IMAGE INPAINTING BASED ON PARTICLE SWARM PTIMIZATION

Sarab M. Hameed , Nasreen J. Kadhim, Mahmood A. Othman
Department of computer science, college of science, University of Baghdad. Baghdad-Iraq.

Abstract

Digital image inpainting is a technique that can repair a portion of damaged or removed image by means of automatic mechanisms.

This paper proposes an automatic approach for reconstruction of small missing or damaged portions of images using Particle Swarm Optimization (PSO). The PSO inpainting approach attempts to find the best match between Red, Green, and Blue (RGB) channels of the pixels masked (i.e. pixels of the region to be removed) and pixels in the surrounding area, and transfer the color of the best matching. The results show that our mechanism is more efficient (i.e. less time required to inpaint the masked region) when compared with genetic - based image inpainting.

Keywords: Image Inpainting, Image Restoration, Particle Swarm Optimization

صبغ الصورة بالاعتماد على افضل سرب طيور

سراب مجيد حميد، نسرين جواد كاظم، محمود عثمان

قسم علوم الحاسبات،كلية العلوم، جامعة بغداد. بغداد – العراق.

الخلاصة

صبغ الصور الرقمية هي تقنية لتصليح الجزء المخرب او حذف من الصورة بطريقة آلية. يقترح هذا البحث طريقة لإعادة الجزء المخرب باستخدام أفضل سرب طيور. يحاول PSO الصبغ عن طريق إيجاد أفضل تطابق بين القنوات الأحمر والأخضر والأزرق للنقطة المغطاة (النقطة المراد حذفها) والنقاط المحاطة بهذه النقطة ثم نقل اللون لافضل تطابق. أثبتت النتائج بان الطريقة المقترحة اكفأ (تحتاج وقت قليل) عند مقارنتها مع صبغ الصورة المعتمد على الخوارزمية الجينية.

Introduction

Inpainting refers to specific image restoration task of reconstructing an image with missing or damaged region. Application ranges from removing objects from the scene all the way to retouching damaged paininting and photo. Therefore, the goal of inpainting is to restore a damaged or corrupted image where part of the information has been lost. Such degradations of an image may have different origins like image transmission problems or degradations in real images due to storage conditions or manipulations. Inpainting may also be an interesting tools for graphics people who need to remove artifcially some parts of an image such as some overlapping text or tricks used in special effects . In any case, the restoration of missing parts has to be done so that the final image looks unaltered for an observer who does not know the original image.

Digital image inpainting was first proposed by Bertalmio et al. [1]. They have introduced a technique for digital inpainting based on partial differential equations (PDEs). A user provided mask specifies the portions of the input image to be retouched and the algorithm treats the input image as three separate channels (R, G and B). For each channel, it fills in the areas to be

inpainted by propagating information from the outside of the masked region along level lines. Their algorithm produces very impressive results, however, usually requires several minutes on current personal computers for the inpainting of relatively small areas. Such a time is unacceptable for interactive system and motivated us to design a simpler and faster algorithm capable of producing similar results in just a few seconds.

Chan and Shen proposed two image-inpainting algorithms [2, 3]. The *Total Variational* (TV) inpainting model uses an Euler-Lagrange equation and inside the inpainting domain the model simply employs anisotropic diffusion [4] based on the contrast of the isophotes. This model was designed for inpainting small regions and while it does a good job in removing noise, it does not connect broken edges (single lines embedded in a uniform background). The *Curvature-Driven Diffusion* (CDD) model extended the TV algorithm to also take into account geometric information of isophotes when defining the"strength" of the diffusion process, thus allowing the inpainting to proceed over larger areas. CDD can connect some broken edges, but the resulting interpolated segments usually look blurry.

Zaynab R. utilizes a Genetic Algorithm (GA) to restore missing parts on the image and also remove undesired object from the image [5]. The user selects the region to be inpainted, then, the GA is used to fill in the selected region in raster scan order.

