Journal of Korea Multimedia Society Vol. 15, No. 8, August 2012(pp. 977-985)

영상 동영상에서의 효율적인 비사실적 렌더링

손태일†, 박경주††

요 약

본 연구에서는 단일영상과 동영상에서의 효율적인 비사실적 렌더링 기법을 제안한다. 단일영상의 경우에는 최근 단일영상 NPR 기법에서 많이 사용되는 플로우 기반 DoG 필터와 Bilateral 필터를 CUDA 환경에서 구현하여 실시간 처리가 가능하게 한다. 또한 동영상의 경우에는 기존의 NPR 동영상 방법인 매 프레임마다 단일영상 NPR 기법을 적용하여 생성하는 방법이 아닌 첫 프레임은 단일영상에 적용되는 NPR기법을 사용하여 스타일화 하고, 다음 프레임부터는 움직임 벡터를 기반으로 한 픽셀 맵핑을 사용하여 이전 프레임에서 움직임이 있는 픽셀의 밝기 값을 다음 프레임의 움직임 벡터 위치로 복사함으로써 불필요한 계산량을 줄이고, 프레임 간의 일관성 또한 유지시키는 방법을 제안한다. 본 연구에서는 실험을 통하여 그 성능을 증명하였다.

Efficient Non-photorealistic Rendering Technique in Single Images and Video

Tae Il Son†, Kyoung Ju Park††

ABSTRACT

The purpose of this study was to present a non-photorealistic rendering technique that is efficient in single images and moving images. In case of single images, they could be processed in a real-time base by realizing flow-based DoG filter and bilateral filter, which have been frequently used in the single image NPR technique recently, in the CUDA environment. In case of moving images, the investigator presented not the existing method of NPR moving images which generating images by applying the single image NPR technique to every frame, but the method of using the single image NPR technique in the first frame and stylizing it, and then of using the motion vector-based pixel mapping in the second frame on and copying the bright values of pixels that move on the frame into the location of next frame's motion vector, thus reducing unnecessary volume of calculation and maintaining the consistency between frames. In this study, the performance of this method was proved via an experiment.

Key words: non-photorealistic rendering(비사실적 렌더링), Realtime(실시간), Motion Estimation(움직임 추정), Abstraction(추상화), Line drawing(선 그리기), CUDA(쿠다)

1. 서 론

NPR(Non-Photorealistic Rendering)은 인간 친화적인 영상이나 동영상을 생성하기 위해 사람이 손으로 그린 듯한 느낌의 영상 생성을 목적으로 한다. 입력된 단일 영상이나 동영상을 사람이 직접 그린 듯한 느낌으로 스타일화 하는 NPR 방법은 주요한 부분을 부각하기 위해 과장을 하거나 사소한 부분을 생략하여 표현한다. 이러한 NPR 방법들은 게임, 애니메이션, CF 등과 같은 엔터테인먼트에 뿐만 아니

※ 교신저자(Corresponding Author) : 박경주, 주소 : 서울시 동작구 흑석동 중앙대학교 305동 604호(156-756), 전화 : 02)820-5823, FAX : 02-820-5497 E-mail : kjpark@cau.ac.kr 접수일 : 2012년 1월 25일, 수정일 : 2012년 6월 2일, 완료일 : 2012년 6월 18일

† 준회원, 중앙대학교 첨단영상대학원 영상학과 석사 (E-mail : sontaeil7@naver.com)
†† 정회원, 중앙대학교 첨단영상대학원 영상학과 교수

라 설계, 과학적 가시화 등 교육 분야에서도 많이 활용된다.

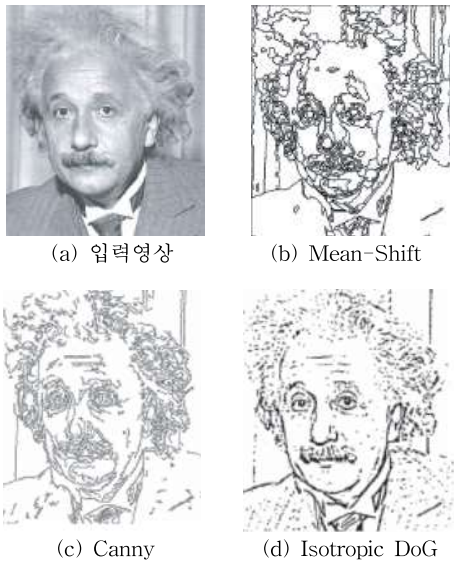
NPR 동영상에서 중요하게 고려되어야 될 요소는 선을 이용하여 프레임 간 객체들(전경과 배경)의 움직임이 발생하는 부분을 손으로 그린 듯한 느낌으로 표현하는 것과 프레임 간 선의 일관성을 유지하는 것이다. NPR 동영상은 연속된 각 프레임마다 단일영상 NPR기법을 적용해서 모든 프레임을 연결하여 생성하는 방법들이 제시 되어 왔으며, 동영상에 특화된 NPR 연구는 활발히 이루어지지 못했다. 또한 과거의 NPR 동영상은 각 프레임마다 단일영상 NPR기법을 적용하기 때문에 계산량이 증가하여 시간이 많이 걸리고, 프레임 간 일관성을 유지하기 어려웠다.

