CAMPUS BIRIGUI

ENTREGA - DESAFIO STANFORD

JOÃO EDUARDO SANTO FARIAS

PROF. DR. MURILO VARGES DA SILVA

SETEMBRO DE 2023

Objetivo: Implementar códigos que utilizam operações básicas combinando duas imagens.

Verificação de defeitos em placas: Basicamente realizando uma operação de subtração entre uma imagem de uma placa sem defeito com uma placa com defeito é possivel encontrar defeitos no processo de fabricação:

https://web.stanford.edu/class/ee368/Handouts/Lectures/Examples/3-Combining-Images/Defect_Detection/

Detecção de movimento: A partir de um vídeo, ao realizar a subtração do fundo da cena sem nenhuma pessoa é possível detectar movimentos:

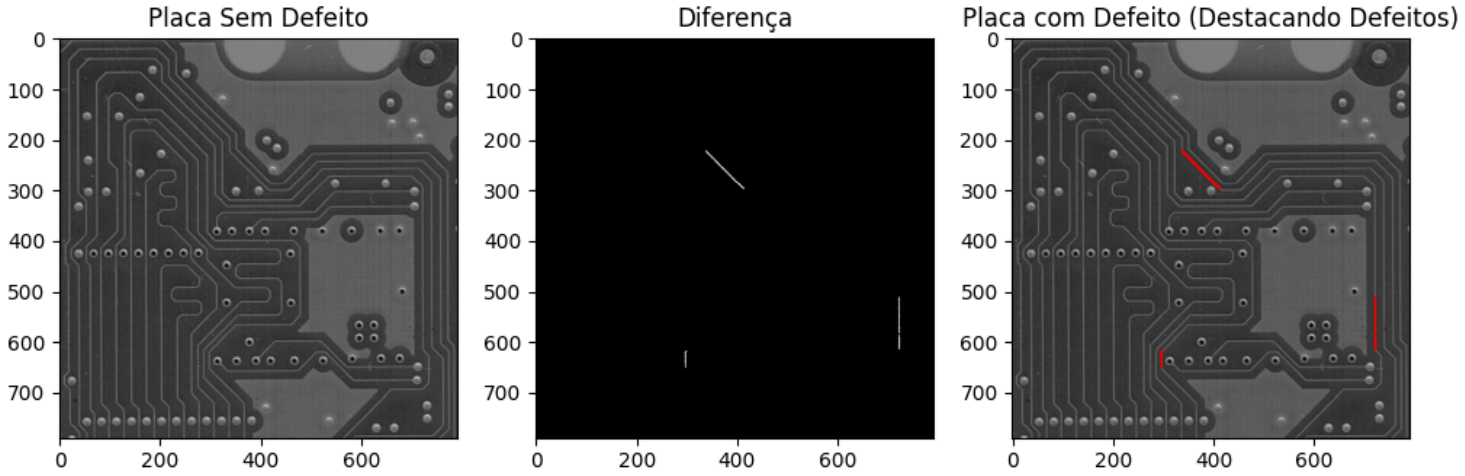
https://web.stanford.edu/class/ee368/Handouts/Lectures/Examples/3-Combining-Images/Background_Subtraction/

Verificação de defeitos em placas

A detecção de defeitos em placas é uma tarefa crucial no controle de qualidade de processos de fabricação. Para realizar essa verificação de forma eficaz, empregamos uma abordagem que envolve a comparação entre duas imagens: uma representando uma placa sem defeito e outra, uma placa com defeito. A essência desse método reside na realização de uma operação de subtração entre as duas imagens, permitindo-nos identificar discrepâncias que indicam potenciais defeitos.

Ao subtrair a imagem da placa sem defeito da imagem da placa com defeito, obtemos uma representação visual das diferenças entre elas. Essas diferenças podem incluir imperfeições, fissuras ou quaisquer outros desvios do padrão esperado. Para realçar essas discrepâncias, aplicamos um limiar, que destaca as regiões onde as diferenças são significativas.

Além disso, utilizamos a detecção de contornos para identificar e mapear áreas específicas que indicam a presença de defeitos. Isso nos permite não apenas encontrar defeitos, mas também quantificá-los e visualizá-los em destaque na imagem da placa com defeito, facilitando a análise e tomada de decisões no processo de controle de qualidade.



