CUDA 를 이용한 실시간 Free Viewpoint TV System 구현

양윤모, 이진혁, 오병태
한국항공대학교
yym064@naver.com, linke01@naver.com, byungoh@kau.ac.kr

Real-Time Free Viewpoint TV System Using CUDA

Yun Mo Yang, Jin Hyeok Lee, Byung Tae Oh
Korea Aerospace University

요 약

본 논문에서는 여러 대의 Microsoft Kinect 와 NVidia 사의 GPGPU 라이브러리 CUDA 를 사용하여 실시간 Free Viewpoint TV System 을 제안한다. Kinect 로부터 얻어진 컬러 및 깊이 영상을 통하여 두 카메라 사이의 가상시점에서 영상을 실시간으로 출력하는 시스템을 설계한다. 이 과정에서 많은 연산량을 요구하는 좌표계 변환 과정과 IR 패턴의 간섭문제를 해결하기 위해 사용되는 Nearest Neighbor 홀 채움 방식을 CUDA 를 이용해 병렬화시켰다. 실험 결과 CUDA 를 이용해 구성한 시스템이 기존의 CPU 만을 이용해 구성한 시스템보다 같은 시간 동안 더 많은 합성영상을 만들 수 있었다.

In this paper, we propose the Real-Time Free Viewpoint TV System with multiple Microsoft Kinects and CUDA of NVidia GPGPU library. It generates a virtual view between two views by using color and depth image acquired by Kinect in real time. In order to reduce complexity of coordinate transformations and nearest neighbor method for hole filling caused by IR pattern interference, we parallelize this process using CUDA. Finally, it is observed that CUDA based system generates more frames than using CPU based system in the same time.

1. 서론

방송기술의 발달함에 따라 기존의 2D방송을 넘어 3D방송 기술의 대한 수요가 늘어나고 있는 추세이다. 현재 가장 대중적으로 사용되는 3D방송기술로는 스테레오스코픽(Stereoscopic) 방식이 있다. [1]

스테레오스코픽 방식은 양쪽 눈의 시각 차이를 이용하는 방식이다. 시차가 있는 한 쌍의 2D 영상을 시청자의 양쪽 눈에 각각 제시하여 영상에 입체감을 지각할 수 있게 해주는 방식이다. 하지만 이 방식은 스테레오스코픽 영상에 특화된 안경을 착용해야 되고 시점의 불일치 문제로 시각 피로, 두통, 어지러움 등의 문제를 유발한다.

따라서 스테레오스코픽 방식의 한계와 문제점을 보완하기 위해 제안된 방식이 다시점 영상(Multiview)방식이다. 다시점 영상 방식은 여러 시점의 영상을 합성하여 만드는 영상으로 임의시점의 영상을 시청할 수 있다. Free Viewpoint TV(FTV)[2]는 다시점 영상방식의 한 종류로 임의의 시점 영상을 자유롭게 제공해 줄 수 있다.

하지만 FTV 방식은 임의의 시점을 합성해서 제공해주기 때문에 실시간으로 사용되기에는 많은 연산량이 요구 된다. 이를 해결하기 위해 최근에 화제가 되고 있는 GPGPU(General Purpose Graphic Processing Units)기술을 사용하여 연산량 문제를 해결할 수 있다.

본 논문에서는 기존에 제한하였던 FTV 시스템 [9]의 실시간 구현을 위하여 GPGPU의 기술 중 하나인 NVidia사의 CUDA를 사용하여 FTV에서 요구되는 연산량 문제를 해결하여 실시간 FTV system의 고속화 방법을 제안한다.

2. 시스템 구성

본 논문에서는 2대의 Kinect를 사용하였다. 카메라는 Parallel하게 위치시키는 것이 아닌, 그림 1과 같이 각각의 카메라가 물체를 바라보게 하여 물체에 초점이 수렴이 되도록

하는 방향으로 좌, 우에 위치시켰다. 이 때, 각각의 카메라는 같은 높이에 위치시키고, 틀어져 있는 각도는 20° 에서 60° 사이로 맞춰놓았다. 이는 사용자가 원하는 각도로 카메라를 위치시킬 수 있도록 하기 위함이다. Kinect 카메라가 수평을 유지하도록 하는 것이 매우 중요하다. 서로 수평을 유지하고, 물체를 바라볼 경우에, 쉽고 빠르게 좌표계를 변화시킬 수 있기 때문이다. 만약 x, y, z축 모두 틀어져 있는 상황이라면 각각의 각도를 고려하여야 한다. 하지만 카메라의 수직방향, 즉 카메라의 지면 수직방향인, y축만 틀어져 있다고 한다면, 계산은 간단해진다. 자세한 식은 다음 장에서 논의한다.