또한 NPR 동영상의 각 프레임에 적용되는 단일영상 NPR 기법에 있어서는 중요한 이슈 중 하나가 영상 내부 물체의 형태를 얼마나 잘 유지하면서 스타일화를 진행할 것이냐는 점이다. 과거에는 등방성(isotropic) 필터에 기반하고 있는 NPR 연구[1-4]가 활발하였다. 등방성 필터는 커널곡선형의 디테일은 사라지고, 계단현상이 발생하기 때문에 이를 보완하기 위해 최근에는 영상의 주요한 요소들을 기반으로 한 플로우 필드 기반의 연구가 제안되어 왔다[5-8]. 또한 양방향 필터(Bilateral Filter)를 사용[9,10]하여 단일영상이나 동영상을 비사실적으로 표현하는 사례가 늘고 있다.

플로우 필드 기반의 NPR 영상은 영상의 주요 외곽선을 따라가는 안정된 방향을 생성하므로, 영상 내부 물체의 형태를 최대한 유지하는 추상화 결과를 가져올 수 있지만, 외곽선 플로우 방향을 알아내기 위해 계산량이 증가하고, 이를 DoG Filter에 반복적으로 적용하기 때문에 시간이 오래 걸린다. 이를 동영상의 각 프레임마다 적용하면 계산량이 더욱 증가하기 때문에 시간이 오래 걸리는 어려움이 있다.

2. 관련연구

과거에는 그림 1과 같이 영상을 비사실적으로 표현하기 위해서 등방성 DoG 필터[11]나 Canny edges [1,3,4]를 이용하여 선을 생성하였다. 또한 Son et al.[12]는 우도 함수(likelihood function)를 계산하여 선을 생성하였다. 이러한 등방성 필터는 곡선형의 디테일은 사라지거나 계단현상이 발생하는 한계점이



(a) 입력영상 (b) Mean-Shift

(c) Canny (d) Isotropic DoG

그림 1. 단일영상에 관한 NPR 기존연구

있다. Kang et al. 은 이를 극복하기 위하여 플로우 기반[5] 이방성 필터를 제안하였다. 하지만 대부분의 선 그리기 방법은 계산량이 많아 실시간 처리가 어렵고, 동영상에 적용하기에는 부적합하다.

동영상의 경우에는 시간적 일관성을 고려하지 않고 렌더링을 하는 것은 매우 위험하다. 시간적 일관성이 고려되지 않을 경우, 동영상은 프레임 간 선의 위치나 모양이 고정되어진 상태로 유지되는 "샤워도어" 현상이나, 프레임 간 깜박거리는 "플리커링(flickering)" 현상이 나타난다. 이를 줄이기 위해 여러 기술들이 개발되었다. Litwinowicz은 각 선의 거리를 조절함으로써 흘을 채웠다[13]. Hertzmann [14,15] and Hays [16]는 소스 영상과 결과 영상간의 색상차이 맵을 사용하였다. Bousseau는 텍스쳐에 대한 일관성을 유지하기 위해 광류의 선(Optical Flow)을 따라는 텍스쳐 어드벡션(texture advection)을 제안하였다[17].

3. 단일영상 NPR 기법

최근 단일영상 NPR 기법은 Bilateral 필터와 플로우 기반 Difference-of -Gaussian(DoG) 필터를 많이 사용한다. 하지만 Bilateral 필터의 시간복잡도는 O(총픽셀수*필터커널사이즈*필터커널사이즈)로 커널사이즈가 커질수록 실시간 처리가 어렵다. 또한 플로

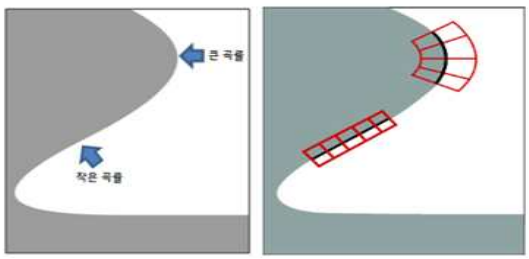
우 기반 DoG 필터의 경우는 외곽선 플로우 방향을 알아내기 위해 계산량이 증가하고, 이를 DoG Filter에 반복적으로 적용하기 때문에 시간이 오래 걸린다.

본 연구에서는 GPU를 이용하여 Bilateral 필터를 병렬처리 할 수 있도록 설계하여 시간복잡도를 줄인다. 또한 플로우 기반 DoG 필터에서 조금 확장하여 곡률 의존 DoG 필터를 사용하여 NPR 단일영상을 생성한다. 마찬가지로 병렬처리 할 수 있도록 설계하여 시간복잡도를 줄인다. 시간복잡도를 줄이기 위하여 GPU 프로그래밍 기법인 CUDA[18]을 사용한다.

