Image inpainting의 성능 개선에 관한 연구

공재웅*, 김성현*, 김태형*, 김두영*
*동아대학교 전자공학과

A Study on The performance Improvement of Image inpainting

Jae-Woong Gong*, Sung-Hyun Kim*, Tae-Hyoung Kim*,
Doo-Yung Kim*
*Dept. of Electronics Engineering, Dong-A University
E-Mail : netgjw@nate.com

요 약

대부분의 영상은 다양한 이유(노이즈, 전송과정 중 발생하는 문제 등)로 인해 항상 좋은 품질을 보여주진 못한다. 이렇게 훼손된 영상의 복원은 다양한 정보를 제공한다. 이런 훼손된 영상을 복원하기 위해 Median filtering과 같은 기존의 처리 방법들은 주변 화소(Pixel)를 평활화(Smoothing) 처리를 하기 때문에 noise 처리에는 좋으나 원 영상의 중요한 에지 성분까지도 평활화 처리를 함으로써 에지 부분의 공간적 이동을 초래할 수 있다. 이러한 문제점을 해결하기 위하여 image inpainting 방법이 제안되고 있으며, inpainting 기법에는 편미분 방정식(PDE)을 이용한 방법, 텍스쳐 병합 기반의 방법들이 있다. 그러나 이러한 inpainting 기법들은 연산 수행시간이 많이 소요되는 단점이 있다. 따라서 본 연구에서는 image inpainting을 수행시 소요되는 연산시간을 줄이는 fast image inpainting 알고리즘을 제안한다.

I. 서 론

원래 inpainting이란 용어는 예술 분야에서 훼손된 미술 작품을 복원하기 위한 방법을 지칭하는 것이며 디지털 영상처리에서 Image inpainting은 오래된 영상이나 손상된 영상을 PC 기반으로 원래 모습으로 복원하는 기술, 지정된 영역을 제거하는 기술을 말한다.[3] Image inpainting의 응용분야는 사진의 복원으로부터 영상에서 불필요한 부분(자막, 스탭, 선전 등)을 제거하고 또한 특수한 효과를 표현하는데 이용된다. Image inpainting 이전의 처리 방법들은 많은 시간을 필요로 할 뿐만 아니라 수작업으로 진행된다. 또한 전문가의 미적 수준에 따라서 결과에 많은 영향을 미치게 된다. 편집시간을 줄이고 일반적이나 고품질의 결과는 반드시 필요하다.[4] 앞으로의 반도체 설계기술의 발달과 함께 컴퓨터의 연산능력 향상은 이전의 편집기법보다 훨씬 어려운 수학적 이론과 수치적 방법이 필요한 방정식을 영상 편집기법에 적용 가능하게 한다. Image inpainting에는 다양한 알고리즘들이 제안되어져 있다. 본 논문에서는 inpainting할 영역을 최소화하여 보다 빠른 연산속도에 관한 알고리즘에 초점을 맞추고 있다.

II. 배경이론

Image inpainting는 [M. Bertalmio]가 제안한 Laplacian을 이용한 Image inpainting[1], [Chan and Shen]이 제안한 Euler-Lagrange equation 방정식을 이용한 Image inpainting[2], 편미분 방정식(PDE)을 이용한 Image inpainting, 텍스쳐 병합기반의 Image inpainting등 다양하게 제안되어 왔다. 이러한 Image inpainting 방법들은 영상의 사이즈와 inpainting 할 영역의 사이즈에 따라 연산 시간이 많이 소요되는 단점이 있다.

2.1 Image inpainting

기존의 Image inpainting algorithm은 원 영상에 손상되어진 영역을 개선한 영상정보를 더하고 반복 수행함으로 Image inpainting을 한다.[4]

$$I^{n+1}(i,j) = I^{n}(i,j) + \triangle t I_t^n(i,j), \forall (i,j) \in \Omega \quad (1)$$

식(1)은 다음과 같이 정의한다.

n = inpainting "time"

(i,j) = 픽셀 좌표

$\triangle t$ = 개선율

$I_t^n(i,j)$ = Image $I^n(i,j)$ 의 업데이트

Ω = 제거할 영역(Target Region)

