특집논문 (Special Paper)
방송공학회논문지 제17권 제6호, 2012년 11월 (JBE Vol. 17, No. 6, November 2012)
http://dx.doi.org/10.5909/JBE.2012.17.6.945
ISSN 1226-7953(Print)

효율적인 멀티 쓰레딩을 이용한 고해상도 깊이지도의 실시간 획득

조 칠 석[a], 전 지 인[a], 추 현 곤[b], 박 종 일[a]‡

High Resolution Depth-map Estimation in Real-time using Efficient Multi-threading

Chil-Suk Cho[a], Ji-In Jun[a], Hyon-Gon Choo[b], and Jong-Il Park[a]‡

요 약

깊이지도를 구하는 방법 중 많이 사용되어지는 방법으로 줄무늬 패턴을 이용하는 방법이 존재한다. 이 방법은 프로젝터-카메라 시스템(Pro-Cam System)을 이용하며 프로젝터로 조사한 패턴을 카메라로 촬영하여 원래의 패턴과 촬영된 패턴간의 기하학적인 관계를 구하여 깊이지도를 구하는 방법이다. 본 논문에서는 이와 같이 구조광을 이용하는 깊이지도 획득 시스템에서 효과적으로 멀티 쓰레드를 사용하여 실시간 처리하는 것을 제안한다. 일반적으로 자주 사용되는 멀티 쓰레딩에는 CPU의 쓰레드를 이용하는 OpenMP와 GPU의 쓰레드를 이용하는 CUDA가 있다. 이 두 가지 기법은 수행하는데 차이점이 존재하기 때문에 상황에 따라 OpenMP가 더 좋은 효율을 보이는 부분이 있고 CUDA가 더 좋은 효율을 보이는 부분이 있다. 따라서 본 논문에서는 이 두 가지에 대해서 각 부분의 특성에 맞게 더 좋은 효율을 보이는 멀티 쓰레드를 적용하였다. 결과적으로 제안된 방법은 1280×800의 영상에 대해 25fps 이상의 깊이지도를 획득할 수 있었다.

Abstract

A depth map can be obtained by projecting/capturing patterns of stripes using a projector-camera system and analyzing the geometric relationship between the projected patterns and the captured patterns. This is usually called structured light technique. In this paper, we propose a new multi-threading scheme for accelerating a conventional structured light technique. On CPUs and GPUs, multi-threading can be implemented by using OpenMP and CUDA, respectively. However, the problem is that their performance changes according to the computational conditions of partial processes of a structured light technique. In other words, OpenMP (using multiple CPUs) outperformed CUDA (using multiple GPUs) in partial processes such as pattern decoding and depth estimation. In contrast, CUDA outperformed OpenMP in partial processes such as rectification and pattern segmentation. Therefore, we carefully analyze the computational conditions where each outperforms the other and do use the better one in the related conditions. As a result, the proposed method can estimate a depth map in a speed of over 25 fps on 1280´ 800 images.

Keyword : depth-map estimation, multi-thread, real-time, high-resolution depth-map, structured light technique

a) 한양대학교(Hanyang University)
b) 한국전자통신연구원(Electronics and Telecommunication Research Institute)
‡ Corresponding Author : 박종일 (Jong-Il Park)
E-mail: jipark@hanyang.ac.kr
Tel: +82-2-2220-4368 Fax: +82-2-2299-7820
※본 연구는 지식 경제부 및 한국산업평가관리원의 산업융합원천기술개발사업(정보통신)의 일환으로 수행하였음. [KI002058, 대화형 디지털 홀로그램 통합서비스 시스템의 구현을 위한 신호 처리 요소 기술 및 SoC 개발]
※본 연구는 방송통신위원회 및 한국방송통신전파진흥원의 방송통신미디어 원천기술개발 과제의 일환으로 수행되었음. [10912-02001, 지상파 양안식 3DTV 방송시스템기술개발 및 표준화]
· Manuscript received September 10, 2012 Revised November 6, 2012 Accepted November 19, 2012