3.1 플로우 기반 DoG 필터를 사용한 단일영상 NPR 기법

본 연구에서는 Kang et al.이 제안한 Edge tangent flow(ETF)를 이용하여 플로우 기반 DoG 필터를 확장한 곡률 의존 DoG 필터를 적용하여 NPR 영상을 생성한다.

그림 2와 같이 이 필터의 커널 모양은 ETF를 따르며, 커널의 크기는 필터가 적용되는 지역의 중심픽셀의 곡률의 크기에 의해 정의된다. 곡률의 크기가 크면, 두껍고 짧은 모양의 커널이 적용되며, 곡률의 크기가 작으면 얇고 긴 모양의 커널이 적용된다. 따라서 커널의 크기는 각 픽셀에 따라 모두 변하는 모습을 볼 수 있다. 이렇게 지역에 따라 변하는 커널은 다양한 두께를 가지며, 두께가 급격히 변하는 픽셀에서 갈라지는 효과를 가지는 라인을 생성하게 한다.



(a) (b)

그림 2. (a) ETF (b) 각 픽셀의 곡률의존 DoG

3.2. Bilateral 필터를 사용한 단일영상 NPR 기법

Tomasi 와 Manduchi에 의해 [2]에서 제안된 Bilateral 필터는 영상의 에지를 보존하며 노이즈는 감소시키는 비선형 필터이다. Bilateral 필터는 두 개의 가우시안 필터 즉, 도메인 필터(Domain Filter) 및 레인지 필터(Range Filter)에 의해 동작한다. 영상 주변의 픽셀 값의 차이를 가우시안 가중치를 넣어 이용한 함수의 convolution이다. 이를 표현한 식이 식 (1)이다.

$$BF[I]_p = \frac{1}{W_P}\sum_{q\in s} G_{\delta_s}(\|p-q\|)\,G_{\delta_r}(|I_p-I_q|)\,I_q \qquad (1)$$

여기서 W_P 는 정규화를 위한 식이며,

$$W_P = \sum_{q\in s} G_{\delta_s}(\|p-q\|)\,G_{\delta_r}(|I_p-I_q|) \qquad (2)$$

식 (1)에서 $BF[I]_P$ 와 I_q 는 각각 결과 영상과 입력 영상이다. p는 영상의 해당 화소이자 중앙화소를, q는 그 이웃 화소를 의미하고, s는 전체 화소를 의미한다. $G_{\delta_s}(\|p-q\|)$ 는 도메인 필터로 중앙화소(p)로부터 공간적으로 가까운 화소들에게 가중치를 부여하는 역할을 한다. 즉 필터가 적용되는 위치에서 주변 픽셀 값의 차를 의미한다. $G_{\delta_r}(|I_p-I_q|)$ 는 레인지 필터로 중앙화소 값(I_p)과 유사한 화소들에게 높은 가중치를 부여한다. 즉 필터가 적용되는 위치에서 주변 픽셀 간의 거리를 의미한다. 식 (1)과 (2)에서 δ_s와 δ_r은 각각 도메인 필터와 레인지 필터의 폭을 의미하며 이 두 인자 값을 조정하여 에지 구조를 보존하면서 효율적으로 평활화를 시킬 수 있다.

3.3 CUDA 프로그래밍

본 연구에서는 3.1절의 Bilateral 필터와 3.2절에서 소개한 플로우 기반 DoG 필터를 실시간으로 처리하기 위하여 GPU 프로그래밍 언어인 CUDA를 사용한다. 연산에 필요한 데이터와 알고리즘을 그래픽스 하드웨어에 적합하도록 변환한다. CUDA에서 프로그램 실행 단위는 쓰레드(Thread)이고 다수의 쓰레드를 효율적으로 관리하고 실행하기 위하여 블록(Block), 그리드(Grid)의 기능을 제공한다. 블록은 동일한 프로그램을 동시에 실행하는 쓰레드의 집합이고, 그리드는 이러한 블록을 논리적으로 1차원 , 2차원 및 3차원으로 분류시키는 기능이다. CUDA를 이용한 병렬처리를 극대화하기 위해서는 커널 프로그램에 따라 적절한 블록당 쓰레드 수와 블록의 개수를 제어하여야 한다.

본 논문에서는 그림 3과 같이 256개의 블록과 256개의 쓰레드를 주어 프로그래밍 하였다. 하나의 쓰레드는 하나의 화소를 처리한다. 하나의 블록은 256개

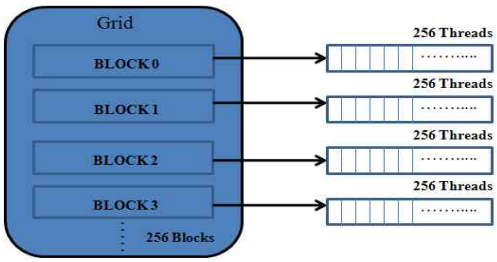


그림 3. 블록당 쓰레드 개수 제어

의 쓰레드를 한 번에 처리한다.

