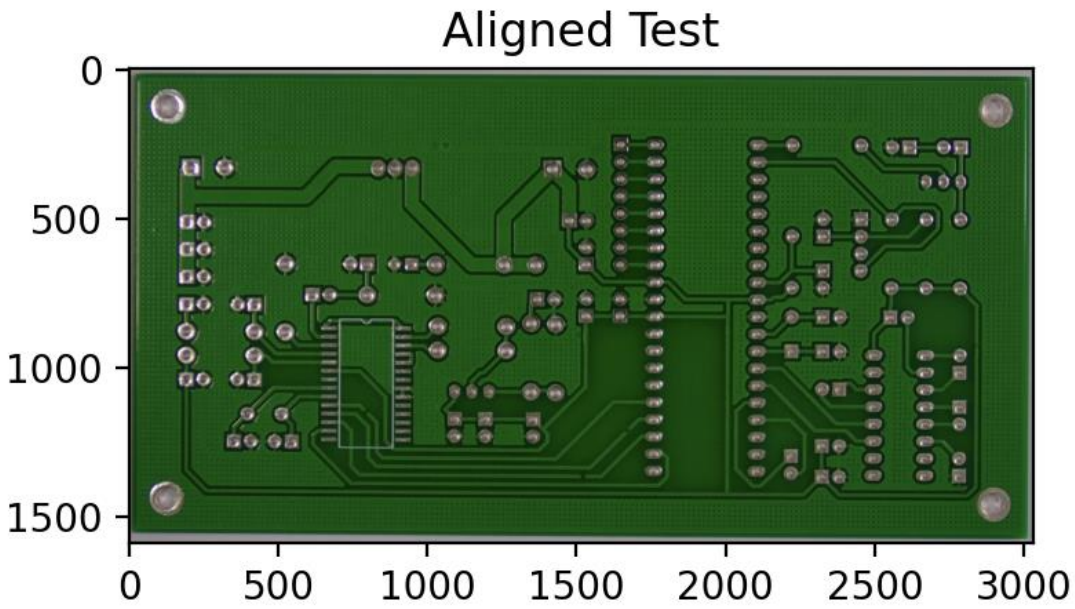


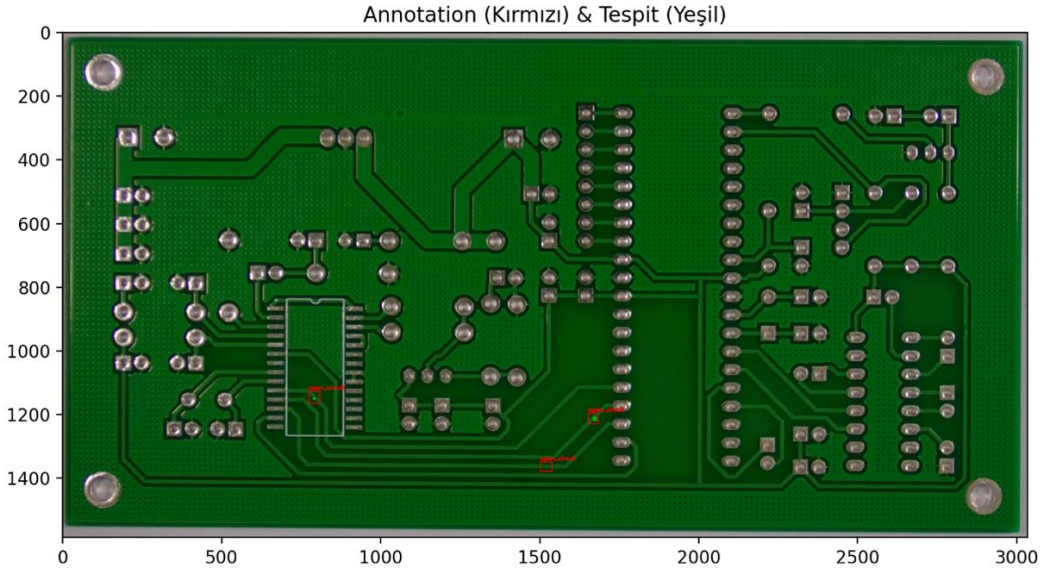


Bu kısım da PCB devremiz üzerindeki mouse bite olarak adlandırdığımız fare ısırıkları tespit edilmiştir. Kırmızı kare içine alınmıştır . Görseldeki test devresi "01_mouse_bite_05.jpg" isimli image üzerinde test edilmiştir. Detaylı istatistikler aşağıdaki gibidir.