Ⅰ. 서 론

3D 영화를 통해 생긴 많은 관심은 3D 기술로 이어지고 있다. 이중 본 논문에서는 3D 방송에 대해 주목하였다. 3D 기술의 핵심요소 중에는 깊이지도(depth map)가 있다. 일반적으로 다뤄지는 3D 기술과는 다르게 원활한 3D 방송을 위해서는 높은 해상도의 영상에 대해 실시간으로 깊이지도를 획득하고 이를 적용하는 기술이 필수적이다. 하지만 기존의 3D 깊이 지도를 획득하는 연구들은 낮은 해상도를 가지거나 실시간으로 다루지 못하는 한계가 있었다[1-8]. 이를 해결하기 위해 높은 해상도의 영상에서 깊이지도를 획득하는 과정에서 멀티 쓰레드(multi-thread)를 활용하는 기법인 CUDA(Compute Unified Device Architecture)와 OpenMP (Open Multi-Processing)를 효과적으로 적용하여 실시간으로 처리 하였다.

구조광을 이용한 깊이지도 획득 기술은 이전부터 많은 연구가 이루어졌던 방법으로 고해상도의 깊이 지도를 얻을 수 있는 방법들을 제안하고 있다. 이는 크게 네 가지로 분류할 수 있으며 여기에는 black&white 패턴[1], gray scale 줄무늬 패턴[2,3], 컬러 패턴[4,5], 컬러 줄무늬 패턴[6-9]을 이용한 방법이 존재 한다. 본 논문에서는 de Bruijn 시퀀스(sequence)를 기반으로 한 컬러 줄무늬 패턴을 이용한 [8]의 방법을 사용하였다. 높은 해상도의 영상을 사용함에 따라 계산에 소모되는 시간이 증가 할 수밖에 없기 때문에 알고리즘의 개선만으로는 한계가 있다. 때문에 본 논문에서는 전체 연산을 각 부분으로 나누어 각 연산중 반복되어 수행되는 작업들에 대하여 각 부분에 맞게 최적화된 멀티 쓰레딩(multi-threading)을 통한 고속화를 제안하였다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 본 논문에서 사용한 깊이지도 획득 시스템에 대하여 설명하고, Ⅲ장에서는 각각의 연산들이 수행하는 작업과 두개의 기법의 비교를 통하여 해당하는 최적화된 멀티 쓰레딩을 제안한다. Ⅳ장에서는 이를 구현한 실험 결과를 보이며 제안한 방법의 타당성을 증명한다. 마지막으로 Ⅴ장에서 결론과 앞으로의 연구방향을 정리 한다.

Ⅱ. 깊이지도 획득 시스템

그림 1은 본 논문에서 사용한 구조광을 이용한 깊이지도 획득 시스템의 구성도이다. 이 기술은 먼저 프로젝터를 통해 패턴 영상을 깊이지도 획득을 원하는 장면에 조사한다. 이후 해당하는 장면을 카메라로 촬영한다. 이때 해당 장면의 깊이의 변화에 따라 프로젝터로 조사한 패턴과 카메라로 촬영된 영상에서의 패턴이 다른 모습을 나타내게 된다. 이 두 패턴의 관계를 통하여 장면의 3차원 위치를 추정하는 것이 구조광을 이용한 깊이지도 획득 시스템의 핵심이다. 이때 사용되어지는 패턴의 종류는 여러가지 존재하지만 두 가지로 분류하자면 black&white 패턴과 gray scale이나 컬러 패턴으로 나눌 수 있다. 이 분류의 기준은 동적인 물체에 대한 깊이지도 획득의 가능 여부이다. 먼저 black&white 패턴은 이진패턴으로 실험방법이 간단하고 쉽게 고해상도의 깊이지도를 얻을 수 있는 장점이 있다. 하지만 한 장면에 대해서 연속적으로 많은 패턴을 조사해야 한다는 단점 때