4. 동영상 NPR 기법

기존의 영상 변환 효과와 달리, NPR 기술은 동영상에 적용하가기가 쉽지 않다. 기존의 NPR 동영상은 연속된 각 프레임마다 단일영상 NPR기법을 적용해서 모든 프레임을 연결하여 생성하는 방법들이 제시되어 왔으며, 카툰이나 일러스트레이션 같은 스타일화를 동영상에 적용하면 느린 처리 속도, 반자동화로 인한 수작업, 결과 영상의 튀는 현상이나 플리커링 현상 등 단점들로 인해 사용하기 곤란하였다. 본 연구에서는 불필요한 계산을 줄이고, 프레임 간 일관성을 유지하여 자연스러운 NPR 동영상을 생성하기 위하여 이전 프레임과 현재 프레임의 객체들의 움직임을 추정하고, 움직임이 없는 객체들에 대해서는 이전 프레임에서 연산한 것을 그대로 사용하고, 움직임이 있는 객체들에 대해서는 움직임 정보를 바탕으로 움직인 위치로 픽셀을 맵핑한다.

자연스러운 NPR 동영상을 생성하기 위하여 첫 프레임은 단일영상에 적용되는 NPR기법으로 스타일화 하고, 다음 프레임부터는 움직임 정보을 기반으로 객체(전경과 배경)들을 이동하여 NPR 동영상을 생성한다.

4.1 움직임 추정

움직임 추정은 블록 매칭 알고리즘[19,20]을 사용한다. 블록 매칭 알고리즘에서 부호화(encoding)될 현재 프레임은 겹치지 않는 블록으로 나누어진 뒤, 각각의 현재 블록과 가장 비슷한 블록을 이전 프레임에서 찾게 된다. 주어진 탐색영역안의 가능한 모든 후보 블록을 현재 블록과 비교한다. 식 (3)은 여기에 비교 척도로 사용되는 SAD(sum of absolute differences)이다.

$$SAD(u,v)=\sum_{j=0}^{N-1}\sum_{i=0}^{N-1} I_t(i,j)-I_{t-1}(i+u,j+v) \qquad (3)$$

여기서 $I_t(i,j)$와 $I_{t-1}(i,j)$는 각각 시간 t, t-1에서의 휘도 픽셀 값이다. N은 블록의 크기를 의미하며, (u,v)는 움직임 벡터(Motion Vector)를 의미한다. SAD(u,v)의 최소값을 갖는 움직임 벡터 $MV(\hat{u},\hat{v})$를 식으로 나타내면 다음과 같다.

$$MV(\hat{u},\hat{v})=\arg\min SAD(u,v) \qquad (4)$$

검색 윈도우 안에서 해당 블록과 SAD값이 가장 작은 위치의 블록을 찾고, 그 위치 변화를 움직임 벡터로 인지하게 된다. SAD값을 가장 작게 만드는 변위 (u,v)의 값이 해당 블록의 움직임 벡터이다. $MV(\hat{u},\hat{v})$는 SAD(u,v)의 최소값을 갖는 움직임 벡터이다. 그림 4는 SAD를 이용한 움직임 벡터 검출 결과를 보여주고 있다. 현재 프레임에 존재하는 N×N 크기의 블록이 이전 프레임의 어느 위치에 해당하는지의 여부를 윈도우를 통해 검색한다. 이 검색 윈도우 안에서 해당 블록과 SAD값이 가장 작은 위치를 블록으로 찾고, 그 위치 변화를 움직임 벡터로 추출한다.

블록 검색 방법은 다이아몬드 검색(Diamond Search)[21]을 사용한다. 특정 블록 주변의 검색 변위에 대하여 SAD를 계산하여 움직임 벡터를 추출한다.

그림 4. SAD를 이용한 움직임 벡터 검출

4.2 움직임 정보 기반 픽셀 맵핑

이렇게 움직임 벡터를 알아내면 그림 5와 같이 픽셀 맵핑을 사용하여 픽셀을 복사시킨다.

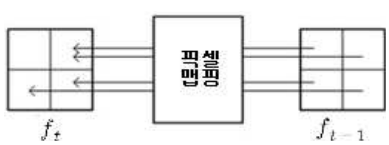


그림 5. 움직임 정보를 기반으로 픽셀 맵핑

픽셀 맵핑은 4.1절에서 설명한 다이아몬드 검색 방법을 사용하여 현재 프레임과 이전 프레임에서의 움직임 벡터를 기반으로 이전 프레임의 픽셀 밝기 값을 다음 프레임의 움직인 위치로 맵핑하는 방법이다.

