

Image Inpainting OpenCV - Implementação em OpenCV de Image Inpainting

Nuno Miguel Soares Silva [72708]

Pedro Marques Ferreira da Silva [72645]

Resumo –O trabalho proposto para o projecto da unidade curricular de Computação Visual consiste no desenvolvimento de uma aplicação Image Inpaiting que é muitas vezes usada nas técnicas de restauração/recuperação de imagens que permitem que uma ou várias regiões da imagem sejam preenchidas de forma a preservar a coerência visual das cenas de fundo. Esta técnica também está historicamente associada a técnicas de restauro de imagens fotográficas e ainda em programas recentes de tratamento de imagem como o Adobe Photoshop, etc...

Para este efeito foi criado um projeto em C++ implementando também as funções das bibliotecas providenciadas pelo OpenCV.

Numa primeira fase pensou-se em criar uma interface user-friendly mas isto mostrou-se ser complicado com a configuração das bibliotecas do OpenCV e por isso para simplificar o trabalho optou-se por utilizar tanto a nível de opções como na própria implementação do OpenCV uma interface com o utilizador com um carácter mais basico. Este relatório vai se focar principalmente na descrição da arquitectura e do código que foi implementado.

Este relatório também vai apresentar passo-a-passo todas as decisões tomadas no desenvolvimento desta aplicação assim como um esclarecimento do que foi usado na implementação.

I IMPLEMENTAÇÃO

O código base utilizado neste projecto foi o código do exemplo demo inpainting.cpp presente na pasta cpp dos samples da pasta sources do OpenCV 2.4.9. Para a implementação deste projecto foi usado o IDE Codeblocks e o compilador do MinGw; Inicialmente optou-se por reproduzir imagens usando a máscara que o utilizador vai produzindo com o rato tendo assim o flooding da imagem em que os pixéis seleccionados pela máscara são então substituídos pelos pixéis presentes na sua vizinhança. Isto torna-se fundamental em certas aplicações como o restauro de imagens e vídeo em que temos de fazer recuperações de regiões rasuradas ou recuperações de "bad" frames em sequências de vídeo ou mesmo até quando temos de fazer a eliminação/extracção de objectos visualmente oclusivos em cenas de interesse do utilizador; Para correr este projecto basta ir à pasta Debug da pasta bin do projecto e correr o comando *my_inpaint*;

II OPENCV

As bibliotecas do OpenCV que foram utilizadas(*ver referências*) foram as bibliotecas *core.hpp* que contêm o header de todas as bibliotecas; a biblioteca *highgui.hpp* que chama as funções de estrutura como a *imread()*, *imshow()* e *namedWindow()* assim como

event flags como *CV_EVENT_FLAG_LBUTTON*, *CV_EVENT_MOUSEMOVE*, etc...;

A biblioteca *imgproc.hpp* que fica encarregue do tratamento das cores da imagem usando para isso enumerations tais como *ColorConversionCodes(cv::COLOR_RGB2RGBA)* ou *cv::COLOR_GRAY2RGBA*) e finalmente a biblioteca *photo.hpp* para chamar a função principal deste projeto: *inpainting*(*).*

Esta função, dependendo do comando que o utilizador submeter no terminal, tem como argumentos a imagem original(*source_img*) caso seja o utilizador a fazer o inpainting ou a imagem original e a máscara de inpainting(*masked_img*, em que no final mostramos estas imagens juntamente com a imagem resultado(*inpainting_img*) em que é possível ver o *radius* do inpainting assim como as fases presentes no método de inpainting. O nosso programa foi definido para ter um *radius* de 3px e foi utilizado o método de Alexandru Telea (*CV_INPAINT_TELEA*) que é baseado no algoritmo de *floodfilling* em que temos o preenchimento por "inundação" dos pixéis a partir dos pixéis das regiões vizinhas. Para além deste método também existe o método de Navier-Strokes(*CV_INPAINT_NS*) que utiliza algoritmos de resolução de imagem baseados em equações diferenciais com derivadas parciais(*ver referências*).

III INTERFACE

Como já foi dito anteriormente, implementar uma interface user-friendly acabou por se tornar uma tarefa com alguma dificuldade. No entanto foi implementado uma interface ao nível duma consola que mostra as *Hot-keys* que o utilizador usa no programa.

```

2lena.jpg
OpenCV Project
Image Inpainting
Made by: Pedro Silva 72645; Nuno Silva 72708;

Cool inpainting module. Inpainting repairs damage to images by floodfilling the
damage with surrounding image areas.
Using OpenCV version 2.4.9
Usage:
    my_inpaint [image_name -- Default lena.jpg]

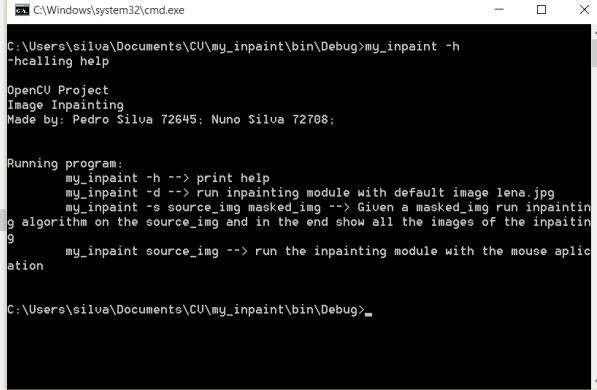
Hot keys:
    ESC - quit the program
    r - restore the original image
    i or SPACE - run inpainting algorithm
        (before running it, paint something on the image)

