Framework for Image Inpainting by using Fast Marching Method

Hsint Hsint Htay, May Phyo Oo
Computer University, Loikaw
hsinthsinthhh@gmail.com, mayphyooo@gmail.com

Abstract

Image inpainting involves filling in part of an image using information from the surrounding area. Applications include the restoration of damaged photographs and movies and the removal of selected objects. A framework for image inpainting by using fast marching method is implemented in this paper. Firstly, the average smoothness weight average of the image is estimate of the input image. After the user selects the regions to be restored, this system has compute the distance map of the inpainted pixels to the boundary and then fills the pixels to the closest pixels to the known image area until up to fill the all pixels of the selected area. This method is very simple to implement, fast and produces nearly identical results.

Keywords: Image Inpainting, Fast Marching Method

1. Introduction

Applications range from removing objects from a scene to re-touching damaged paintings and photographs. The goal is to produce a revised image in which the inpainted region is seamlessly merged into the image in a way that is not detectable by a typical viewer. Traditionally, inpainting has been done by professional artists. For photography and film , inpainting is used to revert deterioration (e.g., cracks in photographs or scratches and dust spots in film), or to add or remove elements (e.g., removal of stamped date and redeye from photographs, the infamous “airbrushing” of political enemies [12]). A current active area of research is to automate digital techniques for inpainting .

Local inpainting is used from the original image of information about the local neighbourhood of a pixel only. The problem of local inpainting as, given a region to be inpainted Ω and its boundary $\partial\Omega$, to synthesize pixel values from the boundary inwards, using neighbourhood pixel information to continue the inpainting process.

These methods assume a prior information about the probability distribution of the relation between a pixel value and its neighbourhood, which will help fill in a pixel lying on the hole boundary. Also, other properties such as high smoothness, low total variation or low curvature are assumed to create the framework on which the actual algorithm runs.

There are two significant approaches to local information based hole filling of images:

(i) Based on Partial Differential Equations
(ii) Based on convolution [13]

Global inpainting is used from the global statistics to aid the local statistics. Inpainting based on global statistics is a fledgling concept but worth a mention here because it highlights a potential weakness in local inpainting. Local algorithms give identical completions when the immediate boundary of the hole is identical. Human visual system takes more global information into account.

Global statistics are similarly important in painting restoration. The major drawback with respect to this paper is that the algorithm does not perform well with natural textures as seen in photographs. It captures the .look. of a training image in a probability distribution over images, using image histograms. Probability of an image is defined by means of a small number of sufficient

statistics or features, each of which can be evaluated at an arbitrary location in the image.

In this paper, a framework for image inpainting based on fast marching method is implemented. Related works concerning with image inpainting are presented in section 2. In section 3, Fast Marching Method is presented. System Design of image inpainting based on FMM is expressed in section 4. In section 5, the implementation is presented and are expressed in section 6.

2. Related Works

M. Bertalmio et.l implemented image inpainting which involves filling in part of an image or video using information from the surrounding area. Their approach used ideas from classical fluid dynamics to propagate isophote lines continuously from the exterior into the region to be inpainted and the method was directly based on the Navier-Stokes equations for fluid dynamics, which has the immediate advantage of well-developed theoretical and numerical results [9].

G.Suneet et.l presented the inpainting which filled the damaged region or "holes" in an image with surrounding color and texture information. Their algorithm was based on solving Laplace equation with Dirichlet boundary conditions, incorporated with texture filling, and its main asset was the bounded search window. [5]

A. Criminisi et.l expressed two classes algorithms which have been addressed texture synthesis algorithms for generating large image regions from sample textures and inpainting techniques for filling in small image gaps. They introduced exemplar-based texture synthesis contains the essential process required to replicate both texture and structure and The actual colour values are computed using exemplar-based synthesis. [1]

A. Telea explained digital inpainting which provided a means for reconstruction of small damaged portions of an image. Their method was based on the fast marching method for level set applications. [3]

Image inpainting is the time-tested art of removing defects from pictures. Defects are arbitrarily defined by the user and vary from a scratch on an old family photograph to graffiti on an otherwise appealing scene to completely removing small objects from a scene. Professional artists have long practiced inpainting, with considerable expense and considerable effort.