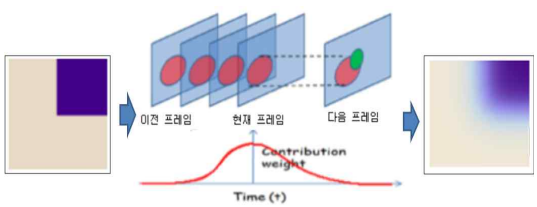
$$I_t(i,j)=\begin{cases} I_{t-1}(i,j) & where\ MV(\hat{u},\hat{v}) < threshold \\ I_{t-1}(i+u,j+v) & where\ MV(\hat{u},\hat{v}) > threshold \end{cases} \quad (5)$$

식(5)에서 t는 시간 상에서의 프레임을 나타낸다. 즉 t는 현재 프레임을 나타내고, t-1은 이전 프레임을 나타낸다. i와 j는 각각 영상의 가로 축과 세로 축의 픽셀 위치이며, u,v는 움직인 픽셀이다. $MV(\hat{u},\hat{v})$는 식(4)에서 얻은 움직임 벡터의 길이이며, threshold는 임의로 지정할 수 있는 어떤 값이다. 본 연구에서는 0으로 지정하였다.

이전 프레임과 현재 프레임에서 움직임이 있는 영역은 움직임 정보를 기반으로 움직인 위치로 맵핑시키고, 움직임이 없는 영역에 대해서는 이전 프레임의 해당 영역을 그대로 맵핑시켜준다.

4.3 시간 축 상의 보간

움직임 정보를 기반으로 픽셀 맵핑을 전체 프레임에 수행을 한 뒤, 마지막으로 시간 축 상에서의 보간을 수행하게 된다. 시간 축 상에서의 보간은 프레임 간의 일관성을 유지시켜준다. 그림 6과 같이 동영상의 인접 프레임 간에 가우시안 스무딩을 적용하면 각이 있고, 딱딱한 객체를 조금 더 부드럽고 매끈한 객체로 바꿀 수 있다.



(a) (b) (c)

그림 6. (a) 입력 영상 (b) 가우시안 스무딩 (c) 출력 영상

식(6)은 가우시안 스무딩을 나타내는 식이다.

$$G(x,y)=\frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (6)$$

x는 2차원 영상에서 가로 축을 나타내며, y는 세로 축을 나타낸다. σ는 가우시안 분포의 표준편차이다. 이는 시간 축 상에서 인접 프레임 간 선의 자연스러운 변화를 가져와 그림 7과 같이 일관성이 유지되고 자연스러운 NPR 동영상을 생성할 수 있게 해준다.

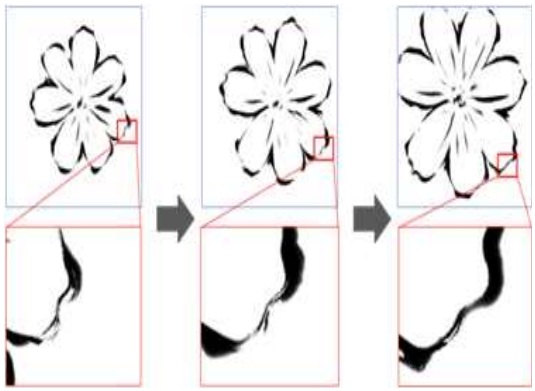


그림 7. 프레임 간의 가우시안 스무딩 적용 결과

5. 성능 실험 및 결과 분석

성능을 비교하기 위하여 기존의 플로우 기반 DoG 필터와 Bilateral 필터를 이용한 선 스타일화 방법과 CUDA를 이용한 방법의 수행시간을 비교하였다. 실험을 위하여 CPU는 Intel i5 760 2.80 GHz, GPU는 NVIDIA 9800GT를 사용하였다.

표 1. 플로우 기반 DoG 필터 수행시간 비교

영상 해상도(픽셀)	기존 방법(초)	CUDA(초)
1024×1024	26.4	0.08
768×512	10.0	0.06
512×512	6.6	0.04
256×256	1.6	0.009

표 1은 CPU에서 구현된 플로우 기반 DoG 필터와 CUDA를 이용하여 구현한 플로우 기반 DoG 필터의 수행시간을 비교한 것이다. 기존 방법보다 약 100~120배 정도 속도가 향상된 것을 볼 수 있다. 기존 방법으로 수행할 때는 영상의 크기가 2배 커질 때마다 전체 수행시간은 4배 이상 늘어나는 반면, 본 연구의 제안된 방법은 영상의 크기가 커져도 수행시간은 거

표 2. Bilateral 필터 수행시간 비교

Processing type	수행시간 (초)			
	Param1	Param2	Param1	Param2
	3	10	7	10
CPU	2.35		8.12	
CUDA	0.02		0.08	

의 차이를 보이지 않는다.

표 2는 CPU에서 구현된 Bilateral 필터와 CUDA를 이용하여 구현한 Bilateral 필터의 수행시간을 비교한 것이다. 성능비교를 위하여 512×512사이즈의 24-bit 영상을 사용한다. Bilateral 필터는 두 개의 파라미터를 사용한다. 파라미터1은 색상 도메인에서 사용되는 가우시안 커널의 너비를 나타낸다. 파라미터1 값이 클수록 더 넓은 분포의 밝기값(또는 색상값)이 평탄화 된다. 파라미터2는 공간 도메인에서 사용되는 가우시안 커널의 너비를 나타낸다. 이는 가우시안 필터에서 시그마 값과 유사하다. 이 파라미터 값이 커질수록 수행시간은 급격히 느려진다. 하지만 CUDA로 구현하였을 경우에는 크게 차이가 나지 않고, 수행시간이 굉장히 줄어든 것을 볼 수 있다.

