UNIVERSIDADE FEEVALE

Afinamento de Zhang-Suen

Josué, Lucas, Murilo, Thauan, Thais e Ricardo

Processamento Digital de Imagens 2020/02 4N

Profª Marta Rosecler Bez

Novo Hamburgo - RS

2020

Introdução

- **Afinamento é uma das técnicas presentes no conceito de esqueletização.**
- **Em processamento de imagens, a esqueletização é utilizada para obter o esqueleto de um objeto.**
- **O afinamento é a redução de uma forma para uma versão simplificada da mesma (esqueleto), mas que ainda mantém suas características originais.**
- **Os esqueletos possuem várias aplicações na área de processamento de imagens, tais como, agrupamento, segmentação, vetorização, descrição de formas, reconhecimento de caracteres, inspeção, etc.**

Regras do Afinamento

- **O afinamento consiste em verificar sucessivamente as bordas dos objetos, onde cada ponto da borda será ou atribuído ao fundo ou ao esqueleto.**

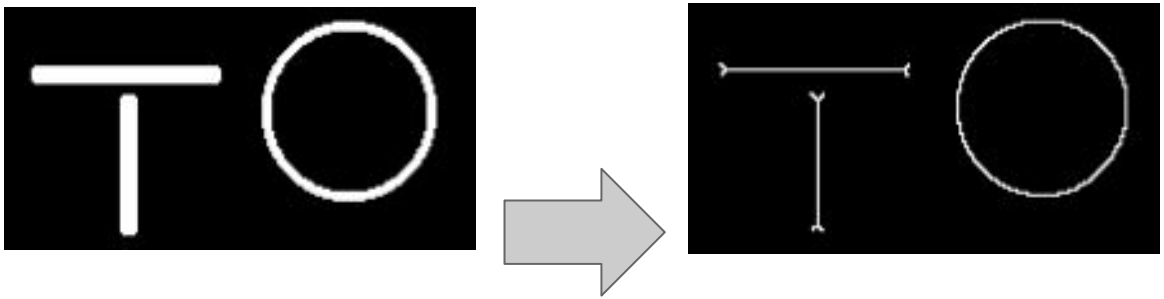
- **Geralmente o afinamento mantém um objeto conexo.**

- **Um conjunto de regras deve ser seguido para que o afinamento seja efetivo, sendo elas:**
  1. Remover pontos extremos;

  2. Quebrar a conectividade;

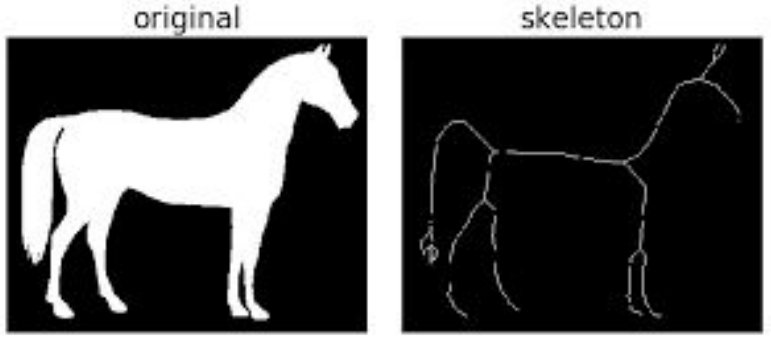
  3. Causar a erosão excessiva da região

Aplicações do Afinamento



Aplicação de afinamento em uma imagem

Aplicação de afinamento pode ser usado em processamento de impressões digitais



Aplicação de afinamento com deficiências

Algoritmo de Zhang Suen

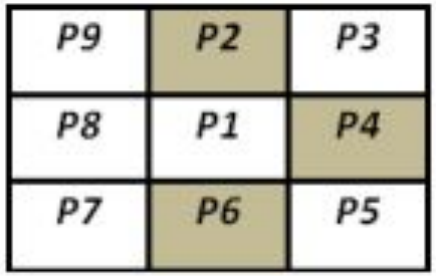
O algoritmo de Zhang Suen consiste na aplicação sucessiva de duas etapas. Na primeira etapa um pixel é eliminado, ou marcado para ser eliminado em outro momento se as seguintes condições se cumprirem:

1. **O número de conectividade é 1;**

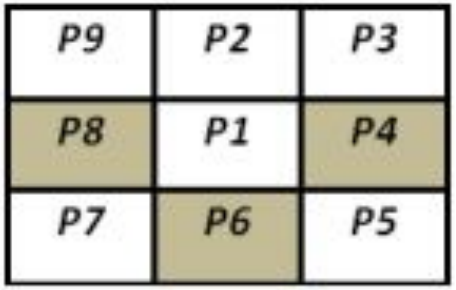
2. **Existem ao menos dois pixels vizinhos pretos, e menos do que sete, sendo que, vizinhos pretos de um ponto são todos os pontos pretos presentes nas 8 direções possíveis a partir do ponto central (pixels de P2 a P9);**

P9	P2	P3
P8	P1	P4
P7	P6	P5

3. Ao menos um dos pixels P2, P4 ou P6 são brancos;



4. Ao menos um dos pixels P4, P6 ou P8 são brancos:



Na segunda etapa, é aplicado outro conjunto de regras, que se o pixel se enquadrar nas condições, ele é eliminado.

A regra 1 e 2 da primeira iteração são mantidas, e somente muda a regra 3 e 4.

3. Ao menos um dos pixels P2, P4 ou P8 são brancos:

P9	P2	P3
P8	P1	P4
P7	P6	P5

4. Ao menos um dos pixels P2, P6 ou P8 são brancos:

P9	P2	P3
P8	P1	P4
P7	P6	P5

O algoritmo se encerra quando nenhum pixel se enquadrar nas condições, o resultado esperado segue na figura abaixo:

(a) Imagem original

(b) Imagem afinada

Desenvolvimento

A implementação do algoritmo foi feita em Python 3 com orientação a objetos;

Foi definida uma interface gráfica utilizando os recursos da biblioteca Tinker;

A interpretação e processamento das imagens foram feitos com a biblioteca PIL (Pillow).

O projeto possui uma classe principal chamada “Afinamento”;

Dentro desta classe, é feita configuração da interface, seleção de arquivos e manipulação de imagem;

Entre os métodos de manipulação, temos:

Binarização da imagem:

```
def binaria(self, img, meio):
    old = img.load()
    new = PIL.Image.new('1', (img.width, img.height))
    pixel = new.load()
    for x in range(img.width):
        for y in range(img.height):
            media = round((old[x, y][0] + old[x, y][1] + old[x, y][2]) / 3)
            pixel[x, y] = PIXEL_BRANCO if media > meio else PIXEL_PRETO
    return new
```

Descobre os 8 vizinhos de um pixel.

```
def vizinhos(self, x, y, img):
    p2 = img[x, y - 1]
    p3 = img[x + 1, y - 1]
    p4 = img[x + 1, y]
    p5 = img[x + 1, y + 1]
    p6 = img[x, y + 1]
    p7 = img[x - 1, y + 1]
    p8 = img[x - 1, y]
    p9 = img[x - 1, y - 1]
    return [p2, p3, p4, p5, p6, p7, p8, p9]
```

Descobre os vizinhos pretos.

```
def gtdeVizinhosPretos(self, vizinhos):
    result = 0
    for v in vizinhos:
        if v == PIXEL_PRETO:
            result += 1
    return result
```

Estabelece o número de conectividade, onde cada transição de branco para preto é quantificada

```
def conectividade(self, vizinhos):
    result = 0
    v = vizinhos + vizinhos[0:1]  # P2, ... P9, P2
    for i in range(len(vizinhos)):
        result = (result + 1) if (v[i] == PIXEL_BRANCO and v[i+1] == PIXEL_PRETO) else result
    return result
```

