Introdução ao Processamento de Imagens

Victória Goularte - 12/0137691

***Resumo*—No trabalho são aplicadas operações consideradas de médio nível no processamento digital de imagens. São as operações morfológicas, de segmanetação e codificação de vídeo.**

I. Introdução

FACE SWAPPING é uma técnica de processamento de imagens que digitalmente envolve troca de rostos de dois ou mais sujeitos retratados em uma determinada fotografia. Uma variação conhecida da prática é facebombing , uma técnica similar que envolve tomar uma face em um grupo e aplicá-lo a todas as faces da foto. A prática aumentou radicalmente em popularidade em 2015, quando vários aplicativos automatizados foram criados para trocar instantaneamente os rostos na fotografia e vídeo.

A origem exata da troca de rosto é desconhecido, mas sua popularidade como um método de criação de imagens exploráveis remonta ao início do Photoshop.

II. Metodologia

O trabalho foi dividido em algumas partes, que serão tratadas separadamente a seguir:

Parte I

Primeiramente foram lidas duas imagens. Uma que será retirada as regiões de interesse como: olhos, nariz e boca; e outra que servirá de rosto base para a inserção dessas regiões.

As imagem foram redimensionadas para 250x250 pixels para que pussuam as mesmas dimenções.

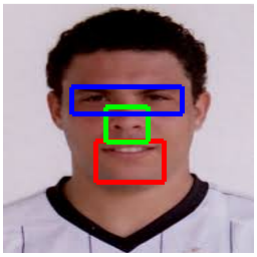
(a) Imagem base

(b) Imagem recorte

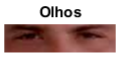
Parte II

Logo em seguida, foram detectadas as regiões de interesse (olhos, nariz e boca) e recortadas da imagem.

Utilizando o MATLAB, um programa foi feito para que a imagem acima apresentada fosse modificada e resultasse em uma imagem binária com os objetos que a compõem pretos e fundo branco. Para isso, antes de qualquer coisa, a imagem foi binarizada e percebido que em certos pontos



(c) Imagem detecção

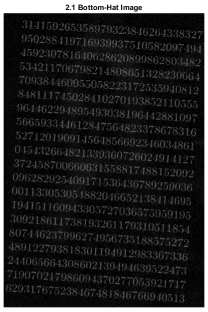


(d)

(e)

(f)

perdia-se informações, pois em alguns lugares o fundo e objetos estavam classificados em um mesmo nível de cinza fazendo com que objetos também se tornassem fundo. Então, fez-se a transformada bottom-hat que é é utilizada para realçar objetos escuros sobre fundo claro utilizando uma função do próprio MATLAB e alternativamente aplicando o fechamento na imagem obtendo apenas o seu fundo e depois subtraindo a fundo da imagem original, que também caracteriza uma operação bottom-hat, e os resultados obtidos foram:



(g) Imagem Bottom-Hat_int

(h) Fundo - Imagem

Nota-se que o objeto se destaca melhor na bottom-hat feito obtendo-se o fundo e posteriormente subtraindo esse fundo da imagem origina. Então, sobre essa imagem foi aplicada a operção de identidade inversa para que os objetos ficassem pretos e fundo branco.

Essa imagem foi binarizada para então destacar os objetos.

Foram então percebidos falhas de preenchimento dos dígitos e certos ruídos na imagem, e para que melhorasse essa imagem foram aplicados outra operação morfológica

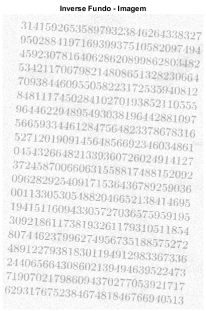


Figura 1. Inverse bottom-hat

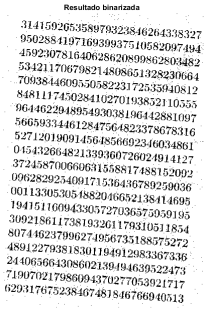


Figura 2. Bottom-hat binarizada

de dilatação para completar os objetos em que havia falhas e um filtro de média para retirada dos ruídos. Por fim, ainda foi feito um fechamento para separar dígitos que se emendaram na dilatação

Figura 3. Resultado

A. Parte II

Na segunda parte, foram seguidos os passos solicitados e os resultados obtidos para tratar a imagem abaixo serão descritos a seguir

2.1: A imagem foi binarizada como na primeira parte, onde as células são pretas e o fundo é branco