표 3. 기존 NPR 동영상 기법과의 수행시간 비교

영상 해상도 (픽셀)	영상 길이 (초)	초당 프레임 (fps)	매 프레임 마다 NPR 적용한 결과 (초)	움직임 추정기반 NPR 적용한 결과 (초)
1024×1024	5.04	24	1920	10.3
768×512	5.04	24	1344	8.6
512×512	5.04	24	960	7.1
256×256	5.04	24	748	6.1

표 3은 매 프레임마다 단일영상 NPR 기법을 적용하여 NPR 동영상을 만드는 기존의 방법과 움직임 추정기반 NPR 적용한 방법의 수행시간을 비교한 것이다. 기존의 방법은 곡률의존 DoG 필터를 매 프레임마다 적용하기 때문에 512×512 동영상 기준으로 960초의 수행시간이 걸린다. 하지만 본 연구의 방법을 이용하면 첫 프레임만 스타일화 하고, 나머지 프레임은 움직임 추정 기반으로 픽셀 맵핑을 하기 때문에 불필요한 계산을 줄이며 보다 효율적이고 빠르게 NPR 동영상을 생성할 수 있다.

그림 8은 기존의 다른 DoG에 관한 연구 결과들과의 비교 영상이다. 그림 9는 Bilateral 필터를 적용한 결과이다. 반복 횟수에 따라서 다른 영상을 얻을 수 있다. 그림 10은 Bousseau의 Texture Advection을 사용하여 동영상을 수채화로 추상화한 연구[17]과 본 연구의 방법으로 추상화 한 결과를 비교하였다. Bousseau는 텍스쳐에 대한 일관성을 유지하기 위해 광류의 선(Optical Flow)을 따라는 텍스쳐 어드벡션(texture advection)을 제안하였지만, 한 프레임 당 어드벡션(Advection)을 구하는데 5초의 시간이 걸린다. 하지만 본 연구의 방법을 사용한 움직임 정보를 기반으로 한 픽셀 맵핑을 사용하면 한 프레임 당 0.2초의 시간이 걸린다.

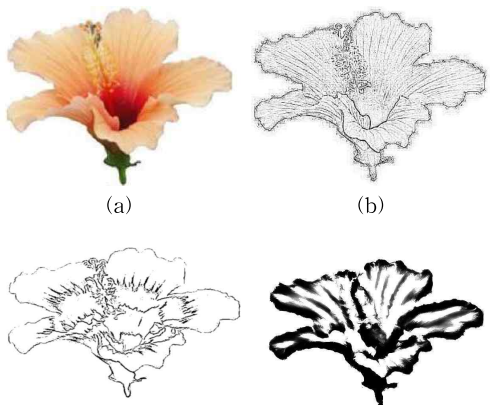


그림 8. 단일영상 NPR기법 비교 2 (a) 입력영상 (b) Isotropic DoG (c) Flow-based DoG (d) 본 연구에서 CUDA로 구현한 CDoG

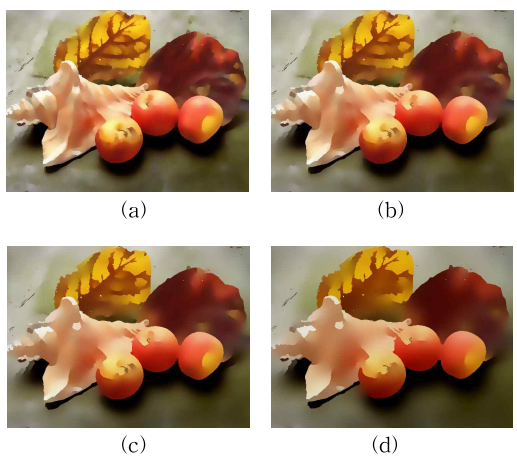


그림 9. CUDA로 구현한 Bilateral 필터 (a) 입력영상 (b) Bilateral 필터 5번 반복 (c) Bilateral 필터 20번 반복 (d) Bilateral 필터 40번 반복

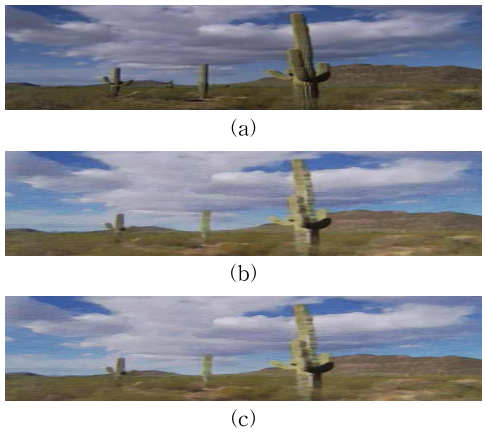


그림 10. 동영상 NPR 연구결과 비교 (a) 원본 영상의 한프레임 (b) 텍스쳐 어드벡션 기반 동영상 NPR[15]. (c) 움직임 추정기반 동영상 NPR