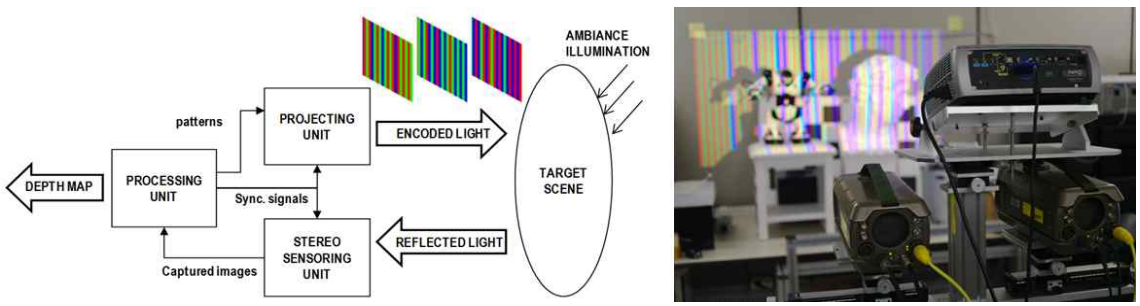

그림 1. 구조광을 이용한 깊이지도 획득 시스템 구성도
Fig. 1. System of depth map acquisition using structured lights

문에 움직이는 물체가 아닌 고정된 장면에 대해서만 깊이지도를 얻을 수 있다. 이진패턴이 아닌 다른 분류인 gray scale이나 컬러 패턴을 사용하는 경우에는 한 장면에 대해 필요한 패턴 영상의 수가 줄어들게 되어 움직이는 물체에 대해서도 깊이지도를 얻을 수 있게 된다. 따라서 본 실험에서는 컬러 패턴을 사용하였다.

전체적인 연산과정은 그림 2와 같다. 본 실험에서 카메라로 촬영한 영상은 1280×800의 크기를 갖는 stereo 영상이다. 영상의 크기가 커지면 그에 비례하여 연산량도 늘어나기 때문에 실시간으로 가능하게 하기 위해서는 병렬화 기법을 적용하여 고속화하는 것이 필수적이다. 전체 연산 과정 중 본 논문에서 중점적으로 고속화를 다루는 부분은 영상 획득, 영역 분할, 패턴 디코딩, 깊이 추정 부분이며 다음 장을 통하여 두 가지 멀티 쓰레딩 기법의 차이를 설명하며 각 연산별로 수행하는 동작과 어떠한 멀티 쓰레딩이 최적화된 방법을 소개 하고자 한다.

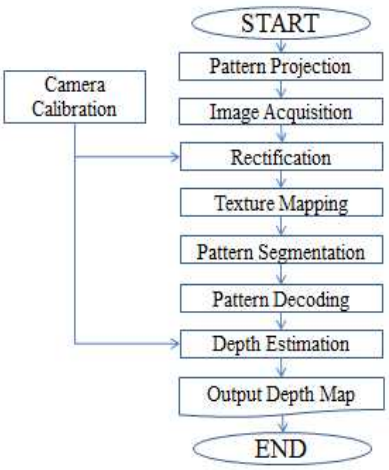

그림 2. 깊이지도 획득을 위한 수행 과정
Fig. 2. The process of depth map estimation

III. 가속화 방향

일반적으로 많이 다루고, 잘 알려진 병렬화 기법에는 CUDA와 OpenMP가 존재한다. CUDA는 GPU(Graphics Processsing Unit)를 이용하며 OpenMP는 multi-coreCPU (Central Processing Unit)를 이용한다. 이 두 가지 방법은 비슷한 방법으로 병렬화를 수행하지만 큰 차이점이 존재한다. 먼저 쓰레드의 수이다. 기본적으로 병렬화 방법을 다룰 때 사용되어지는 쓰레드의 수에 비례해서 속도개선이 이루어진다. 때문에 많은 고속화를 다루는 기존의 연구들에서는 쓰레드의 수가 많은 GPU를 이용한 병렬화를 다뤄왔다. 최신식의 CPU를 사용한 본 실험에서 CPU의 쓰레드는 12개이다. 하지만 요즘 사용되어지는 대부분의 그래픽카드의 사용가능한 쓰레드는 보통 천개가 넘으며, 본 실험에서 사용한 GPU의 쓰레드는 1024개이다. 잘 설계되어진 병렬 프로그래밍일수록 두 기법에서의 고속화 비율이 달라진다. 다음으로 큰 차이점은 바로 동작하는 방식이다. OpenMP에서의 쓰레드는 독립적이다. 하나의 명령을 모든 쓰레드가 동시에 수행을 하거나, 모든 쓰레드들이 각기 다른 명령을 수행한다. 하지만 GPU에서의 쓰레드는 종속적이다. 데이터를 Grid와 Block으로 나누어 모든 쓰레드가 같은 명령을 수행하게 된다. 이 때문에 GPU를 사용하는 CUDA를 다룰 때는 분기문이나 조건문, 재귀 알고리즘 등에 취약한 모습을 보이며, 원하는 성능에 한참 미치지 못하는 결과를 나타내기도 한다. 하지만 OpenMP를 다룰 때는 일반적인 CPU 연산과 동일하기 때문에 CPU의 코어(core)의 수에 비례하여 고속화가 이루어진다. 그래서 본 논문에서는 병렬화를 통한 고속화를 수행 할 때 각각의 부분에 맞는 멀티 쓰레드를 이용하는 것을 제안한다.