In this research, Particle Swarm Optimization (PSO) is adopted for image inpainting. PSO is a stochastic, population-based evolutionary computer algorithm for problem solving. A problem is given, and some way to evaluate solution to it exists in the form of a fitness function. A communication structure or social network is also defined, assigning neighbors for each individual to interact with. Then a population of individuals defined as random guesses at the problem solutions is initialized. These individuals are candidate solutions. They are also known as the particles, hence the name particle swarm. The particles iteratively evaluate the fitness of the candidate solutions and remember the location where they had their best success. The individual's best solution is called the particle best or the local best. Each particle makes this information available to their neighbors. They are also able to see where their neighbors have had success. Movements through the search space are guided by these successes, with the population usually converging, by the end of a trial, on a problem solution better than that of non-swarm approach using the same methods [6] [7].

In the rest of this paper, the proposed inpainting algorithm is presented in section 2. Then, in section 3, the PSO image inpainting results are illustrated and compared with those results obtained using GA in [5].

The Inpainting Algorithm

The first step of inpainting algorithm is specify the regions to be inpainted (i.e. mask the region to be filled). Then, PSO is applied to fill the masked region with colors browsed from the pixels in the surrounding region of the mask. The PSO inpainting process is illustrated in the following steps:

Step1: Particle Representation and Swarm Initialization

Initialize a swarm of particles of size *s* with random positions and random velocities. The positions of each particle, *Pi,* are represented by *x* and *y* coordinates such that:

$$0 \le P_i.x \le \text{ImageWidth}$$

$$0 \le P_i.y \le \text{ImageHeight}$$

Additionally, each particle has two velocities, one move the particle toward x-dimension, vx and the other, vy, moves the particle towards y-dimension. x and y coordinates are randomly initialized but constrained with values out of the masked region (i.e. region to be removed from the original image). Velocity values are initialized randomly and must be restricted to some minimum and maximum values using piece-wise function. The velocity of particle Pi towards x coordinate is calculated using equation 1.

$$P_i.v_x^0 = V_{\min x} + [V_{\max x} - V_{\min x}] * rand() \quad (1)$$

Similarly, the velocity of particle Pi towards y coordinate is calculated using equation 2.

$$P_i.v_y^0 = V_{\min y} + [V_{\max y} - V_{\min y}] * rand() \quad (2)$$

Where rand () is random function in the range [0, 1]. *Vminx= -Vmaxx, Vmaxx* is set to the half image width (w) and *Vminy=-Vmaxy*, and Vmaxy is set to the half image height (h).

Step2: Fitness Evaluation

For each particle, evaluate the desired optimization fitness function. To evaluate the fitness function, one has to formulate a desired objective function. In the inpainting problem, one has to fill the masked region in scan order(i.e. from top to bottom and from left to right). Thus, at each step of inpainting, one can get a set of pixels, all of them except one were filled with colors. The shape of this set is L, so-called L-shape. If we have to point P with color that is consistent with the remaining part of the image, then we have to search the image for all possible source L-shaped window outside the masked region that *best fit* the target L-shaped window centered at P. By this, best fit means minimum sum of Euclidean distance between source pixels and their corresponding target pixels. Formally speaking:

$$ObjF = \sum_{i=1}^{12} \sqrt{(R_{s,i} - R_{t,i})^2 + (G_{s,i} - G_{t,i})^2 + (B_{s,i} - B_{t,i})^2} \quad (3)$$

Where

R, G, and B are Red, Green , and Blue channels.
t pixel to be filled (i.e. pixel in the masked region)
s pixel in the original image out of the masked region.
The, in PSO, each particle has a fitness value that ranks this particle within the swarm. Fitness function is evaluted using equation (4).

$$Fitness = \frac{1}{ObjF} \quad (4)$$

Step 3:Update local and global best Particle

Each particle has a best fitness associated with its best position in *bestPi*:

$bestP_i.x$: Best position of Pi in x coordinate.

$bestP_i.y$: Best position of Pi in y coordinate.

For each particle, compare particle's fitness with the best fitness obtained so far. If current value is minimum than *Pbest*, then set *Pbest* to current value. Then, identify the best position (*Gbest*) among all particles in the swarm.