O método zhangSuen implementa o afinamento de fato, na imagem abaixo mostra a aplicação da primeira iteração do algoritmo.

```
def zhangSuen(self):
    width = self.img.width
    height = self.img.height
    new = self.binaria(self.img, 123)
    img = new.load()

    iteracao1 = iteracao2 = [(-1, -1)]
    while iteracao1 or iteracao2:
        iteracao1 = []
        for x in range(1, width - 1):
            for y in range(1, height - 1):
                if img[x, y] == 0:
                    vizinhos = P2, P3, P4, P5, P6, P7, P8, P9 = self.vizinhos(x, y, img)
                    if (self.conectividade(vizinhos) == 1  # 1
                        and 2 <= self.qtdeVizinhosPretos(vizinhos) < 7  # 2
                        and (P2 == PIXEL_BRANCO or P4 == PIXEL_BRANCO or P6 == PIXEL_BRANCO)  # 3
                        and (P4 == PIXEL_BRANCO or P6 == PIXEL_BRANCO or P8 == PIXEL_BRANCO)  # 4
                    ):
                        iteracao1.append((x, y))
        for x, y in iteracao1:
            img[x, y] = 1
```

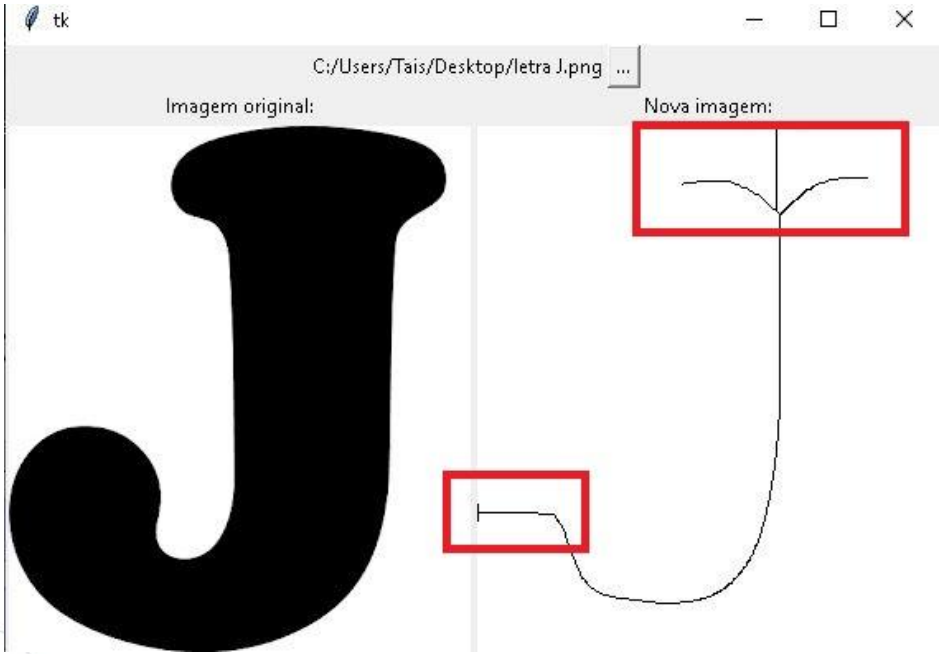
Aplicação da segunda iteração de remoção de pixel do método de Zhang Suen

```
    iteracao2 = []
    for x in range(1, width - 1):
        for y in range(1, height - 1):
            if img[x, y] == 0:
                vizinhos = P2, P3, P4, P5, P6, P7, P8, P9 = self.vizinhos(x, y, img)
                if(self.conectividade(vizinhos) == 1  # 1
                    and 2 <= self.qtdeVizinhosPretos(vizinhos) <= 6  # 2
                    and (P2 == PIXEL_BRANCO or P4 == PIXEL_BRANCO or P8 == PIXEL_BRANCO)  # 3
                    and (P2 == PIXEL_BRANCO or P6 == PIXEL_BRANCO or P8 == PIXEL_BRANCO)  # 4
                ):
                    iteracao2.append((x, y))
    for x, y in iteracao2:
        img[x, y] = 1
self.novaImg = new.convert('RGB')
self.photoNew = ImageTk.PhotoImage(self.novaImg)
self.imgEditada['image'] = self.photoNew
```