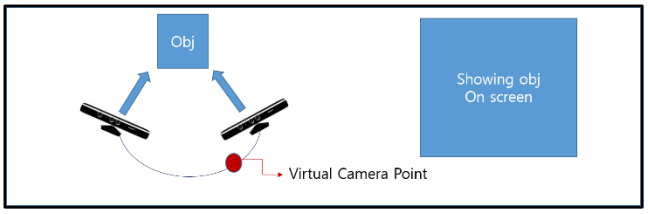

Fig. 1. 시스템 구성도

3. FTV Algorithm

각각의 Kinect 들은 Calibration 을 통해 내부 파라미터를 구한다(K_{in}), 그 후, Epipolar Geometry 를 통하여, 카메라간의 기하학적 관계를 구한다. 각각의 카메라에서 나온 영상들에서 특징점들을 찾고, 그것들을 이용하여 Fundamental Matrix 를 구한다. 그 후, 각각의 카메라 내부 파라미터를 통하여 Essential Matrix 를 구한다.

$$E = K_{in\,1} F K_{in\,2} \tag{1}$$

Essential matrix 에서 Rotation matrix 를 추출할 때의 잘 알려진 방법은 SVD decomposition 이다. $E = UDV^T$ 를 이용하여 아래의 식들을 만들 수 있다 [3-6].

$$\mathrm{R}_1 = UWV^T,\ \mathrm{R}_2 = UW^TV^T \text{ with } \mathrm{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

가상시점의 영상을 만들기 위해서는 우선, 두 카메라에서 얻은 Depth 영상의 정보를 이용하여 좌표계 변환이 필요하다. Pixel 좌표를 World 좌표로 변환시키고, Calibration 을 통해 얻은 카메라 내부 파라미터와 두 카메라간의 각도를 통하여 가상시점으로 point 들을 옮긴다.

Translation Model 은 카메라가 가상의 원 위에 위치해 있고, 카메라들 사이의 가상의 원 위에서 가상시점을 선택한다는 가정하에 아래와 같은 모델로 설정하였다.

$$\mathrm{R} = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix},\quad \mathrm{T} = \begin{bmatrix} -d\sin(\theta) \\ 0 \\ d\,(1-\cos(\theta)) \end{bmatrix} \tag{3}$$

$$M_{ex} = \begin{bmatrix} cos(\theta) & 0 & sin(\theta) & -dsin(\theta) \\ 0 & 1 & 0 & 0 \\ -sin(\theta) & 0 & cos(\theta) & d(1-cos(\theta)) \end{bmatrix} \tag{4}$$

$$\begin{bmatrix} \mathrm{xh} \\ yh \\ w \end{bmatrix} = K_{in} \begin{bmatrix} wx \\ wy \\ wz \end{bmatrix},\quad \begin{bmatrix} wx \\ wy \\ wz \end{bmatrix} = {K_{in}}^{-1} \begin{bmatrix} \mathrm{xh} \\ yh \\ w \end{bmatrix}$$
$$,\ where\ \ wx, wy, wz = world\ coordinates \tag{5}$$

$$x_{im} = xh/w\,,\qquad y_{im} = yh/w\,,\qquad \mathrm{w} = \mathrm{wz}$$
$$,where\ \ x_{im}, y_{im} = pixel\ coordinates \tag{6}$$

식 (5)-(6)을 이용하여, Pixel 좌표를 World 좌표로 변환시키고, 식 (4)를 이용하여, 새로운 가상의 지점의 좌표로 변환을 시킨다. 그 후, 다시 World 좌표를 Pixel 좌표로 변환시킨다.

위 좌표계 변환 과정을 각 카메라에서 얻은 영상의 픽셀마다 적용하기 때문에 많은 연산량을 요구한다. 따라서 이를 CUDA 를 이용해 병렬화 시켰다. 먼저 카메라에서 얻은 컬러 및 깊이 영상을 GPU 로 복사한 후 한 픽셀당 하나의 스레드(Thread)를 할당하여 위의 좌표계 변환 과정을 수행하였다. 두 대 이상의 Kinect 를 사용하게 되면, 각각에서 나오는 IR 패턴이 간섭을 일으켜 홀을 발생시킨다.[8] 이는 불확실한 깊이 값을 제공하여, 좌표 변환과 3D Reconstruction 을 수행하는 과정에서 문제가 된다. 따라서, 이렇게 발생한 홀을 채우기 위해 Nearest Neighbor(이하 NN) 방식과 Inpainting 기법을 사용하였다 [7].