Essa abordagem eficiente e visualmente informativa oferece uma maneira confiável de identificar e avaliar defeitos em placas durante o processo de fabricação, contribuindo para a melhoria contínua da qualidade e a redução de resíduos.

Detecção de movimento

É possível detectar movimentos em vídeos usando Python através da subtração do plano de fundo da cena, identificando variações significativas nas imagens. Essa técnica é valiosa para diversas aplicações, como vigilância por vídeo, controle de qualidade e muito mais. O Python, com bibliotecas como OpenCV, oferece uma abordagem eficaz para a detecção de movimentos, destacando as regiões em que ocorrem mudanças, independentemente da presença de pessoas. Isso permite uma análise automatizada de vídeos em busca de eventos de interesse, contribuindo para uma variedade de aplicações práticas.

´Código placa

```
import cv2

import numpy as np

import matplotlib.pyplot as plt



# Carregue a imagem da placa sem defeito e a imagem da placa com
defeito

placa_sem_defeito = cv2.imread('pcb.png', cv2.IMREAD_GRAYSCALE)

placa_com_defeito = cv2.imread('pcb2.png', cv2.IMREAD_GRAYSCALE)



# Verifique se as imagens têm o mesmo tamanho

if placa_sem_defeito.shape == placa_com_defeito.shape:

    # Subtrai as duas imagens para encontrar defeitos

    diferenca = cv2.absdiff(placa_sem_defeito, placa_com_defeito)

    # Aplique um limiar para destacar os defeitos

    limite = 30

    _, imagem_limiarizada = cv2.threshold(diferenca, limite, 255,
cv2.THRESH_BINARY)



    # Encontre os contornos dos defeitos na imagem limiarizada

    contornos, _ = cv2.findContours(imagem_limiarizada,
cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)



    # Crie uma cópia colorida da placa com defeito para destacar os
defeitos em vermelho

    placa_com_defeito_colorida = cv2.cvtColor(placa_com_defeito,
cv2.COLOR_GRAY2BGR)
```

```
    # Desenhe os contornos dos defeitos na cópia colorida
    cv2.drawContours(placa_com_defeito_colorida, contornos, -1, (0, 0, 255), 2)

    # Exiba as imagens original, diferença e placa com defeito destacando os defeitos
    plt.figure(figsize=(12, 4))
    plt.subplot(131)
    plt.title("Placa Sem Defeito")
    plt.imshow(placa_sem_defeito, cmap='gray')

    plt.subplot(132)
    plt.title("Diferença")
    plt.imshow(imagem_limiarizada, cmap='gray')

    plt.subplot(133)
    plt.title("Placa com Defeito (Destacando Defeitos)")
    plt.imshow(cv2.cvtColor(placa_com_defeito_colorida, cv2.COLOR_BGR2RGB))

    plt.show()
else:
    print("As imagens têm tamanhos diferentes. Verifique suas imagens.")
```

Código vídeo

```
import cv2

import numpy as np



# Inicialize o objeto do vídeo a partir de um arquivo de vídeo ou uma
câmera

video = cv2.VideoCapture('output.avi')  # Substitua 'seu_video.mp4'
pelo caminho do seu vídeo



# Inicialize o objeto para o modelo de fundo

background_subtractor = cv2.createBackgroundSubtractorMOG2()



while True:

    # Leia o próximo frame do vídeo

    ret, frame = video.read()



    if not ret:

        break  # Encerra o loop quando o vídeo terminar



    # Aplique a subtração de fundo no frame atual

    foreground_mask = background_subtractor.apply(frame)



    # Aplique uma operação de limiarização para destacar as áreas de
movimento

    _, thresh = cv2.threshold(foreground_mask, 128, 255,
cv2.THRESH_BINARY)
```

```
    # Encontre os contornos das áreas destacadas
    contours, _ = cv2.findContours(thresh, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

    # Desenhe retângulos ao redor das áreas de movimento no frame
original
    for contour in contours:
        if cv2.contourArea(contour) > 200:  # Ajuste o valor do limiar
conforme necessário
            x, y, w, h = cv2.boundingRect(contour)
            cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0),
2)

    # Exiba o frame com as áreas de movimento destacadas
    cv2.imshow('Detecção de Movimento', frame)

    # Encerra o loop quando a tecla 'q' for pressionada
    if cv2.waitKey(20) & 0xFF == ord('q'):
        break

# Libere o objeto de vídeo e feche a janela
video.release()
cv2.destroyAllWindows()
```