Step 4: Update particle velocities and position

Particle swarm has two primary operators: velocity upadtae and position update. During each iteration each particle is accelerated toword the particles pervious best position and global best position. At each iteration a new velocity value for each particle is calculted based on its current velocity, the distance from its pervious best position, and the distance from the global best position. The new velocity value is then used to calculate the next position of the particle in the search space. Change the velocity and position of the particle according to equation 5 and 6 respectively [8].

$$\begin{aligned} v_{id}.x &= v_{id}.x + c_1 rand()(bestP_{id}.x - p_{id}.x) + c_2 Rand()(Gbest_{id} - x_{id}) \\ v_{id}.y &= v_{id}.y + c_1 rand()(bestP_{id}.y - p_{id}.y) + c_2 Rand()(Gbest_{id} - y_{id}) \end{aligned} \quad (5)$$

Where c1 and c2 are positive constants. rand () and Rand() are two random function in the range[0,1].

$$x_{id} = x_{id} + v_{id} \quad (6)$$

Step 5: Termination

Step 2 to step 4 is then iterated until a number of iteration, called maximum iteration (*Maxi*), is reached. After termination, the *Gbest* represent the best source pixel that matches the target pixel to be filled. Then, the Red, Green, and Blue channel of the source pixel are assigned to the target pixel.
The process of applying PSO inpainting algorithm is repeated for every pixel in the masked region. Finally, the restored image (i.e. removing masked region) is obtained.

Experimental Results

This section illustrates the results of PSO inpainting algorithm. The results are compared with those in [5] that used GA for filling masked region. We test our algorithm on a number of images of different sizes, different numbers of pixels in the masked region, and different textures. We explain our results with six experiments depicted in figure 1 and 2. The setting of PSO parameter is as shown in table 1.

Table 1: PSO Parameters

Parameter	value
C_1 , C_2	2
Max_i	3
Swarm size	Average 15

Table 2 illustrates the image size, the number of pixels in the masked region, the percentage

occupation of the masked region from the total image area. Formally speaking.

Table 2: Image Properties

Image Name	Image Size	Number of Pixels in the masked region	Masked Region Occupation
Car	214 x168	2343	6.45
Freedom	330 x 213	3757	5.34
Donkey	277 x 200	2077	3.74
Shalal	128 x 174	4491	20.16
Rock	131 x 99	1340	10.33
Color Line	198 x 148	731	2.49

The quality of PSO inpainting results is good and accepted and the time required is too little when compared with GA inpainting time.

Table 3: Compassion between PSO and GA Computation Time

Image Name	PSO Time(sec)	GA Time (sec)
Car	29	338
Freedom	34	628
Donkey	18	286
Shalal	49	133
Rock	16	133
Color Line	20	102

Table 4 illustrates the Signal to Noise Ration (SNR) for PSO-based inpainting algorithm results and GA-based inpainting results.

Table 4: SNR of PSO -Based Inpainting and GA- Based Inpaininting

Image Name	PSO SNR	GA SNR
Car	0.998	1.008
Freedom	0.933	0.928
Donkey	0.914	0.909
Shalal	1.009	1.013
Rock	0.910	0.910
Color Line	1	1

