UNIVERSIDADE FEDERAL DE SÃO CARLOS
DEPARTAMENTO DE COMPUTAÇÃO

Processamento Digital de Imagens

Transformada Distância

Eduardo Lopes Freire, 759025

Ingrid Lira dos Santos, 790888

Ivan Duarte Calvo, 790739

Vinícius Borges de Lima, 795316

São Carlos, SP, 2023

1 Resumo

Esse artigo tem como o tema demonstrar o funcionamento de uma transformada distância e o efeito da dilatação em uma imagem. Também será mostrado a diferença de resultado quando se é usado duas medidas de distâncias distintas, a euclidiana, e a city block (também conhecida como manhattan).

2 Introdução

A transformada distância é uma técnica de processamento de imagens muito utilizada em imagens binárias; para cada pixel do objeto, é calculada a distância entre ele e o pixel de borda mais próximo.

A utilização desse método é recomendado para: detecção de bordas, de forma a calcular a distância entre pixels vizinhos; reconhecimento de padrões, podendo calcular a distância de um conjunto de pixels de referência; e segmentação de imagens, de forma a segmentar objetos em primeiro plano em uma imagem calculando a distância de cada pixel para o objeto mais próximo.

3 Objetivo

O principal objetivo deste artigo será a implementação da transformada distância demonstrada em slide.

Como forma de aprimorar os conhecimentos obtidos em aula, serão utilizadas dois tipos de distâncias: euclidiana e city block. As quais as fórmulas matemáticas serão demonstradas mais abaixo neste documento.

Além de documentar o passo a passo da implementação, o artigo apresenta partes do código implementado, de forma a explicar de forma clara seu funcionamento.

4 Metodologia

A Transformada Distância é uma técnica de processamento de imagens que permite medir a distância de cada pixel de uma imagem para o pixel mais próximo que possui uma determinada característica.

Essa técnica é útil para a detecção de bordas, segmentação de objetos e reconhecimento de padrões em imagens. A transformada distância é baseada na criação de uma rede de células ao redor dos pixels de interesse, onde cada célula representa a região do espaço mais próxima de um determinado pixel.

A distância entre os pixels é calculada utilizando fórmulas matemáticas como a distância Euclidiana e a city block. Distância Euclidiana:

$$d_{12} = \sqrt{(p_1[0] - p_2[0])^2 + (p_1[1] - p_2[1])^2}$$

Distância City block:

$$d_{12} = |(p_1[0] - p_2[0])| + |(p_1[1] - p_2[1])|$$

Isso permite a criação de uma mapa de distância, que indica a distância de cada pixel em relação ao pixel mais próximo com a característica de interesse.

Nesse artigo, também foi usado a dilatação em imagens, que é uma técnica de processamento de imagens que serve para aumentar o tamanho de objetos ou regiões de interesse em uma imagem, sendo útil em várias aplicações, como reconstrução de imagens, remoção de ruídos e detecção de bordas, sendo também utilizado principalmente para preencher lacunas e buracos.

5 Implementação

Primeiramente para a implementação é necessário calcular a "imagem de borda externa", para isso foi realizada a operação de dilatação e depois a subtração da imagem "dilatada"pela imagem original, como mostra a seguinte fórmula:

$$I_b = I \oplus B - I$$

A implementação utilizada foi feita através do seguinte código:

```
1 def dilation(img, elem_est, origin):
2
3     num_rows, num_cols = img.shape
4     num_rows_ee, num_cols_ee = elem_est.shape
5
```

```
    set_ee = []
    for row in range(num_rows_ee):
        for col in range(num_cols_ee):
            if elem_est[row, col]==1:
                set_ee.append((origin[0]-row, origin[1]-col))

    img_res = np.zeros_like(img)
    for row in range(num_rows):
        for col in range(num_cols):
            z = (row, col)
            has_intersect = intersects(img, set_ee, z)
            if has_intersect:
                img_res[row, col] = 1

    return img_res

def intersects(img, set_ee, z):

    for point in set_ee:
        trans_point = (point[0]+z[0], point[1]+z[1])
        if trans_point[0]>=0 and trans_point[0]<img.shape[0] and \
           trans_point[1]>=0 and trans_point[1]<img.shape[1]:

            if img[trans_point]==1:
                return True
    return False

elem_est = np.array([[1, 1, 1],
                     [1, 1, 1],
                     [1, 1, 1]])

img_res = dilation(img, elem_est, origin=(1, 1))
```

É interessante notar, na linha 33, que foi utilizado um elemento estruturante de tamanho 3x3 contendo o valor 1 em todos os *pixels*.