Inpanting has been studied with various forms in research field. The various forms of inpainting applications are:

(1) Image restoration (Repairing images physically damaged by tears, stains, scratches, writing, etc)
(2) Correction (Fixing undesirable artifacts such as read eyes, flash reflection, blemish, wrinkles, etc)
(3) Wireless image transmission (Lost data recovering) and
(4) Special effect (Removing object)

3. Fast Marching Method

To inpaint the whole Ω, we iteratively apply Equation 1 to all the discrete pixels of $\partial\Omega$, in increasing distance from $\partial\Omega$'s initial position $\partial\Omega$i, and advance the boundary inside Ω until the whole region has been inpainted.

$$I(P) = \frac{\sum_{q \in B\varepsilon(p)} w(p,q)[I(q) + \nabla I(q)(p-q)]}{\sum_{q \in B\varepsilon(p)} w(p,q)} \quad (1)$$

Where Ω is region to be inpainted. $\partial\Omega$ is the boundary of the region to be inpainted. $B_\varepsilon(p)$ is neighborhood of size. I(q) is original image. I(p) is inpainted image.

In brief, the FMM is an algorithm that solves the Eikonal equation:

$|\nabla T| = 1$ on Ω, with $T = 0$ on $\partial\Omega$ **(2)**

The solution T of Equation 3 is the distance map of the Ω pixels to the boundary $\partial\Omega$. The level sets, or isolines, of T are exactly the successive boundaries $\partial\Omega$ of the shrinking Ω that we need for inpainting. The normal N to $\partial\Omega$, also needed for inpainting, is exactly ∇T. The FMM guarantees that pixels of $\partial\Omega$ are always processed in increasing order of their distanceto-boundary T , i.e., that the closest pixels to the known image area are inpainted frist.

The FMM over other Distance Transform (DT) methods that compute the distance map T to a boundary $\partial\Omega$ is described. The FMM's main advantage is that it explicitly maintains the narrow band that separates the known from the unknown image area and specifies which is the next pixel to inpaint. Other DT methods compute the distance map T but do not maintain an explicit narrow band. Adding a narrow band structure to these methods would complicate their implementation, whereas the FMM provides this structure by default. To explain our use of the FMM in detail–and since the FMM is not straightforward to implement from the reference literature. The FMM maintains a so-called narrow band of pixels, which is exactly our inpainting boundary $\partial\Omega$. For every image pixel, we store its value T, its image gray value I (both represented as floating-point values), and a flag f that may have three values:

- BAND: the pixel belongs to the narrow band. Its T value undergoes update.
- KNOWN: the pixel is outside $\partial\Omega$, in the known image area. Its T and I values are known.
- INSIDE: the pixel is inside $\partial\Omega$, in the region to inpaint. Its T and I values are not yet known.

The FMM has an initialization and propagation phase as follows. First, we set T to zero on and outside the boundary $\partial\Omega$ of the region to inpaint and to some large value inside, and initialize f over the whole image as explained above. All BAND points are inserted in a heap NarrowBand sorted in ascending order of their T values. Next, we propagate the T, f, and I values using the code shown in the following algorithm.

```
while (NarrowBand not empty)
{
extract P(i,j) = head(NarrowBand);
                    /* STEP 1 */
f(i,j) = KNOWN;
for (k,l) in (i1,j),(i,j1),(i+1,j),(i,j+1)
if (f(k,l)!=KNOWN)
{
if (f(k,l)==INSIDE)
{
f(k,l)=BAND;                    /* STEP 2 */
inpaint(k,l);
                                    /* STEP 3
*/
}
T (k,l) = min(solve(k1,l,k,l1),     /* STEP 4
*/
solve(k+1,l,k,l1),
solve(k1,l,k,l+1),
solve(k+1,l,k,l+1));
insert(k,l) in NarrowBand;      /* STEP 5
*/
}
}
float solve(int i1,int j1,int i2,int j2)
{
float sol = 1.0e6;
if (f(i1,j1)==KNOWN)
if (f(i2,j2)==KNOWN)
{

float r =
sqrt(2(T(i1,j1)T(i2,j2))*(T(i1,j1)T(i2,j2)));
float s = (T(i1,j1)+T(i2,j2)r)/2;
```

```
if (s>=T(i1,j1) && s>=T(i2,j2)) sol = s;
else
{ s += r; if (s>=T(i1,j1) && s>=T(i2,j2))
sol = s; }

}
else sol = 1+T(i1,j1));
else if (f(i2,j2)==KNOWN) sol =
1+T(i1,j2));
return sol;

}
```