그림 11,12는 동영상 입력을 매 프레임 마다 NPR 기법을 적용하여 추상화한 결과와 움직인 기반을 사용한 추상화 한 본 연구의 결과를 비교하였다. 매 프레임에 NPR기법을 적용하여 선을 스타일화 하면 플리커링을 유발하는 영역이 생기지만 본 논문의 연구 방법을 사용하면 보다 자연스럽고 일관성이 유지된 동영상을 생성할 수 있다. 그림 13은 논문 이외의 상용툴인 포토샵의 플러그인으로 사용되는 Toonit를 이용하여 동영상의 첫 프레임만을 생성한 뒤, 본 논문의 방법을 사용하여 생성하였더니 Toonit을 이용하여 생성한 동영상과 비슷한 효과를 낼 수 있었다.

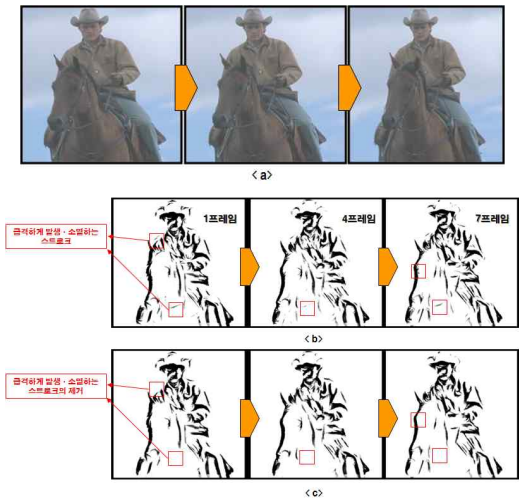


그림 11. 움직임 추정 기반 동영상 NPR 의 플리커링 제거효과 1 (a) 원본 영상 (b) 매 프레임 NPR기법 적용한 결과 (c) 본 연구의 결과

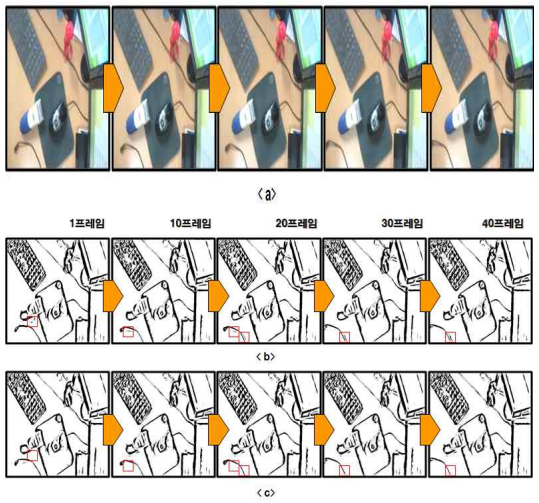


그림 12. 움직임 추정 기반 동영상 NPR 의 플리커링 제거효과 2 (a) 원본 영상 (b) 매 프레임 NPR기법 적용한 결과 (c) 본 연구의 결과

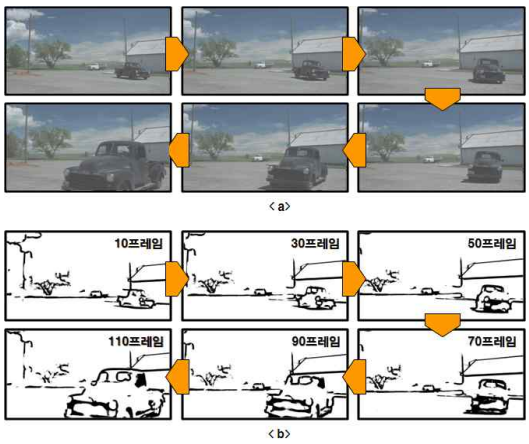


그림 13. 움직임 추정 기반 동영상 NPR 결과 (a) 원본 영상 (b) 본 연구의 결과

6. 결 론

본 연구에서는 단일영상의 경우에는 최근 단일영상 NPR 기법에서 많이 사용되는 플로우 기반 DoG 필터와 Bilateral 필터를 CUDA 환경에서 구현하였다. 또한 동영상의 경우에는 기존의 NPR 동영상 방법인 매 프레임마다 단일영상 NPR 기법을 적용하여 생성하는 방법이 아닌 첫 프레임은 단일영상에 적용되는 NPR기법을 사용하여 스타일화 하고, 다음 프레임부터는 움직임 벡터를 기반으로 한 픽셀 맵핑을 사용하여 이전 프레임에서 움직임이 있는 픽셀의 밝

기 값을 다음 프레임의 움직임 벡터 위치로 복사함으로써 불필요한 계산량을 줄였고, 프레임 간의 일관성 또한 유지시켰다.

