

# GİRİŞ:

Bu projede bir printed circuit board kontrol sistemi geliştirdik. Amacımız belirli bir referans görüntü kullanarak, test görüntülerindeki hataları tespit etmektir. Hatalı bölgeleri işaretleyecek ve sonuçları kaydedeceğiz.

## GÖRÜNTÜ ÇAKIŞTIRMA:

Bu projeyi yapabilmek için ilk önce referans görüntüyü alıp, test görüntüleriyle aralarında ortak noktalar bulunur ve görüntüleri birbirlerine hizalanır.

1. **Özellik Algılama:**

* **ORB (Oriented FAST and Rotated BRIEF):** OpenCV'nin hızlı ve hafif bir özellik algılama ve eşleme algoritmasıdır. Görüntünün önemli kısımlarını temsil eden "anahtar noktaları" (keypoints) belirler. Aynı zamanda bu noktaların çevresindeki bilgileri açıklayan "açıklayıcılar" (descriptors) üretir.

2. **ORB ile Anahtar Nokta ve Açıklayıcıların Bulunması:**

* orb.detectAndCompute yöntemi ile hem referans görüntü hem de test görüntüsü için:
  + Anahtar noktalar (kp1, kp2).
  + Açıklayıcılar (des1, des2) oluşturulur.
* Eğer bir görüntü için anahtar noktalar bulunamazsa, bu görüntü atlanır.

3. **Özellik Eşleme:**

* cv2.BFMatcher (Brute-Force Matcher) ile referans görüntü ve test görüntüsü arasındaki açıklayıcılar karşılaştırılır.
* crossCheck=True ile eşleşmeler iki yönlü doğrulanır (test noktasından referansa ve tersi doğru eşleşme kontrolü yapılır).
* Eşleşmelerin sonuçları mesafelere göre sıralanır (en iyi eşleşmeden başlayarak).

4. **Eşleşme Kontrolü:**

* Eğer eşleşme sayısı 4’ten azsa, işlem başarısız olarak değerlendirilir ve test görüntüsü işlenmez.

## HOMOGRAFİ HESAPLAMA:

Referans görüntümüzdeki geometriği bütün test görüntülerine uygulayarak, daha kolay bir karşılaştırma yapılabilir. Bunu kaydırma, çevirme veya ölçeklendirme kullanarak yapar.

1. **Kaynak ve Hedef Noktaların Seçimi:**
   * İyi eşleşmiş olan özellik noktalarından ilk 10 eşleştirme seçilir.
   * Bu noktaların referans (kaynak) görüntüdeki (src\_pts) ve test görüntüsündeki (dst\_pts) koordinatları alınır.
2. **Homografi Matrisi Hesaplama:**
   * cv2.findHomography fonksiyonu, bu noktalar arasındaki dönüşümü ifade eden homografi matrisini oluşturur:
     + RANSAC yöntemi (Random Sample Consensus) kullanılarak, hatalı eşleşmeler filtrelenir.
   * Eğer yeterli sayıda güvenilir nokta bulunamazsa ya da homografi matrisi oluşturulamazsa işlem atlanır.

## GÖRÜNTÜYÜ HİZALAMA:

Homografi matrisi kullanılarak test görüntümüzü, referans görüntüleriyle hizalayacağız.

* 1. **Hizalama İşlemi:**
* cv2.warpPerspective, homografi matrisini kullanarak test görüntüsünü dönüştürür (ölçekleme, döndürme ve kaydırma işlemlerini uygular).
* Bu işlemden sonra test görüntüsü, referans görüntüyle aynı boyut ve geometride olur.

## **Fark Hesaplama ve Kusur Tespiti:**

Hizaladığımız görüntüdeki kusurların nasıl bulunduğunu daha detaylı bir şekilde açıklayacağız.

1. **Piksel Bazında Fark Hesaplama:**

* cv2.absdiff, referans ve hizalanmış test görüntüsü arasındaki piksel farkını hesaplar.
* Bu, iki görüntü arasındaki farklı bölgeleri vurgular.

2. **Fark Görüntüsünde Eşikleme:**

* cv2.threshold, fark görüntüsündeki küçük farklılıkları yok etmek için kullanılır.
* Belirli bir eşik değerin üzerindeki pikseller kusur olarak kabul edilir ve binary görüntü oluşturulur (kusurlu pikseller = beyaz, diğerleri = siyah).

3. **Kontur Analizi ile Kusurların Belirlenmesi:**

* cv2.findContours, binary görüntü üzerindeki beyaz alanların kenarlarını algılar.
* Her konturun alanı ölçülür.
* Alanı 50 pikselden küçük olan kusurlar dikkate alınmaz; yalnızca daha büyük kusurlar değerlendirilir.

## KAYNAKÇA:

**OpenCV Documentation**

* URL: <https://docs.opencv.org/>

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

* URL: <https://github.com/opencv/opencv>

**GeeksforGeeks - Introduction to ORB (Oriented FAST and Rotated BRIEF)**

* URL: https://www.geeksforgeeks.org/orb-oriented-fast-and-rotated-brief/