표 1. CUDA와 OpenMP의 비교
Table 1. compare CUDA with OpenMP

구분	CUDA	효율성	OpenMP
고속화 한계	1000배 이상	>	Core의 수에 비례
전송 시간	CPU->GPU & GPU->CPU	<	없음
처리 방식	Grid 와 Block으로 나누어 처리	<	일반적인 CPU 연산과 동일
동작 방식	SIMT(Single-Instruction, Multi-Thread) 구조	<	

1. 영상 획득 (Image acquisition)

실험에 사용되는 입력 영상은 de Bruijn color 시퀀스를

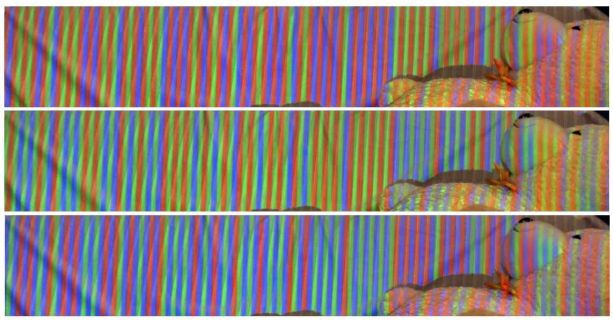

그림 3. De Bruijn 패턴이 조사된 영상
Fig. 3. The images for de Bruijn patterns

조사하여 촬영해놓은 영상을 이용한다. 하나의 깊이지도를 얻기 위해서는 좌,우 시점 각각 3장씩의 영상이 필요하다. 그림 3은 한 장면에 3가지 다른 de Bruijn 패턴이 조사된 영상이다. 영상을 읽어 들이는 시간도 전체 수행과정의 시간측정에 포함하였기 때문에 이 과정은 좌우를 각각 OpenMP를 활용하여 처리 하였다. FIFO(First Input, First Out)구조로 처음 시스템이 구동될 때는 좌, 우 각각 3장씩의 영상을 얻어오며, 이후에는 영상이 가지고 있던 메모리 공간상에 처음의 영상을 빼고 현재의 영상을 추가로 얻어와 항상 3장씩의 영상을 다루게 된다.

실험에 앞서 카메라 교정(camera calibration)[10]을 수행하였으며, 이를 통해서 얻어진 대응 데이터(map data)를 통하여 정류(rectification)[11]를 수행하였다.

그림 4와 같이 얻어온 좌, 우 각각 3장의 영상(패턴이 다른 영상)은 한장의 영상으로 합쳐져 텍스쳐 영상(textured image)을 만들게 된다. 이 텍스쳐 영상은 ROI(Region of Interest) 영역을 구하기 위한 과정이다.

영상 획득 과정의 영상 불러오는 과정을 제외하고 정류와 텍스쳐 영상을 만드는 과정은 화소 대 화소로 이루어진 간단한 연산으로 이루어져있다. 주변값을 참조하지 않으며 조건문을 포함하는 연산이 아니기 때문에 병렬처리를 하는데 있어서 가장 최적화 되어있는 부분이다. 그래서 이 과정은 CUDA를 통해 병렬처리를 했을 때 GPU의 많은 쓰레드와 함께 최고의 효율로 고속화가 가능하다.