본 연구에서의 실험결과는 첫 프레임은 ETF를 이용하여 플로우 필드를 생성하고, 플로우 필드를 기반으로 곡률의존 DoG를 적용하여 선의 다양한 두께와 갈라짐을 표현한 NPR기법을 사용하여 스타일화 하였고, 계산량이 많은 NPR기법을 병렬처리 함으로써 실시간 처리가 가능하게 하였다. 실험을 통하여, 그 성능을 증명하였다.

참 고 문 헌

[1] J. Canny, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* PAMI-8, Issue 6, pp. 679-698, 1986.

[2] C. Tomasi and R. Manduchi, "Bilateral Filtering for Gray and Color Images," *Proc. of the 1998 IEEE International Conference on Computer Vision*. pp. 839-846, 1998.

[3] D. Decarlo and A. Santella, "Stylization and Abstraction of Photographs." *ACM TOG (Proceedings of SIGGRAPH 2002)*, pp. 769-776, 2002.

[4] A. Orzan, A. Bousseau, P. Barla, and J. Thollot, "Structure-preserving Manipulation of Photographs," *Proc. Non-Photorealistic Animation and Rendering*, pp. 103-110, 2007.

[5] H. Kang, S. Lee, and C. Chui, "Coherent Line Drawing," *Proc. NPAR (2007)*, pp. 43-50. 2007.

[6] H. Kang, S. Lee, and C. Chui, "Flow-Based Image Abstraction," *IEEE Transactions on Visualization and Computer Graphics,* Vol. 15, Issue 1, pp. 62-76, 2009.

[7] H. Kang, and S. Lee, "Shape-Simplifying Image Abstraction," *Computer Graphics Forum,* Vol. 27, Issue 7, pp. 1773-1780, 2008.

[8] J. E. Kyprianidis, and J. Dollner, "Image Abstraction by Structure Adaptive Filtering," *Proc. EG UK Theory and Practice of Computer Graphics, Eurographics Association*, pp. 51-58. 2008.

[9] H. Kang, C. Chui, and U. Chakraborty, "A Unified Scheme for Adaptive Stroke-based Rendering," *The Visual Computer*, Vol. 2, No. 9, pp. 814-824, 2006.

[10] H. Winnemöller, S. C. Olsen, and B. Gooch, "Real-Time Video Abstraction," *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, pp. 1221-1226, 2006.

[11] B. Gooch, E. Reinhard, and A. Gooch, "Human Facial Illustrations: Creation and Psychophysical evaluation," *ACM Trans. Graph.,* Vol. 23, Issue 1, pp. 27-44, 2004.

[12] M. Son, H. Kang, and Y. Lee, "Abstract Line Drawings from 2D Images," *Proc. of the IEEE Pacific Conference on Computer Graphics and Applications (PG'07)*, pp. 333-342. 2007.

[13] P. Litwinowicz, "Processing Images and Video for an Impressionist Effect," *SIGGRAPH'97*, pp. 407-414, 1997.

[14] A. Hertzmann, and K. Perlin "Painterly Rendering for Video and Interaction," *NPAR'2000*, pp. 7-12, 2000.

[15] A. Hertzmann, "Fast Paint Texture," *NPAR'2000*, pp. 91-96, 2002.

[16] J. Hays, and I. Essa, "Image and Video Based Painterly Animation," *NPAR'2004*, pp. 113-120, 2004.

[17] A. Bousseau, F. Neyret, J. Thollot, and D. Salesin, "Video Watercolorization using Bidirectional Texture Advection," *ACM Transactions on Graphics*, Vol. 26, No. 3, pp 104, 2007.

[18] 계희원, 김준호, "GPGPU 환경에서 최대휘소투영 렌더링의 고속화 방법," 멀티미디어학회논문지, 제14권, 제8호, pp.981-991, 2011.

[19] Jianhua Lu and Ming L. Liou, "A Simple and Efficent Search Algorithm for Block-Matching Motion Estimation," *IEEE Trans. Circuits*

And Systems For Video Technology, Vol. 7, No. 2, pp. 429-433, 1997.

[20] 박성모, 유태경, 정용재, 문광석, 김종남, "움직임 벡터의 분포와 적응적인 탐색 패턴 및 매칭기준을 이용한 유사 무손실 고속 움직임 예측 알고리즘," 멀티미디어학회논문지, 제13권, 제7호, pp. 991-999, 2010.

[21] Shan Zhu and Kai-Kuang Ma, "A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation," *IEEE Transactions on Image Processing*, Vol. 9, No. 2, PP. 292-296, 2000.

손 태 일

2010년 한신대학교 컴퓨터학과 학사

2012년 중앙대학교 첨단영상대학원 영상학과 석사

관심분야 : Computer Graphics, Image processing, Non-Photo Realistic, Time Rendering

박 경 주

1997년 이화여자대학교 컴퓨터공학 학사

2000년 University of Pennsylvania 컴퓨터정보과학 석사

2005년 University of Pennsylvania 컴퓨터정보과학 박사

2007년~현재 중앙대학교 첨단영상대학원 영상학과 교수

관심분야: Physics Based Simulation, Deformable Models, Image/Video