[Table 1]은 Image inpainting을 수식적으로 표현한 것이다.[1]

```
for n=0 to n = Number-of-Iterations
  for each point(pixel)(i)∈ Ω
   compute the smoothness of
        I^n(i) : L^n(i) := I^n_xx(i)
        I^n(i-1) : L^n(i-1) := I^n_xx(i-1)
   compute the smoothness variation in -N:
        I_t^n(i) := L^n(i) - L^n(i-1)
   I^{n+1}(i) = I^n(i) + △t I_t^n(i)
  end
 end
L^n(i) = Laplacian smoothness estimation
```

[Table 1] base inpainting algorithm

2.2 텍스쳐 병합 기반의 Image inpainting

텍스쳐 병합 기반의 Image inpainting 방법은 제거될 영역의 경계에서 우선순위에 따라 일정 크기의 표본을 선택하여 제거될 영역을 제외한 영역에서 표본을 비교하여 오차가 가장 적은 표본을 찾아 제거될 영역에 채우는 방법이다. 텍스쳐 병합 기반의 Image inpainting 과정은 다음과 같다.

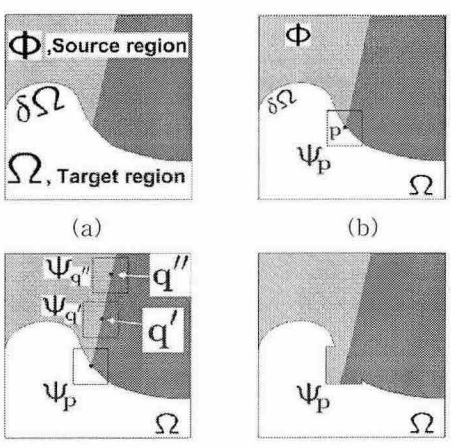


그림 1. inpainting processing

우선 그림 2에서 다음과 같이 정의한다.

Ω : 제거할 영역(Target Region)

δΩ : Ω의 경계

Φ : 제거할 영역의 외부(Source Region)

p : δΩ위의 픽셀

q : Φ위의 픽셀

Ψ_p : p를 중심의 일정크기 표본

Ψ_q : q를 중심의 일정크기 표본

그림 1의 (b)와 같이 제거할 영역의 경계에서 일정 크기의 표본을 우선순위에 따라 선택하고 그림 1의 (c)와 같이 제거할 외부영역 표본과 비교하여 최소 오차를 갖는 표본을 검색하고 제거할 영역만을 채운다.

2.2.1 표본의 우선순위 결정

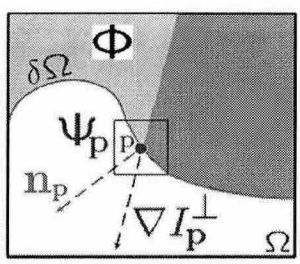


그림 2. inpainting priority

그림 2와 같이 δΩ위의 점 p에서 Ψ_p가 주어졌을 때 우선순위 $P(p)$는 다음과 같은 두 함수의 곱으로 표현된다.

$$P(p) = C(p)D(p) \qquad (2)$$

여기서 $C(p)$는 표본 Ψ_p안에서 Φ영역에 속하는 픽셀들의 포함비율이며 $D(p)$는 p에서의 등조선(isophote)의 강도를 의미한다.

$$C(p) = \frac{\sum C(q)}{|\Psi_p|} \qquad q \in \Psi_p \cap \overline{\Omega} \qquad (3)$$

$$D(p) = \frac{|\nabla I_p^{\perp} \cdot n_p|}{\alpha} \qquad (4)$$

그림 2에서와 같이 $\nabla I_p^{\perp}$는 등조선의 방향, n_p는 점 p에서 δΩ에 대하여 법선벡터이며 α는 정규화 파라미터(Grey-level 영상에서 α는 255)이다.

2.2.2 표본 검색

δΩ위의 모든 점 p에서 우선순위가 결정되면 우선순위가 가장 높은 $\Psi_{\hat{p}}$을 찾아서 영역 Φ에서 $\Psi_{\hat{p}}$와 가장 비슷한 $\Psi_{\hat{q}}$을 검색한다.

$$\Psi_{\hat{q}} = \arg \min d(\Psi_{\hat{p}}, \Psi_{q}) \qquad \Psi_{q} \in \Phi \quad (5)$$

여기서 두 표본 Ψ_{a}, Ψ_{b}의 거리 $d(\Psi_{a}, \Psi_{b})$는 이미 채워져 있는 픽셀들의 SSD(sum of squared differences)로 정의한다. 식 (4)에서 얻어진 $\Psi_{\hat{q}}$은 $\Psi_{\hat{p}}$에 대응하는 픽셀 로 채운다.(단 $\Psi_{\hat{p}}$에서 Ω에 속하는 픽셀만 해당 된다.)