이때, NN 방식을 통한 홀 채움 과정에서 홀이 발생한 픽셀의 주변의 픽셀 값을 모두 비교해야 하기 때문에 많은 비교연산이 들어간다. 따라서 각 픽셀 마다 하나의 스레드를 할당하고 각 스레드에서 비교과정을 수행하게 하였다. 또한 추가적인 속도향상을 위해 Inpainting 기법은 깊이 영상의 사이즈를 줄이고 적용한 후 다시 사이즈를 늘려 기존의 Inpainting 기법 보다 시간을 크게 단축하였다.

4. 결과

실험 결과 아래 표의 1 과 같이 CUDA 를 이용하여 구현한 시스템이 CPU 만 이용해 구성한 시스템보다 약 5 배 빠른 속도향상을 이루게 되었다. 특히 좌표계 변환과 NN 기법은 주변의 픽셀과 관련이 없이 각각의 픽셀마다 개별적으로 적용할 수 있기에 병렬화가 용이하여 크게 시간을 단축할 수 있었다. 최종적으로 위의 과정을 통해 아래의 Fig. 2 와 같은 결과 영상을 얻게 되었다.

표 1. CPU 와 GPU 사용시 속도 비교

	CPU	GPU
평균 fps	1.2 fps	6 fps
평균 연산시간	0.833 sec	0.1666 sec

Fig. 2. CUDA 를 이용해 중간위치의 합성영상

5. 결론

제안 시스템을 통해 2 대의 Kinect 와 CUDA 를 사용하여 임의의 시점의 영상을 고속으로 합성하는 시스템을 개발하였다. 하지만, 여전히 많은 연상량을 요구하는 Inpainting 기법의 경우 병렬화가 필요하다. 또한 깊이 영상과 컬러 영상의 경계불일치 문제로 경계부근에 오차가 발생한다.

추후에는 Inpainting 기법의 병렬화 연구와 컬러영상과 깊이 영상의 경계불일치 문제를 해결하는 연구 및 두 시점의 영상을 정합하는 연구를 진행할 예정이다.

감사의 글

이 논문은 2013 년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (NRF-2013R1A1A1057779).

참조문헌

[1] 윤국진, 엄기문, 김진웅, 이광순, 허남호, *"3DTV 기술 동향"* TTA Journal, No. 122, pp. 92-97, March - April. 2009.

[2] A. Kubota, A. Smolic, M. Magnor, M. Tanimoto, T. Chen, and C. Zhang, *"Multiview Imaging and 3DTV"*, IEEE Signal Processing Magazine, pp. 10-21, Nov. 2007

[3] G. Slabaugh, "Computing Euler angles from a rotation matrix", Technical Report, http://www.gregslabaugh.name/publications, 1999.

[4] Li Ling, Eva Cheng, I. S. Burnett, *"Eight solutions of the essential matrix for continuous camera motion tracking in video augmented reality"*, ICME, 2001 IEEE International Conference on, July. 2011.

[5] R. I. Hartley, *"Estimation of relative camera positions for uncalibrated cameras"*, Computer Vision-ECCV'92, Lecture Notes in Computer Science Volume 588, pp. 579-587, May. 1992

[6] W. Wang, H. T. Tsui, *"A SVD decomposition of essential matrix with eight solutions for the relative positions of two perspective cameras"*, Proceedings. 15th International Conference on Pattern Recognition, Vol. 1, pp. 362-365, Sept. 2000.

[7] A. Telea, *"An image inpainting technique based on the fast marching method"*, Journal of Graphics Tools, Vol. 9, No. 1, pp. 25-36, 2004.

[8] C. Limin, C. Mingyu, X. Shizhe, L. Yin, *"Analysis of interference between multiple Kinect sensors and a noise reduction method for mobile robots"*, JCIT, Vol. 7, No. 21, Nov. 2012.

[9] 이준협, 양윤모, 오병태, "MS Kinect 를 이용한 Free Viewpoint TV System 설계", 2015 년도 한국방송공학회 하계 학술대회