2. 영역 분할 (Pattern segmentation)

영상 획득 단계를 통해서 얻은 패턴이 존재하는 영상으로부터 패턴을 구분하는 단계이다. 해당 영역에서 각 화소별 정확한 줄무늬 색상을 검출하기 위해 CIELab 색 공간[12]으로 변환하여 화소별 색 성분을 구한다. 변환된 색 공간은 그림 5(a)와 같다. 검출된 색에 따라 0-빨간색, 1-녹색, 2-파란색, 4-알 수 없거나 검은 영역으로 인덱스를 부여한다.

한 번의 영역 분할 과정이 끝나면 정확도 향상을 위하여 정제(refinement) 과정을 수행한다. 정제과정은 가로 방향으로 사용된 패턴을 이용함에 따라 수직 방향으로 연속성이 존재할 것으로 가정하고, 연속성을 비교하는 구문으로 구성된다. 정제과정을 수행 했을 때의 결과는 그림 5(b)와 같다. 그림 5(c)는 정제 과정을 수행하기 전과 후의 일부분을 확대한 모습으로 잡음이나 구멍이 많이 사라진 것을 볼 수 있다.

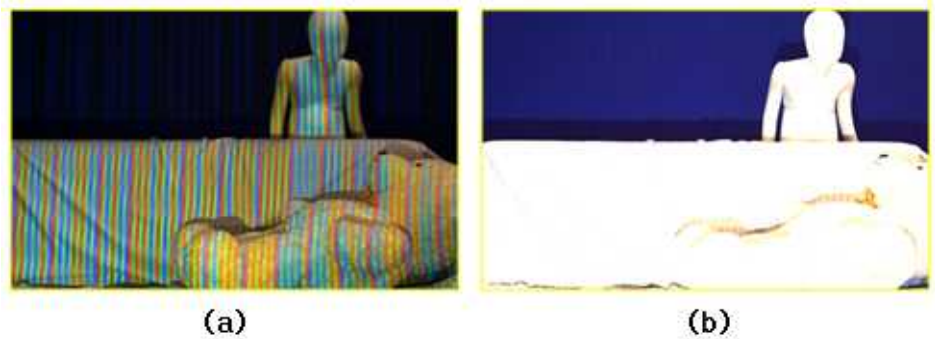

그림 4. (a) 패턴이 조사된 입력 영상, (b) 텍스쳐 영상
Fig. 4. (a) Input image with projected pattern, (b) Textured image

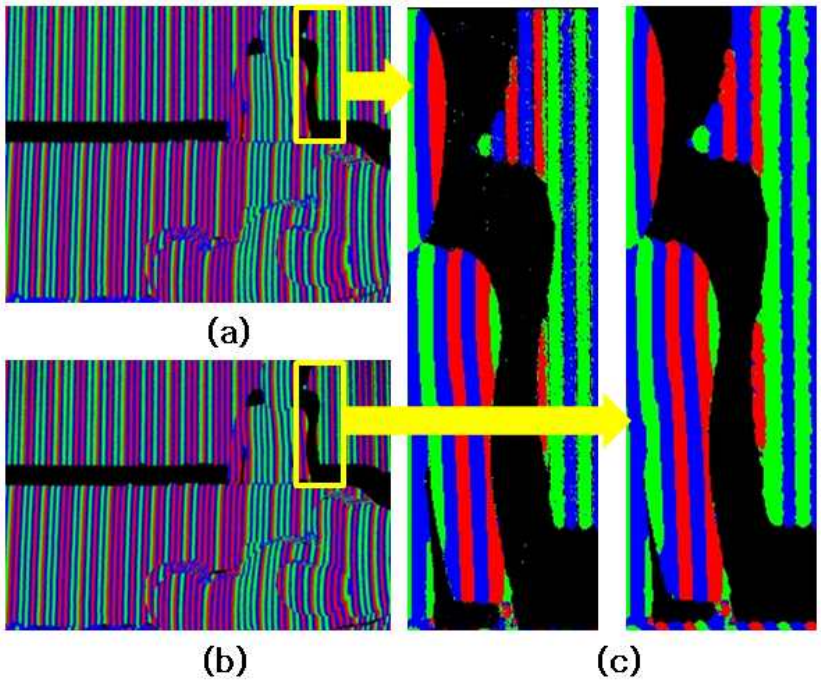

그림 5. (a) 분할 과정을 거친 영상, (b) 정제 과정을 거친 영상, (c) 각각의 확대 영상
Fig. 5. (a) Segmentation image, (b) Refinement image, (c) Enlarged part of (a) and (b)