Step 1 extracts the BAND point with the smallest T. Step 2 marches the boundary inward by adding new points to it. Step 3 performs the inpainting. Step 4 propagates the value T of point (i, j) to its neighbors (k, l) by solving the finite difference discretization of Equation 3 given by

$$\max(D^{-x}T, -D^{+x}T, 0)\ 2 + \max(D^{-y}T, -D^{+y}T, 0)\ 2 = 1 \quad (3)$$

where $D^{-x}T(i, j) = T(i, j) - T(i - 1, j)$ and $D^{+x}T(i, j) = T(i+1, j) - T(i, j)$ and similarly for y. Following the upwind idea of Sethian, it can be solved by Equation 3.4 for (k, l)'s four quadrants and retain the smallest solution. Finally, Step 5 (re)inserts (k, l) with its new T in the heap.

4. System Design of Image Inpainting based on FMM

In this paper, image inpainting based on FMM is implemented. Image inpainting is the process of filling in missing data in a designated region of a still or video image. Applications range from removing objects from a scene to re-touching damaged paintings and photographs. The goal is to produce a revised image in which the inpainted region is seamlessly merged into the image in a way that is not detectable by a typical viewer. System design of image inpainting based on fast marching method is shown in figure 1.

There are four components in the inpainting based on fast marching method. They are :

(i) Estimate the average smoothness weight average function
(ii) Select inpainted region
(iii) Compute distance map T of the inpainted pixels to the boundary
(iv) Fast marching method

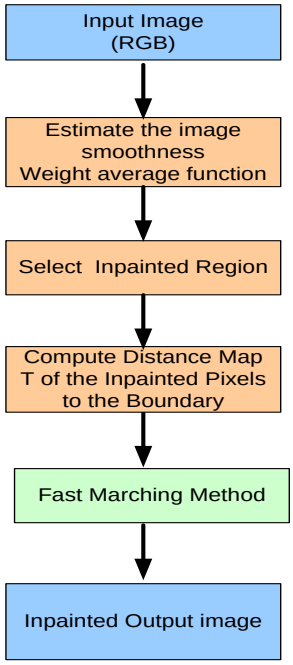

Figure 1. System Design

Input image is RGB (red, green, blue) light. Although all colors of the visible spectrum can be produced by merging red, green and blue light, monitors are capable of displaying only a limited gamut (i.e., range) of the visible spectrum. File format is .jpg format.

In the estimate the image smoothness component, it is estimated the smoothness as a weighted average over a known image neighborhood of the pixel to inpaint. Grayscale intensity vaules of the known image are calculated in this step.

And then the desired inpainting region is selected in the select inpainted region component. The image intensity function plays the role of the stream

function whose isophote lines define streamlines of the flow. After the user selects the regions to be restored, the algorithm automatically transports information into the inpainting region.

In the Compute distance map T of the inpainted pixels to the boundary component, the Fast Marching Method (FMM) is an algorithm that solves the Eikonal equation in equation 3. The solution T of equation 3 is the distance map of the Ω pixels to the boundary ∂Ω. The level sets, or isolines, of T are exactly the successive boundaries ∂Ω of the shrinking Ω that is required for inpainting. The normal N to ∂Ω, also needed for inpainting, is exactly ∇T. The FMM guarantees that pixels of ∂Ω are always processed in increasing order of their distanceto-boundary T. Distance Transform (DT) methods compute the distance map T to a boundary ∂Ω.

Since FMM inpainting algorithm is designed for both restoration of damaged photographs and for removal of undesired objects on the image, the regions to be inpainted must be marked by the user.

5. Implementation

Image Inpainting based on Fast Marching Method is implemented in this system. There are four Menus: File Menu, Estimate Weight Function Menu, Compute Menu and Fast Marching Method Menu.