```

Fig. 1: Interface numa consola/terminal

Esta janela abre quando o programa arranca. Como podemos ver na figura, além do título do projeto e dos créditos dos autores, o utilizador tem à sua disposição as operações que pode efetuar com o teclado. Ao premir a tecla *i* ou a

barra de espaços faz-se o inpainting da imagem, com tecla *r* reinicia-se a máscara aplicada e consequentemente faz refresh à imagem, e com a tecla *esc* o programa fecha. O utilizador também dispõe do comando *'-h'* para mostrar um texto de ajuda.



```
C:\Windows\system32\cmd.exe
C:\Users\silva\Documents\CU\my_inpaint\bin\Debug>my_inpaint -h
-h calling help
OpenCU Project
Image Inpainting
Made by: Pedro Silva 72645; Nuno Silva 72708;

Running program:
    my_inpaint -h --> print help
    my_inpaint -d --> run inpainting module with default image lena.jpg
    my_inpaint -s source_img masked_img --> Given a masked_img run inpainting
g algorithm on the source_img and in the end show all the images of the inpaitin
g
    my_inpaint source_img --> run the inpainting module with the mouse applica
tion

C:\Users\silva\Documents\CU\my_inpaint\bin\Debug>~
```

Fig. 2: Texto de ajuda numa consola-terminal

Este texto indica ao utilizador como deve proceder para inicializar o programa. Alternativamente, o utilizador também dispõe do comando *'-d'* em que pode fazer o inpainting com a imagem default *lena.jpg*.

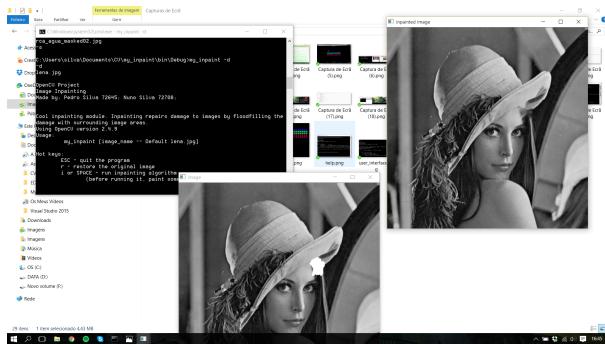


Fig. 3: Inpainting da imagem default - lena.jpg

O programa pode ser inicializado de duas formas: passando como argumento a imagem na qual se vai trabalhar, ou utilizando o comando *'-s'*, passando como argumento essa imagem com uma máscara a ser aplicada, ou seja, duas imagens onde uma delas é a máscara e a outra é a imagem original obtendo no final o resultado do inpainting dessas duas imagens.

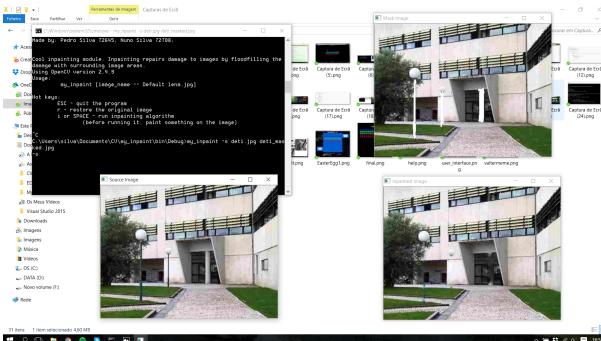


Fig. 4: Resultado do inpaiting com as duas imagens com o comando *-s*

IV INPAINTING

Quando se abre o programa com uma dada imagem, além da interface na consola já apresentada abre-se outra janela onde podemos aplicar a máscara de inpainting. Para efeitos de apresentação vamos usar a seguinte imagem:

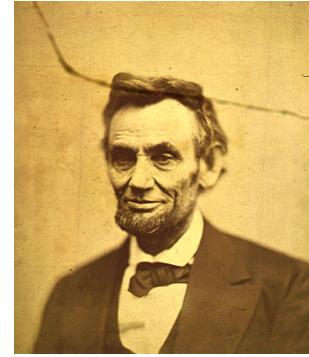


Fig. 5: Imagem de teste

O utilizador usa o rato para aplicar a máscara, desenhando-a a branco sobre a imagem.

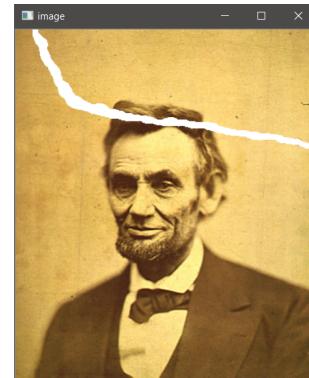


Fig. 6: Imagem de teste com máscara

Após aplicar a máscara o utilizador prime a tecla *i* ou a *barra de espaços* para produzir um resultado.

