Background Estimation Using Image Processing Technique

Maha Thafar¹, Salwa Aljehane²

¹,² Department of Computer Science, Kent State University, Kent, OH, USA

Abstract - Image Inpainting is the art of reconstruct missing areas. It is used to fix any damage in the images by removing the damage and filling this region based on estimating background. Background estimation is used to interpolate an estimate color of the background based on surrounding pixels’ color as an image processing technique. The purpose of this paper is to implement an existing digital image inpainting algorithm. This code is proposed to be used as a plugin in image processing framework in order to be easily accessed and used to fill a missing region with smooth background color in reasonable way.

Keywords: Image Processing, Background Estimation, ImageJ, and Inpainting.

1 Introduction

Nowadays, digital images are widely used in everyday life for different purposes. However, having a desirable image could be a challenge. Maintaining any damage in the image, or removing a selected object from the image, can be done using several methods to fill in the missing area, or to modify the damaged region. The most popular techniques are image inpainting and texture synthesis [1]. Image inpainting is a technique, which is used to fill or restore the area of removed object using the surrounding pixels to generate an appropriate color [2]. Texture synthesis is another technique to fill the missing image regions using texture information [3]. Background estimation technique is used for both static image and video [4].

In this paper, inpainting technique is explained and used in purpose of editing biological images such as estimating the background color after removing specific cell manually. In contrast, Oliveira [1] applied mask on the whole image to remove the damage, and then fill the removing area using inpainting technique. In general, inpainting methods are done by the following steps. First, the desired region is selected automatically using different techniques such as color detection, or manually by the user. Then, specify the boundary of this region and clear its color. After that, the missing value will be computed to color this area. The last step is different from algorithm to algorithm. Nevertheless, this project used an existing method that presented in [1] and add some more steps to improve the results using some existing enhancement and filtering techniques.

The remaining sections of this paper are organized as follow: literature review is presented in Section 2. The project concept is discussed in section 3. Design and development stages are explained in section 4. Section 5 includes the implementation details. Then results and discussions are given in section 6. Finally the conclusion with future work are drawn in the last section.

2 Review of literature

There has been a significant amount of research that has been done in the digital image-inpainting in computer vision, computer graphics, and other computer science fields. They manipulate the inpainting approach in several processes. Many algorithms have been presented in this area such as Fast Marching Method Algorithm [5], Exemplar-Based Image Inpainting [2], etc. There are different factors that affect the inpainting process such as missing region size, surrounding pixels, image type, and the speed of the algorithm.

Oliveira introduced a Fast Digital Image Inpainting algorithm, which used a convolution process [1]. In this method, the region to be inpainted is convolved with the kernel to compute the weighted average value based on the neighbor pixels’ value. Two diffusion kernels were used in this method in order to compute the missing value. It works by convolving the boundary with the filter and propagating the new color toward the target region. However, one hundred iterations were used in the inpainting process to generate an accurate color [1].

Alexandra [5] used inpainting technique based on the Fast Marching Method (FMM). This algorithm starts from the boundary of selected region and goes inside gradually by filling every pixel in the boundary first. The pixel to be inpainted is replaced by normalized weighted sum of all the known pixels in the neighborhood. Selection of the weights is an important matter. Once a pixel is inpainted, it moves to
next nearest pixel using Fast Marching Method. This method gave a good result with short run time.

Another study proposed a novel algorithm for digital image inpainting [6]. It is built based on mathematical theory of the Navier-Stoked equations [7] for fluids dynamic to apply image inpainting. This algorithm automatically transfers information into selected region. The main concept is: first travels along the edges from known regions to unknown regions. It continues isophotes, which means propagates the image Laplacian in the level-lines, while matching gradient vectors at the boundary of the inpainting region. Thus, some methods from fluid dynamics are used and the color is filled to reduce minimum variance in that area. The advantage of this approach is providing theoretical and numerical results [6].

3 Project concept and design

The project idea is based on outlining manually a specific area in a biological image such as cell, then removing everything inside the boundary by clearing the color. After that, filling that area with an estimation background color using neighbor pixels around the selected area as shown in figure 1. The last step is to make the colors blend seamlessly. In addition, many variables that affect the background should be taken into the account such as color, illumination, etc.

![Figure 1: Background Estimation Concept](image1.png)

4 Design and development

4.1 Data

A biological image from biology department at Kent State University in TIFF extension (Tagged Image File Format) was used. This format is chosen among different image formats since it is the default format of ImageJ [8], which is used later in implementation stage.

4.2 Software

1) ImageJ: It is a public domain Java Image Processing program. It can display, edit, analyze, process, and save images. In addition, it can print 8-bit, 16-bit and 32-bit images. It can read many image formats. We will use this tool, which provides a lot of built-in or user-define macros and plugins that can be applied on the image [9].

2) Eclipse: It is an Integrated Development Environment (IDE). Eclipse can be used to develop application, which is written by Java programming language and then export plugins to be used in different applications. All plugins are executed through Eclipse software.

Process:

1. Make a segmentation of target area manually
2. Remove this area or object
3. Clear the background
4. Fill the region by applying specific method
5. Improve the image by applying different filters.

5 Implementation

Our code is developed based on Fast Digital Image Inpainting Algorithm using Convolution Based Method [1]. As most inpainting algorithms, four steps were used to perform this task. The first step is allowing the user to select the desired region by using any selection tools. The second step is detecting this region which called Region of Interest (ROI). The third step is initializing the ROI by clearing its color using clear function. The final step is generating the color based on the surrounding pixels to inpaint the target region. The main idea to generate the color depends on convolution operation. A kernel with specific coefficients’ values that were given in Oliveira’s paper [1] is used. These coefficients are \( a = 0.073235, b = 0.176765, c = 0.125 \). The most important instructions in the pseudocode and two diffusion kernels that used in the convolution operation are shown in figure 2.

![Figure 2: Diffusion kernels and pseudocode](image2.png)

The kernel is convolved with the ROI and generates the color starting from the border inward to the region as shown in figure 3.
The developed code is divided into three major parts as the following:

Part 1: Accessing the ImageJ framework for image processing and its plugins through Eclipse. The image is considered as a variable to be read.

Part 2: Detecting the ROI and getting its information, such as the start point coordinates, the width, and the height of the ROI. These parameters are sent to the kernel convolution method.

Part 3: Kernel-Convolution method is the main part of the code. It receives the information including: start point of the ROI, and its width and height then starts to do inpainting. The process of this method is iterative. It applies the convolution kernel on each pixel in a spiral-like fashion, and then shrinks the border by going inside ROI. The last step of the project is applying ImageJ filters such as Mean and Gaussian blur filters on the ROI to improve and smooth the result. All these steps are shown in figure 4.

The code is exported to be an inpainting plugin, which is used through ImageJ framework.

## 6 Results and discussion

We applied the plugin on biological images using ImageJ framework. Different regions of interest were tested and the results are shown in figures 5 and figure 6. The results are better if the user selects a small area to be inpainted rather than larger area.

![Figure 5a: ROI selection on original image](image1)

![Figure 5b: Result of inpainting](image2)

![Figure 6a: ROI selection on original image](image3)

![Figure 6b: Result of inpainting](image4)

## 7 Conclusion and future work

In this paper, an image inpainting implementation based on Oliveira algorithm has been presented. However, generating the background color of the removing area is done based on surrounding pixels colors. Some filters are applied to improve the result such as removing the blur. ImageJ and Eclipse java were used to perform this task.

For future works, we are looking to improve the result to be more accurate. In addition, we want to include different filters while designing the plugin. This is useful to increase the performance and minimize user interaction to do everything automatically.
8 References