Loaded original image (300X225) is shown in figure 2 which has noise yellow line exists and estimate weight of grayscale image is shown in figure 3. Here, gray value of yellow nose line is calculated for noise removing.

Figure 2. Loaded Original Image

If the user is selected in noise region of grayscale image is as shown in figure 4, the algorithm is recognized in this selected portion (150X100). And the level set or yellow lines of pixels are computed to the boundary of the grayscale images. In this implementation, the number of iteration is 2000 times is computed and inpainted in the grayscale image. The inpainted output image is as shown in figure 5.

Figure 3. Estimate Weight of Grayscale Image

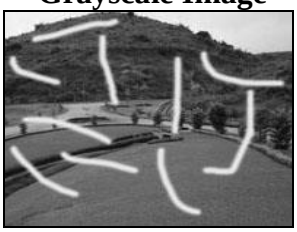

Figure 4. Select Region of Grayscale Image

Figure 5. Inpainted Image

If the number of iteration is less than the number of pixels of selection portion, the noise of yellow lines remains in the inpainted image as shown in figure 6. The number of iteration is 1000. Figure 6 (a), (b), (c) and (d) are original image, grayscale image, selected region and inpainted image respectively.

Figure 6(a)

Figure 6(b)

Figure 6(c)

Figure 6(d)

6. Conclusion

In this paper, a framework for image inpainting by using fast Marching method is implemented and tested with 100 images. Grayscale images have to be converted from RGB color images. Therefore, it doesn't take runtime in this implementation. The tested images have low resolution and noises are same color. If the different color noise are used in the original image, the result inpainted image is no good result. Any condition noises will be restored in further extension.

References

[1] A. Criminisi, P.P. erez and K. Toyama, " Region Filling and Object Removal by Exemplar-Based Image Inpainting", IEEE Transactions on Image Processing, Vol.13, No.9, September 2004.

[2] A. P. Witkin, "Scale-space filtering", Proceedings International Joint Conference on Artificial Inteligence, pages 1019 - 1022, 1983.

[3] A. Telea, " An Image Inpainting Technique Based on the Fast Marching Method", Journals of Graphics Tools, Vol.9, No.1, 25-36.

[4] A.A. Farag and M.S. Hassouna, "Theoretical Foundations of Tracking Monotonically Advancing Fronts Using Fast Marching Level Set Method", Technical Report, Computer Vision and Image Processing Laboratory, February 2005.

[5] G.Suneet, M. Ankush and G. Sumit, " A Unified Approach for Digital Image Inpainting Using Bounded Search Spact".

[6] http://www.eecs.harvard.edu/~sanjay /inpainting/

[7] J. A. Sethian. "A Fast Marching Level Set Method for Monotonically Advancing Fronts." Proc. Nat. Acad. Sci. 93:4 (1996), 1591—1595.

[8] J. A. Sethian. Level Set Methods and Fast Marching Methods, Second edition. Cambridge, UK: Cambridge Univ. Press, 1999.

[9] M. Bertalmio, A. L. Bertozzi, and G. Sapiro. "Navier-Stokes, Fluid Dynamics, and Image and Video Inpainting." In Proc. ICCV 2001, pp. 1335-1362, IEEE CS Press 1. [CITY]: [PUB], 2001.

[10] M. Bertalmio, G. Sapiro, V. Caselles, C. Ballester, “Image Inpainting”, *SIGGRAPH 2000*, pages 417-424.

[11] M. Oliveira, B. Bowen, R. McKenna, and Y. -S. Chang. “Fast Digital Image Inpainting.” In Proc. VIIP 2001, pp. 261—266, [CITY]: [PUB], 2001.

[12] P. Ninad “ Digital Image Restoration Techniques And Automation”, Project.

[13] R.Holm, “ Image Inpainting using Nonlinear Partial Differential Equations”, Thesis in Applied Mathematics.

[14] T. Chan and J. Shen. “Mathematical Models for Local Non-Texture Inpaintings.” Technical Report CAM 00-11, Image Processing Research Group, UCLA, 2000.

[15] T. Chan and J. Shen. “Non-Texture Inpainting by Curvature-Driven Diffusions (CDD).” Technical Report CAM 00-35, Image Processing Research Group, UCLA, 2000.