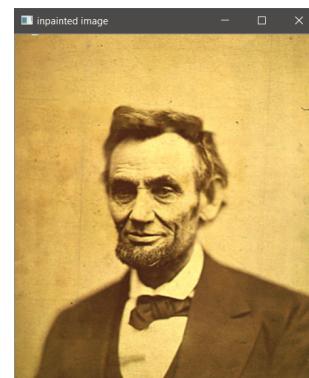


Fig. 7: Resultado da primeira experiência

Alternativamente, é possível obter-se outro resultado

carregando o programa com uma imagem e com uma máscara previamente feita.

Usando a mesma imagem e a seguinte máscara foi conseguido o seguinte resultado.

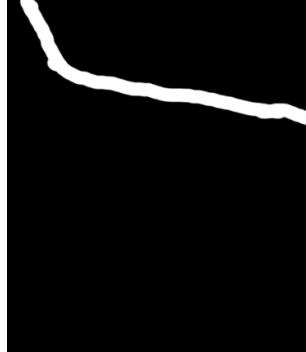


Fig. 8: Máscara previamente concebida



Fig. 9: Resultado da segunda experiência

Ao fazer-se este processo, uma janela com o resultado aparece automaticamente sem necessitar interação do utilizador.

Ora esta imagem resulta da diferença entre a máscara e a imagem original que é convertida para uma imagem em tons de cinzento e que depois de utilizar um threshold que vai detectar os pixels a branco (os pixels com um valor acima de 10 é lhes atribuído o valor 255) vai ser então usada no inpainting da imagem resultado fazendo o inpainting da imagem com o threshold referente a essa imagem;

Usando este processo é possível retirar marcas de água a imagens que era um dos objectivos deste trabalho. Ora criando uma imagem máscara sublinhando as letras da marca de água a branco é possível observar os seguintes resultados:

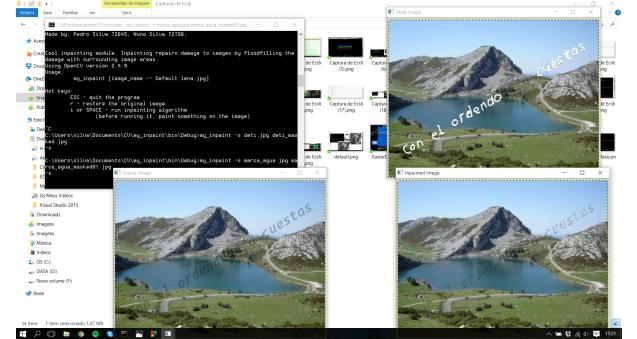


Fig. 10: Primeiro exemplo da remoção da marca de água da imagem

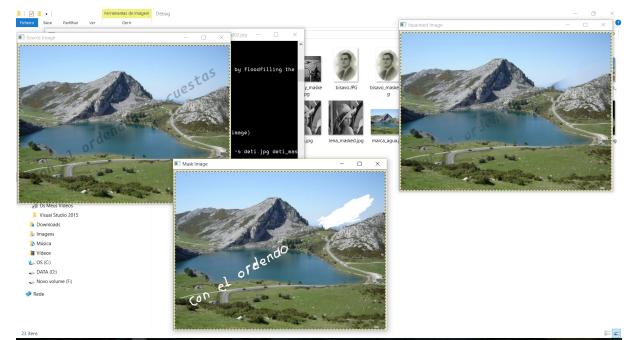


Fig. 11: Segundo exemplo da remoção da marca de água da imagem

Contudo também é possível observar que existem limitações neste modelo nomeadamente na preservação do detalhe da imagem e na preservação das regiões que se encontram muito "texturadas" ou seja regiões da imagem em que temos conjuntos de pixels com uma dada textura que em alguns casos pode ser diferente das texturas dos conjuntos de pixels presentes na vizinhança.

V CONCLUSÃO

O principal objetivo foi conseguido. Optou-se por reutilizar o código disponibilizado pelos samples do source do OpenCV e fazer uma reformulação eliminando algumas componentes que nos foram desnecessárias e acrescentando outras quando fosse necessário. Os principais problemas prenderam-se em desenvolver soluções que implementassem as funcionalidades duma interface user-friendly, portanto o trabalho inicialmente pensado foi simplificado mas continuado de maneira que o utilizador pudesse interagir com o programa de uma maneira fácil. De um modo geral e como estes problemas foram superados, tendo contornado estes obstáculos ajudou ao nosso crescimento como futuros engenheiros. Foi uma tema bastante interessante dado que as ferramentas de inpainting são bastante utilizadas na edição de imagens em programas como o Photoshop e foi uma oportunidade que nos permitiu desenvolver uma ferramenta deste género e ultrapassar as nossas expectativas e capacidades.

REFERENCES

Stack Overflow
OpenCV Inpating Tutorial
core.hpp
highgui.hpp
imgproc.hpp
photo.hpp
Alexandru Telea - An Image Inpainting Technique Based on the Fast
Marching Method
Navier-Stokes, Fluid Dynamics, and Image and Video Inpainting