픽셀에 값을 채운 후 2.2.1과 2.2.2과정을 반복하여 영역 Ω는 줄어들게 한다.[5]

Ⅲ. 제안 Algorithm

제안하는 Fast Image inpainting 방법은 인간의 지각 시스템은 큰 명암대비 에지 영역이 아닌 부분에선 약간의 블러링 현상이 생겨도 유연하게 지날 수 있다는 이론에 기초를 두고 있다.

높은 명암대비를 가지는 에지영역에선 사용자가 마스크를 씌어 이방성 확장을 수행하며 그 외의 영역에서는 등방성 필터링을 수행하여 inpainting 처리를 한다.

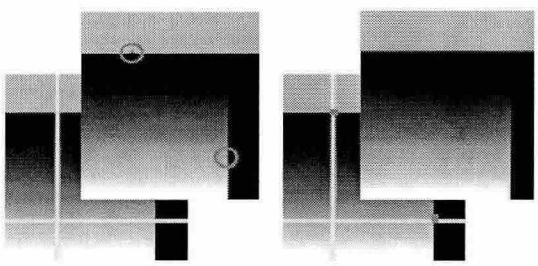


그림 4. 높은 명암대비 에지영역에서의 처리

그림3.의 좌측 그림은 높은 명암대비의 에지영역에서 Image inpainting 을 실행 하였을 때 나타내어지는 블러링 현상을 나타내고 있다. 오른쪽 그림은 높은 명암대비의 에지영역에서 Diffusion barrier를 취하여 블러링 현상을 감소시킨 모습이다.

3.1 등방성 확장 필터링

```
initialize Ω
for (iter =0; iter < num_iteration; iter++)
  convolve masked regions with kernel
```

0.732	0.176	0.732
0.176	0	0.176
0.732	0.176	0.732

0.125	0.125	0.125
0.125	0	0.125
0.125	0.125	0.125

[Table 2] diffusion kernels used with the algorithm

Ω : 제거할 영역(Target Region)

δΩ : Ω의 경계

Ω영역이 작을 경우, inpainting 처리 방향은 δΩ영역에서 Ω까지 확산하는 등방성 필터를 적용하는 방향과 유사해진다. Table 2에서 나타내어진 확장 커널을 사용하여 Diffusion barriers 영역 전까지 수행한다. 마지막 Diffusion barriers의 δΩ 주변 Pixel 값은 Diffusion barriers의 Pixel 값이 사용자가 마스크를 씌운 Pixel 값을 가짐으로 그 자신의 Pixel 값을 가진다.

3.2 High Contrast Edges

inpainting 처리가 끝난후 결과 영상은 높은 명암대비를 가지는 에지영역에서 블러링 현상이 나타난다. 이 블러링 현상을 감소시키기 위해 inpainting 처리가 끝난후 Diffusion barriers(High Contrast Edge의 Ω 내부에서 블러링현상을 최소화시키기 위한 영역)에서 이방성 확장 처리를 한다.

그림 5. Diffusion barriers

Ⅳ. 실험 결과

모든 실험은 Intel Celeron CPU 2.40Ghz, 128MB memory, window XP 환경에서 실험하였다.

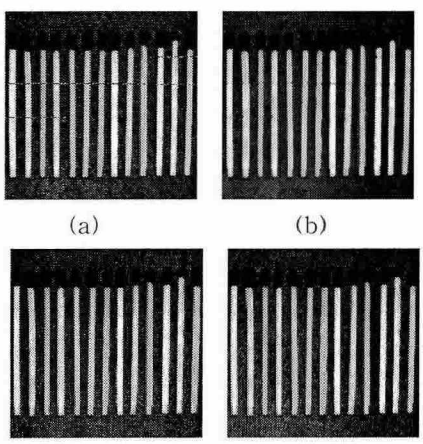
아래의 그림6,7는 각기 다른 크기를 지닌 영상의 실험결과를 보여준다. 그림6,7의 (a)는 원 영상을 나타내며, 그림 6의 (b)는 Median filter를 사용하여 noise만 부분적으로 여러 차례 반복 처리한 결과를 나타내며, 그림 7의 (b)는 영상 전체를 Median filter를 사용하여 여러 번 반복 수행하여 noise를 제거한 결과이다. 그림6,7의(c)는 텍스쳐 병합 기반의 Image inpainting 영상을 나타내며, 각 그림의 (d)는 제안한 방법인 Fast Image inpainting에 의해 noise를 제거한 결과 영상이다.