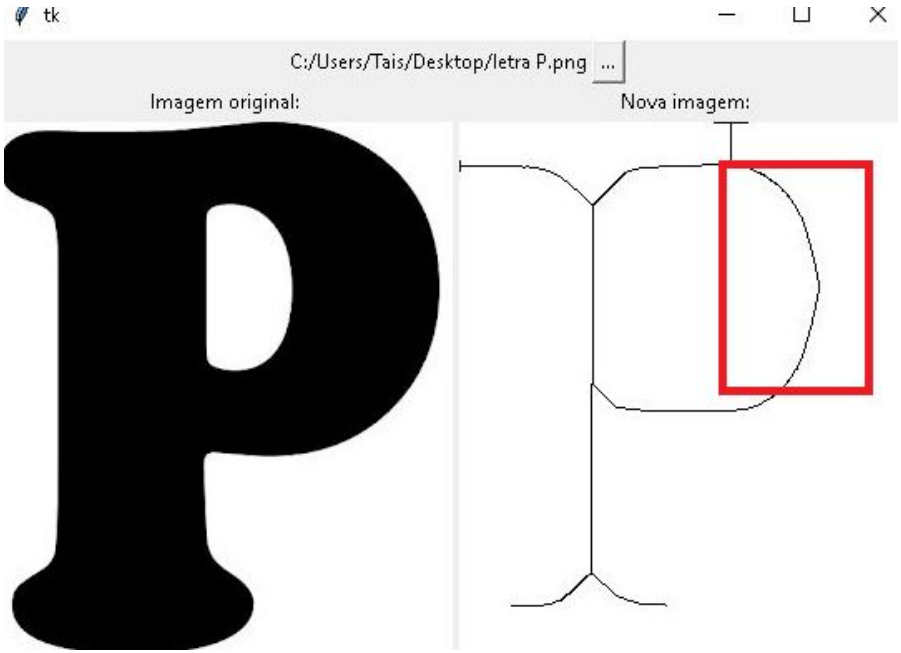
Conclusão

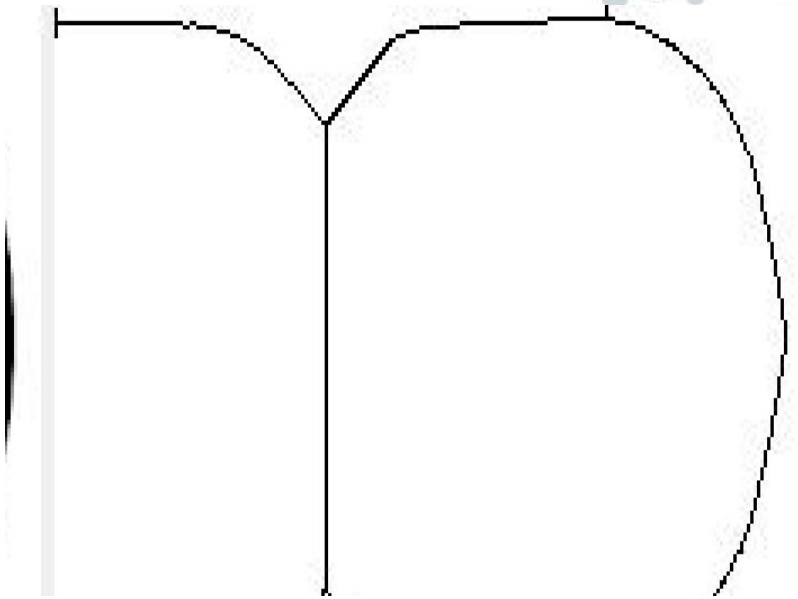
A aplicação do algoritmo de Zhang Suen pode ser um pouco mais demorada em função das iterações pixel a pixel.