Após a criação da "imagem de borda externa"foram calculadas as distâncias através das seguintes funções:

```
def calc_dist(x, y, img_borda, dist_type):
    n_rows, n_cols = img_borda.shape
```

```
    dists = []
    for row in range(n_rows):
        for col in range(n_cols):
            if img_borda[row][col]:
                if dist_type == 'euclidiana':
                    dists.append(( (x-row)**2 + (y-col)**2 )**0.5)
                elif dist_type == 'city':
                    dists.append(( abs(x-row) + abs(y-col)))

    return dists

def distancia(img, img_borda_ext, dist_type):
    n_rows, n_cols = img.shape
    img_dist = [[0.0 for j in range(len(img[0]))] for i in range(len(img
    ))]

    for row in range(n_rows):
        for col in range(n_cols):
            img_dist[row][col] = min(calc_dist(row, col, img_borda_ext,
    dist_type))

    return img_dist
```

A função *calc_dist()* percorre todos os pixels da imagem de borda externa, calculando em cada um deles a distância entre o pixel da imagem original. A função *distancia()* percorre todos os pixels da imagem para cada um deles escolhe o menor valor no vetor de distâncias retornado pela função *calc_dist()*.

6 Resultados

Para analisar as aplicações da transformada distância foram utilizadas três imagens com características diferentes.

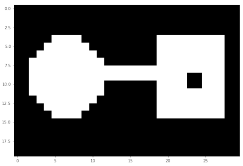

Figura 1: Imagem com diversos objetos

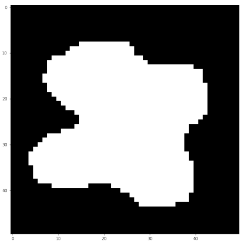

Figura 2: Imagem com diversas curvas e grande área

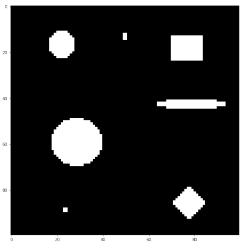

Figura 3: Imagem com um furo

Para o processamento das imagens, primeiro foi utilizada a operação da dilatação para suavizá-las e obter suas bordas externas. Os resultados obtidos foram os seguintes:

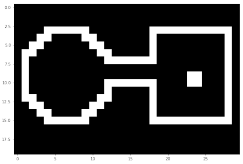

Figura 4: Borda externa de diversos objetos

Figura 5: Borda externa com diversas curvas e grande área

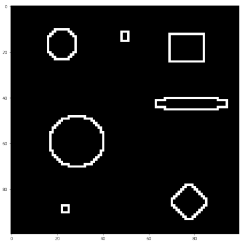

Figura 6: Borda externa com furo

Após isso, foi aplicada sobre as imagens das bordas a transformada utilizando as distâncias euclidiana e city block, gerando os seguintes resultados, apresentando primeiro a imagem original seguida da ordem mencionada acima.