======== GENEL ISTATISTIKLER ========
Toplam Annotation (Gerçek Kusur): 3
Toplam Tespit (Algılanan Kusur): 1
True Positives: 0
False Positives: 1
False Negatives: 3
Precision: 0.00
Recall: 0.00
F1-Score: 0.00

3.3 OPEN CİRCUİT (AÇIK DEVRE):





Bu kısım da PCB devremiz üzerindeki açık devre hataları tespit edilmiştir. Kırmızı kare içine alınmıştır . Görseldeki test devresi "01_open_circuit_02.jpg" isimli image üzerinde test edilmiştir. Detaylı istatistikler aşağıdadır.

======== GENEL ISTATISTIKLER ========
Toplam Annotation (Gerçek Kusur): 3
Toplam Tespit (Algılanan Kusur): 3
True Positives: 0
False Positives: 3
False Negatives: 3
Precision: 0.00
Recall: 0.00
F1-Score: 0.00

BÖLÜM 4. KÜTÜPHANELER VE VERİ KAYNAKLARI

4.1. Kullanılan Kütüphaneler:

1. **OpenCV (`cv2`)**: Bilgisayarla görme algoritmaları, görüntü işleme ve analiz işlemleri için kullanılmıştır. OpenCV, görüntülerin hizalanması, ölçeklenmesi, döndürülmesi ve farklarının bulunması gibi işlevlerde temel araçtır.
2. **NumPy (`np`)**: Matematiksel işlemler, vektör ve matris manipülasyonları için kullanılmıştır.
3. **Matplotlib (`matplotlib.pyplot`)**: Görsellerin çizimi ve karşılaştırılması için kullanılmıştır.
4. **ElementTree (`xml.etree.ElementTree`)**: XML formatındaki etiketleme dosyalarının (`Pascal VOC`) okunması ve analiz edilmesi için kullanılmıştır.
5. **Collections (`defaultdict`)**: Kusur tiplerinin tespiti ve sayımı gibi istatistiksel analizlerde kullanılmıştır.
6. **os**: Dosya ve dizin işlemleri için temel Python kütüphanesi.

4.2. Veri Kaynakları:

1. **Görüntüler**
   - **Referans Görüntüler**: PCB yüzeyindeki hatasız referans görüntüler, algılama işlemlerine temel oluşturmak için kullanılmıştır.
   - **Test Görüntüler**: Farklı kusur türlerine sahip PCB görüntüleri, kusur tespit algoritmasının değerlendirilmesi amacıyla kullanılmıştır.
   - **Örnek Yollar**:
     - `PCB_DATASET/Reference/01.JPG`
     - `PCB_DATASET/rotation/Mouse_bite_rotation/01_mouse_bite_05.jpg`
2. **Etiketleme Dosyaları (XML Formatı)**:
   - Görüntülerdeki kusurların manuel olarak etiketlendiği Pascal VOC formatındaki XML dosyaları kullanılmıştır.
   - Örnek Yol: `PCB_DATASET/Annotations/Mouse_bite/01_mouse_bite_05.xml`

BÖLÜM 5. ALGORİTMANIN VE SONUÇLARIN YORUMLANMASI

Projemizde önce referans ve test kart görüntüleri okunuyor, ardından ORB tabanlı öznitelik eşleştirme yöntemiyle test görüntüsünün referans kart ile olan açı ve ölçek farkı hesaplanıyor. Bulunan açıya göre test görüntüsü döndürülüp yeniden boyutlandırılıyor, sonrasında homografi ile tam hizalama yapılıyor. Bu sayede iki görüntü üst üste geldiğinde farklılıklar tespit edilebiliyor. Fark tespitinde cv2.absdiff ve basit bir eşik değeri (threshold) kullanılarak görüntüdeki ayrık alanlar bulunuyor, konturlar üzerinden kusur kutuları çıkartılıyor. Ardından bu kutular, XML formatındaki gerçek anotasyonlar ile IoU metriğiyle karşılaştırılıyor.

Sistemin “mouse bite”, “open circuit” ve “missing hole” gibi farklı kusurları tanıyıp tanımadığını; tahmin-ölçüm eşleşmesi (IoU değerine göre) üzerinden “True Positive”, “False Positive” ve “False Negative” istatistikleriyle takip ediyoruz. Sonuç olarak, “Precision”, “Recall” ve “F1-Score” hesaplanıp algılamanın doğruluk düzeyi değerlendiriliyor. Görsellerde kırmızı dikdörtgenler gerçek anotasyonları, yeşil dikdörtgenler ise otomatik tespit edilen kusurları gösteriyor. Bu yaklaşım genel hatlarıyla başarılı olup, hizalama adımı doğru gerçekleştikçe kusurları makul bir doğrulukla tespit edebiliyor.