Em algumas situações apresenta distorções em imagens que foram aplicadas o afinamento;

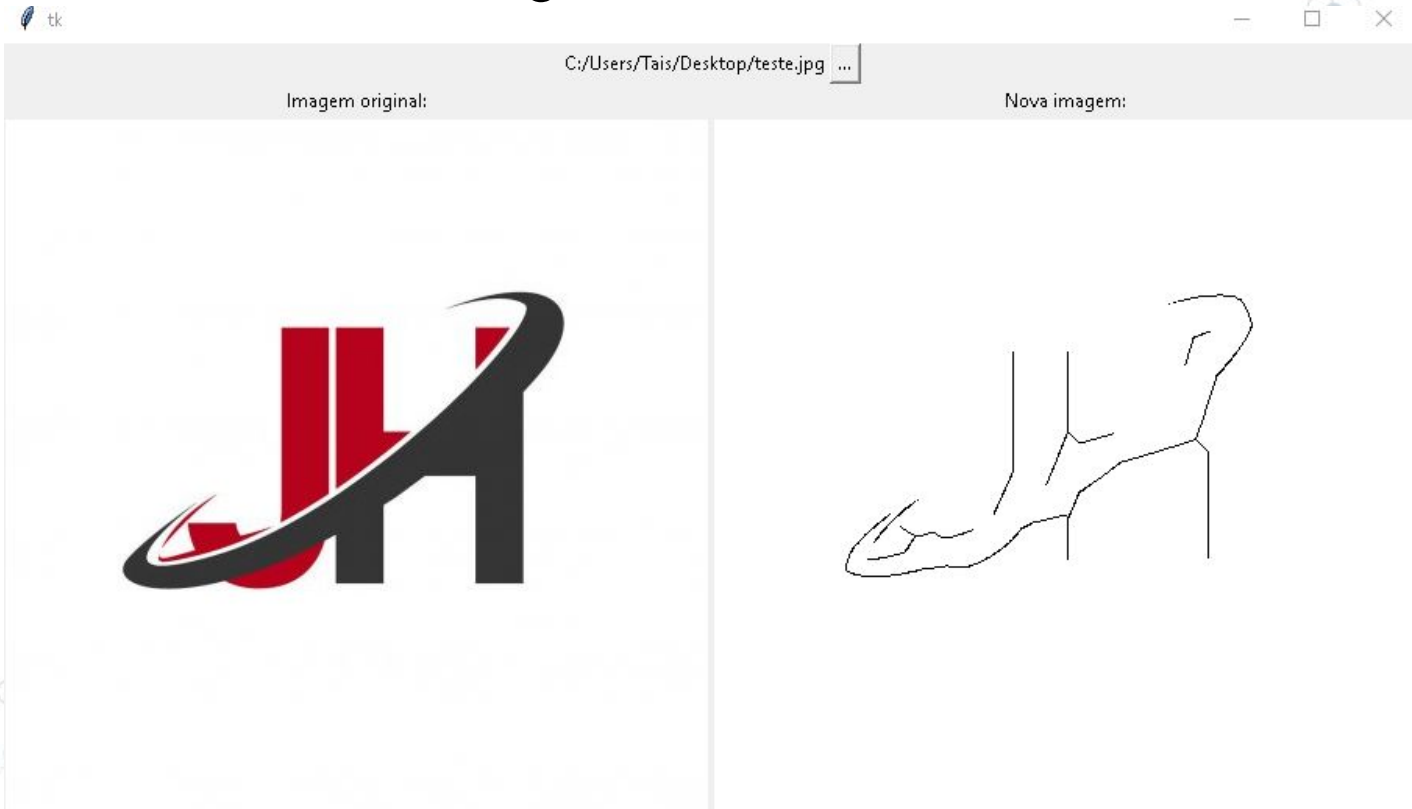


Em alguns exemplos de imagens com arredondamento, o afinamento apresenta o efeito “staircase”:

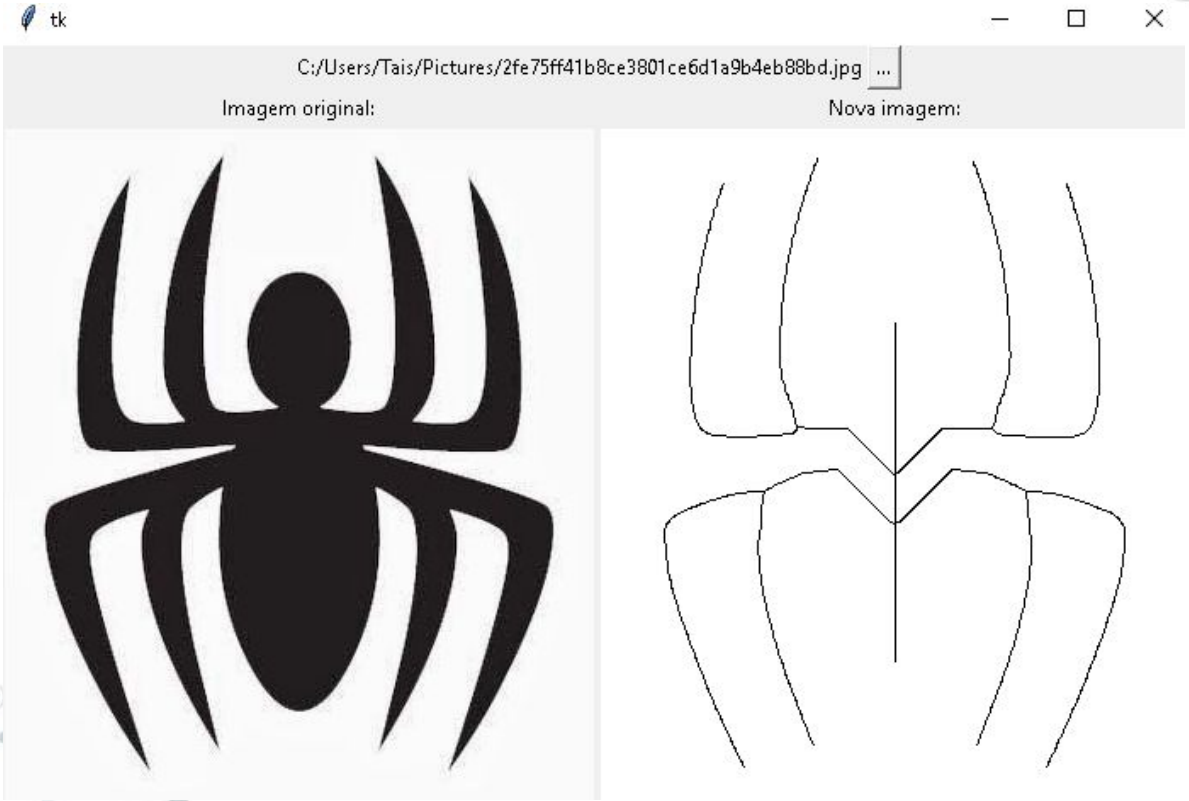




Algumas imagens apresentam uma grande perda de características morfológicas:



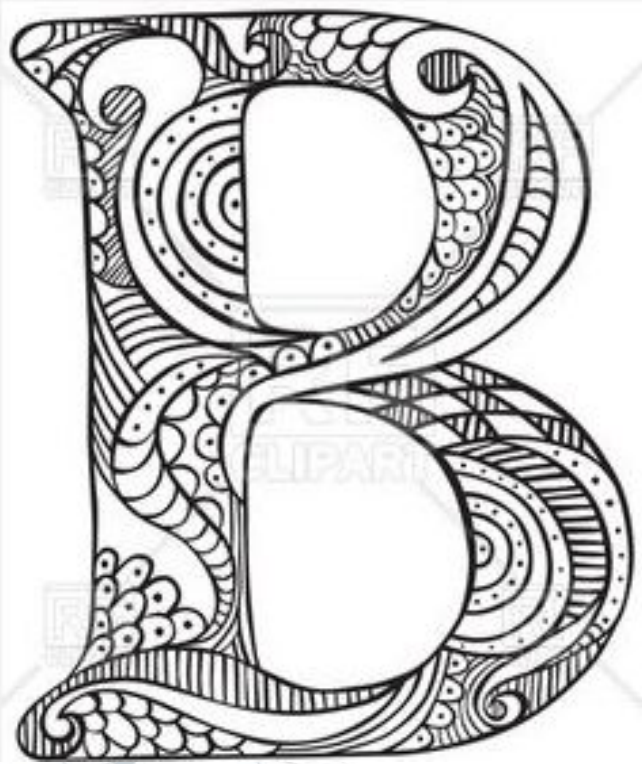
Embora tenha apresentado limitações em algumas situações, existem casos em que o esqueleto apresentou ótima qualidade:



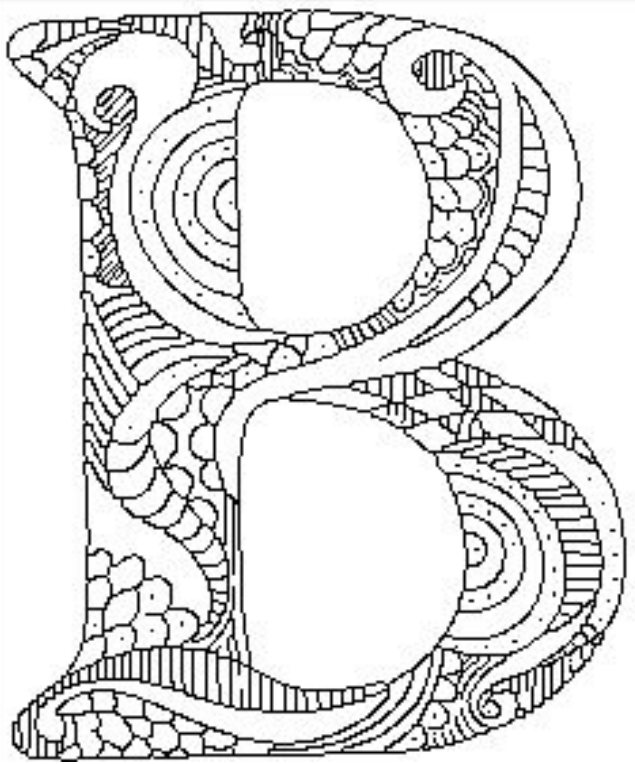
tk

C:/Users/Tais/Desktop/teste B.jpg ...

Imagem original:



Nova imagem:



Referências Bibliográficas

Galvanin, E. A. S., Vale, G. M., Dal Poz, A. P e Telles, S. S. S. **DETECÇÃO E AFINAMENTO DE BORDAS UTILIZANDO SUAVIZAÇÃO ANISOTRÓPICA E ESQUELETIZAÇÃO,** IV Colóquio Brasileiro de Ciências Geodésicas - IV CBCG, Curitiba, 16 a 20 de maio de 2005.

R. O. Plotze, e O. M. Bruno, **Estudo e comparação de algoritmos de esqueletonização para imagens binárias**, IV Congresso Brasileiro de Computação – CBComp 2004.

CORRÊA, Fernando Porto, FESTA, Leidmar Magnus, **Avaliação de técnicas para afinamento de imagens digitais**, Curitiba, 2005, Universidade Federal do Paraná;

GUILHERME, Luis Renato Woiski, **Uma abordagem de afinamento por aprendizagem através de exemplos**, Curitiba 2007, a Pontifíicia Universidade Católica do Paraná;

PRATES, Jorge Marques, **Algoritmo de Thinning e suas aplicações**, Presidente Prudente 2011, Universidade Estadual Paulista Júlio de Mesquita Filho.