(a) (b)

(c) (d)

그림6. 250*333 Pixel Image

위의 그림 6에서 나타난 것처럼 부분적인 Median filter를 사용한 경우 noise가 여전히 남아 있음을 확인 할수 있다. Median filter를 사용하여 noise를 제거한 영상에 비해 제안한 Fast Image inpainting 방법으로 noise를 제거한 영상이 나은 성능을 보였음을 확인할 수 있었다. 그림(c),(d)를 살펴보면 텍스쳐 병합기반의 Image inpainting 방법과 제안한 Fast Image inpainting 방법을 사용하였을때 화질의 결과가 유사함을 볼수 있었다.



(a) (b)

(c) (d)

그림7. 800*600 Pixel Image

위의 그림 7의 (b)에서와 같이 noise를 포함한 영상의 전체를 Median filter를 사용하여 처리한 경우, 블러링 현상이 나타나는 문제점을 보였고 Median filtering 처리를 여러 차례 반복해야만 noise가 제거되었다. 그림 (c),(d)를 보면 noise가 큰 명암대비를 가지는 에지영역에 많이 발생하였을 경우 제안한 Fast Image inpainting 방법은 텍스쳐 병합 기반의 Image inpainting을 사용하였을 때 보다 블러링 현상이 없음을 확인할 수 있었다.

처리방법 / 실험데이터	Median Filter	텍스쳐 병합기반	Fast inpainting
그림 6	1.23 s	6.7 s	0.6 s
그림 7	3.6 s	1.4 m	1.5 s

표 1. Median filter와 Inpainting의 처리속도 비교

위의 표에서 나타난 것처럼 Median Filter를 사용할 경우 보다 텍스쳐 병합기반의 Image inpainting 방법으로 noise를 처리했을 경우 연산 수행 시간이 길지만 제안한 Fast Image inpainting 방법으로 nosie를 제거 했을 경우 연산 수행 시간이 단축되는 것을 확인할수 있었다.

V. 결 론

본 연구의 목적인 연산 속도 개선을 위해 Fast Image inpainting을 수행하였다. 사용자는 Inpainting 영역의 마스크 처리와 Diffusion barriers 마스크 처리를 필요로 하며, 나머지 처리는 약간의 시간으로 알고리즘에 의해서 처리가 된다. 처리 속도 시간에서 텍스쳐 병합기반의 Image inpainting 방법보다 연산 수행 시간이 짧으며 Median Filter를 사용할 경우와 비교 하였을 때에도 연산 수행 시간이 짧음을 확인했다. 결과 영상의 화질은 Median filter을 사용하였을 경우보단 블러링 현상이 감소하였지만 텍스쳐 병합 기반의 Image inpainting방법을 사용할 경우보단 블러링 현상이 나타나는 것을 확인했다. 향후 동영상에서의 inpainting 알고리즘을 연구할 계획이다.

참 고 문 헌

[1] M. Bertalmio, G. Sapiro, V. Caselles and C. Ballester. "Image Inpainting." Proceedings of SIGGRAPH 2000, New Orleans, USA, July 2000.

[2] T. Chan and J. Shen, "Mathematical models for local deterministic inpaintings," UCLA CAM TR 00-11, March 2000.

[3] T. Chan and J. Shen, "Local inpainting models and TV inpainting," SIAM J. Appl. Math. 62:3, pp. 1019-1043, 2001.

[4] A. Zorzo Barcelos, A. Batista, M. Martins and Antonio Calos Nogueira "Level Lines Continuation based Digital Inpainting", SIBGRAPI'04

[5] Sung-Gon Kim "Detection of the Optic Disk Boundary in Retinal Images using Inward and Outward Curve Evolution" ,koreacontents '04 Vol. 4 No. 4