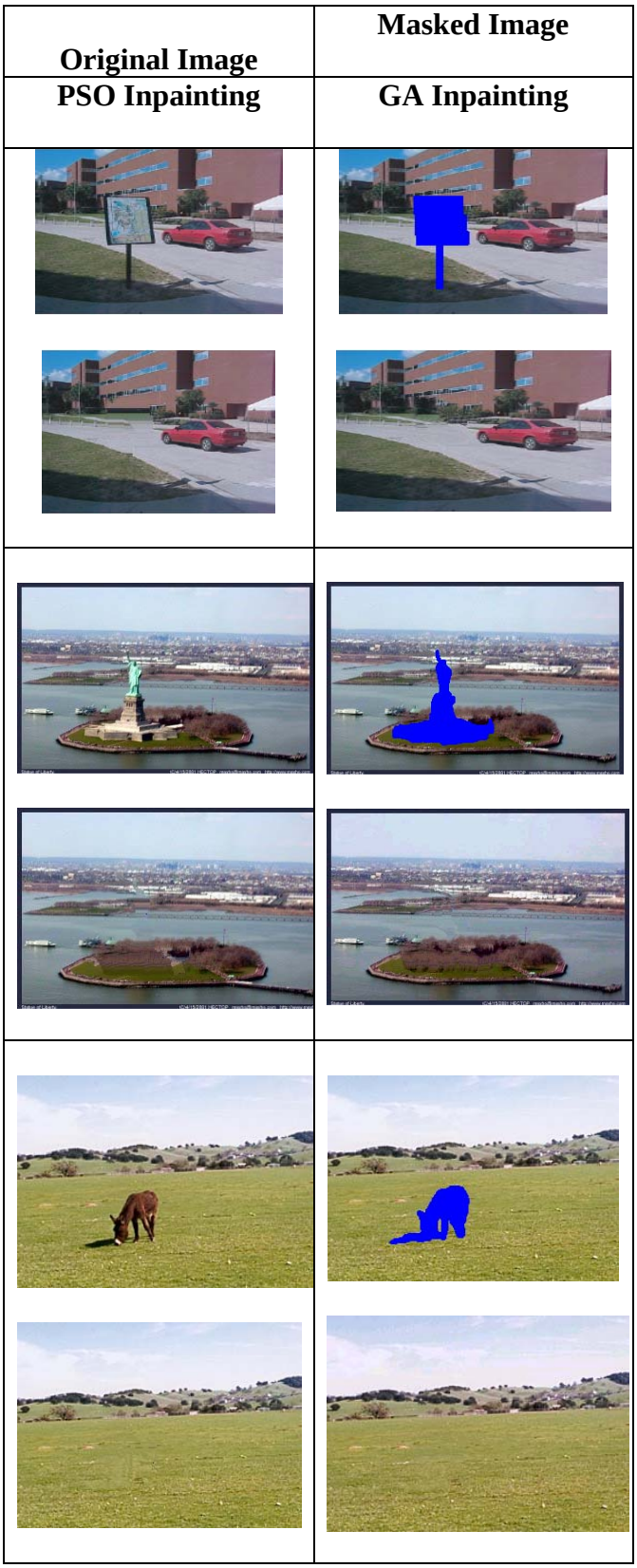

Figure 1: Inpainting Comparison Results between PSO and GA

Figure 2: Inpainting Comparison Results between PSO and GA

Conclusions

This paper presents a simple and fast inpainting algorithm filling holes in an image based on PSO. The results produced by this simple algorithm are comparable to those obtained by GA but faster (i.e. required too little time to produce the results), thus making inpainting practical for interactive applications.

References

1. Bertalmio, M, Sapiro, G., Caselles, V., Ballester, C., **2000**, "*Image Inpainting*", In Proceeding ACM conference computer graphics (SIGGRAPH), pages 417- 424 , Neworleans , USA . http : / / mountais . ece . umn .edu/~guille/inpainting.htm
2. Chan, T., Shen, J. **2000**, "*Mathematical Models for Local Deterministic Inpaintings*", UCLA CAM TR 00-11.
3. Chan, T., Shen, J. **2000**, "*Non-Texture Inpainting by Curvature-Driven Diffusions (CCD)*", J. visual comm. Image Rep.,4(12).
4. Perona, P. Malik, J. **1990**, "*Scale-space and Edge Detection Using Anisotropic Diffusion*", IEEE PAMI 12, pp. 629-639.
5. Zayneb R., **2005**, *"Utilizing Evolutionary Algorithm for Inpainting Problem*", M. SC. Thesis, University of Baghdad
6. Kennedy J.and Eberhard R. C., **1995**, "*Particle Swarm Optimization*", Proc. of IEEE Int'l Conf. on Neural Networks, Piscataway, NJ, USA, pp. 1942-1948.
7. Kennedy J., Eberhart R. C. and Shi Y., (2001), "*Swarm Intelligence*", Morgan Kaufmann, San Mateo, CA.
8. Yuhui Shi, **2004**, "*Particle Swarm Optimization*", IEEE Neural networks Society.