2.2: A função *bwareopen* para preencher espaços desconectados

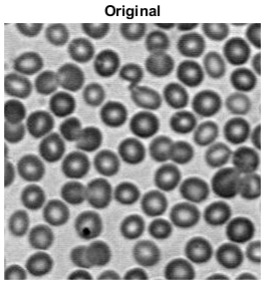


Figura 4. Original

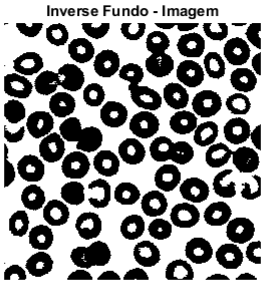


Figura 5. Imagem binarizada

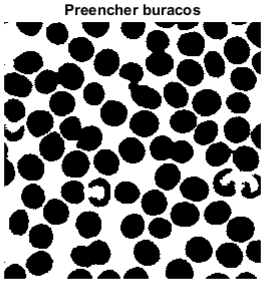


Figura 6. Imagem Buracos Preenchidos

2.3: A distância foi calculada através da função *bwdist* usando o complemento da imagem

2.4: Por fim, essa imagem foi segmentada, a fim de dividir os objetos que a compõe

Nota-se que a imagem não foi perfeitamente segmentada, ou seja, os objetos não foram todos perfeitamente divididos, já que na binarização algumas células se mantiveram unidas, mesmo aplicando outras operações morfológicas para separá-las.

B. Parte III

Na terceira e última parte, foi feita uma função ler_yuv que recebe como parâmetros um arquivo YUV, sua reso-

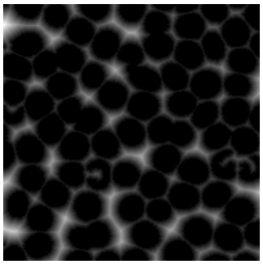


Figura 7. Distância

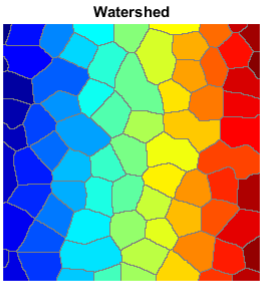


Figura 8. Distância

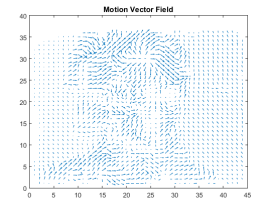
lução, o formato (4:2:0) e o número do quadro a ser lido, e seu retorno é a imagem desse quadro.

A seguir são feitas novas funções que estimam o movimento (DPCM) entre um quadro e outro, recuperados a partir da função já citada. Essas funções foram feitas a partir do algoritmo de Block Matching para estimação de movimento.

As funções implementadas no projeto foram:

- LogSearch;
- Motion_Est;
- reconstruct;
- FullSearch;
- Bidirectional_ME.

Resultados obtidos com blocos de tamanho 8x8 a partir dos frames 100 e 150 do arquivo '*foreman.yuv*':

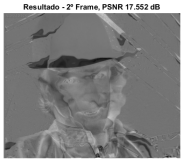


(a) Vetores de movimento

(b) 1º Frame

(c) 2º Frame

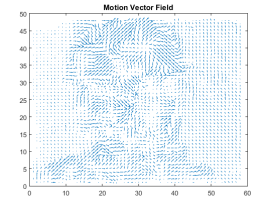
(d) Resultado



(e) SPRN

Resultados obtidos com blocos de tamanho 4x4 a partir dos frames 100 e 150 do arquivo '*foreman.yuv*':

É notório como quanto menor o bloco, melhor a estimativa do movimento.

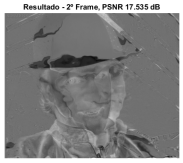


(f) Vetores de movimento

(g) 1º Frame

(h) 2º Frame

(i) Resultado



(j) SPRN

III. Conclusão

A partir dos resultados obtidos, na primeira parte nota-se que a definição da morfologia bottom-hat, que destaca objetos escuros sobre um fundo claro, aplicando fechamento na imagem para obter o fundo e posteriormente subtraindo a imagem por esse fundo encontrado é afirmada. Na segunda parte, aplicando segmentação seguindo os passos instruídos, tem-se o subdivisão da imagem em objetos ou regiões como era esperado, podendo servir para diversas aplicações que necessitam dos objetos isolados. E, por fim, um vídeo YUV é lido a partir dos parâmetros solicitados, e são recuperados frames especificos nessa função e aplica-se o algoritmo de Block Matching para estimação do movimento que foi claramente aplicado.

Referências

[1] http://scholar.harvard.edu/stanleychan/software/subpixel-motion-estimation-without-interpolation
Materiais da disciplina