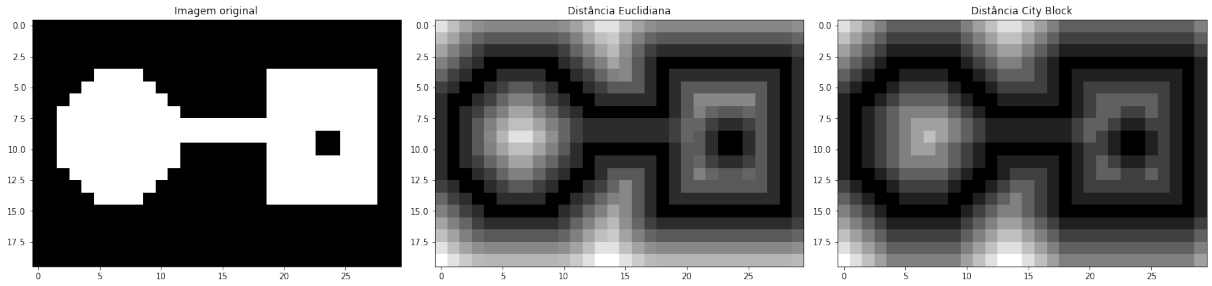

Figura 7: Resultado com diversos objetos

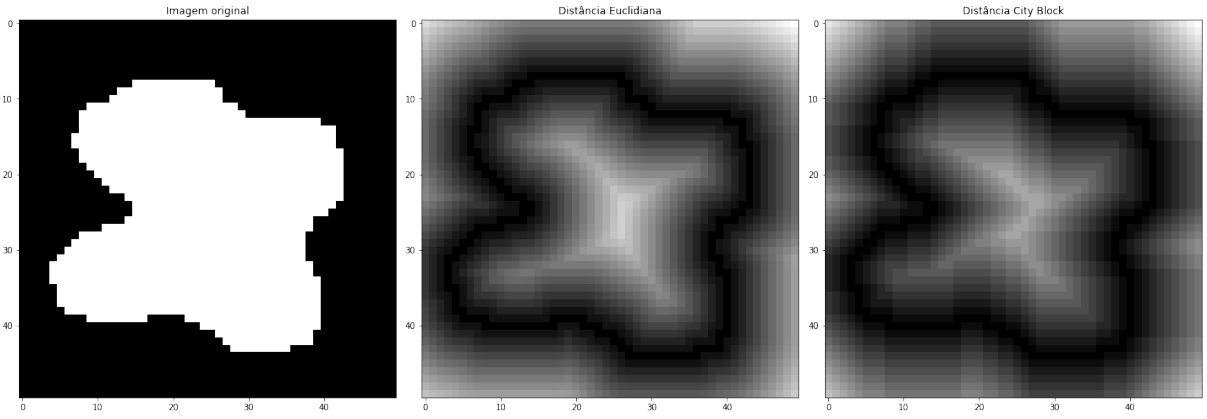

Figura 8: Resultado com diversas curvas e grande área

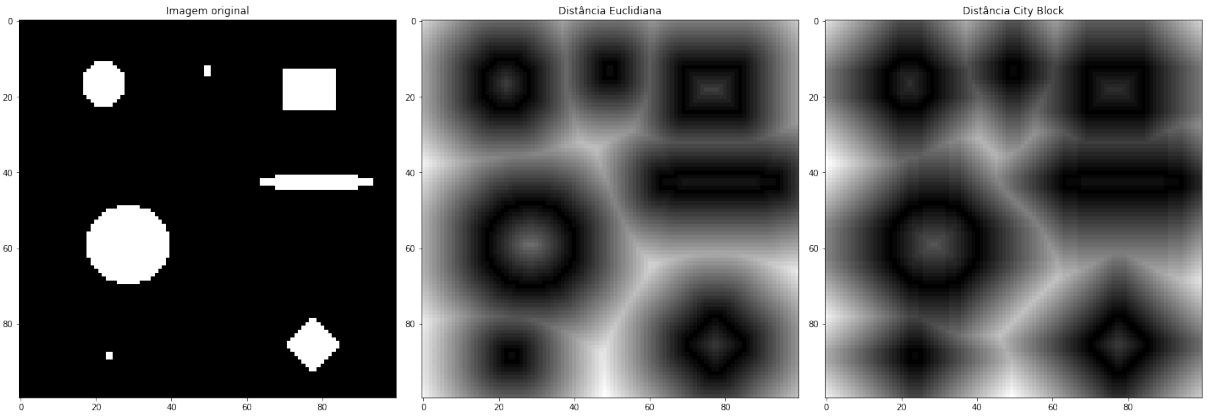

Figura 9: Resultado com furo

É possível observar na Figura 7 que a transformada funciona bem para a segmentação de objetos com qualquer distância utilizada, apesar de apresentar, na distância city blocks, uma presença maior de artefatos na forma predominante de listras ao redor das bordas. Sabendo que a transformada distância aplicada desta forma representa as cores brancas como regiões mais próximas do objeto (no geral como um todo ou seja, seu centro, mas isso será discutido mais a fundo na Figura 9) e as cores escuras como regiões mais distantes, podemos distinguir de forma clara a forma e o número de objetos presentes na imagem, apesar de objetos pequenos não terem tamanho ou distância o suficiente para gerar um pixel branco. É possível observar também nessa figura a presença de artefatos no exterior dos objetos na forma de pixels brancos, causada pelo fato dos pixels externos terem sido "empurrados" para longe do objeto, de certa forma, no cálculo do valor das distâncias,

consequência esta que se mostra útil na aplicação de segmentação de objetos.

Com os resultados obtidos na Figura 8, devido ao seu tamanho, podemos observar uma característica marcante da transformada distância que é a presença de uma representação em forma de "raiz" do interior do objeto das maiores distâncias até o seu contorno, informação esta que pode ser utilizada para reconstruir a imagem a partir de suas distâncias. Outro detalhe que podemos notar é uma maior suavização na borda do objeto ao utilizar a distância city block, o que mostra que, apesar de ser um cálculo mais fácil, representa pior o objeto original se comparado à distância euclidiana.

Pode-se notar na Figura 9 que, apesar de ser um único objeto contendo dois centros distintos com bastante "massa", a presença do furo no centro à direita garantiu que, devido à diferença das distâncias em relação ao objeto como um todo, não houvesse um pixel branco, presente no centro à esquerda.

7 Conclusão

Pode-se notar, portanto, que a transformada distância é uma ferramenta muito poderosa no processamento de imagens, sendo capaz de realizar diversas tarefas com cálculos simples, o que permite que seja utilizada em conjunto com outras operações. Foi possível notar no decorrer do trabalho que a transformada, tanto com a distância euclidiana quanto city block, foi capaz de, com êxito, segmentar objetos e delimitar bordas de modo a ser capaz de reverter à imagem original enquanto mantém as características internas dos objetos.