색 공간의 변환은 각 화소별로 독립적으로 이뤄지는 연산을 통해 수행이 되고, 인덱스 부여 역시 각 화소별로 독립적으로 값의 비교를 통해 수행이 된다. 병렬화가 잘 진행될 수 있는 조건이기 때문에 OpenMP를 사용 할 경우 우리의 실험에서는 약 12배 의 성능 향상까지만 기대 할 수 있지만, CUDA를 사용 할 경우 훨씬 큰 향상을 기대 할 수 있다. 따라서 이전 영상 획득 단계와 마찬가지로 CUDA를 사용했을 때 더 높은 효율의 고속화를 수행한다. 정제과정은 위, 아래 화소의 비교를 통하여 이루어지지만 그 복잡도가 높지 않고 각 픽셀별로 독립적으로 수행이 가능하기 때문에 CUDA를 사용하여 수행하였을 때 손실이 크지 않다.

3. 패턴 디코딩 (Pattern decoding)

패턴 디코딩은 [8]의 방법을 사용한다. 이 과정은 구분된 색상 값을 통해 원래의 패턴과 촬영되어진 패턴의 비교를 통하여 이루어진다. 실험에서는 LUT(Look Up Table)을 이용하였다. LUT을 이용한 방법은 초기에 프로젝터에서 투사되는 패턴의 각 자리에 대해 곱셈 연산과 덧셈 연산을 통해 해당되는 테이블의 값에 위치 정보를 저장하도록 LUT을 구성한 후, 촬영된 영상 내의 얻어진 색상 정보에 대해 동일한 연산을 통해 얻어진 값을 통해 LUT의 값을 가져오는 방식으로 위치를 찾아낸다. 이 LUT을 이용할 경우, 연산이 빠르다는 장점을 가지지만, 중간에 오류가 발생되는 픽셀의 정보를 찾을 수 없다는 단점을 가지게 된다. 때문에 이를 보완하여 이용하게 된다.

먼저 초기 패턴 ID 값을 위해 구조광을 최초 생성시 패턴 정보를 이용하여 LUT 을 구성한다. 사용된 de Bruijn 시퀀스의 특성에 따라 시퀀스 내의 모든 부분 시퀀스에 대해서 고유의 테이블 값을 가지게 된다. 앞서 부여된 색상정보에 따른 인덱스를 통하여 한 라인에 대해서 식(1)의 계산 값을 통해서 LUT에 저장된 값을 현재의 초기 패턴 ID 값으로 사용한다.

$$id = \overrightarrow{s_0}n^0 + \overrightarrow{s_1}n^1 + \overrightarrow{s_2}n^2 + \cdots + \overrightarrow{s_{k_{dec}-1}}n^{k_{dec}-1} \quad (1)$$

초기 패턴 ID 값에 대해 LUT을 사용할 때 중간에 에러가 발생하는 경우 값을 얻어올 수 없게 된다. 이러한 픽셀들에 대해서는 보완을 해 주어야 한다. 우선 초기 패턴 ID 값에 대한 신뢰도를 계산한다. 1차원 수직방향 시퀀스를 패턴영상으로 사용하기 때문에 수직방향의 패턴 값의 경우, 유사하거나 동일한 값을 가질 확률이 크다. 이를 바탕으로 초기 패턴 ID 값에 대해서 현재 픽셀의 주변 상하 라인에서의 이웃 픽셀의 패턴 값을 이용하여 유사한 픽셀의 수의 비를 신뢰도로 계산한다. 신뢰도에 대한 계산이 끝나면, 영상 전체에 대해 패턴 값이 없는 픽셀에 대해서 일정 거리 이내에 있는 상하 픽셀의 색상 정보를 비교한다. 색상 값이 같은 픽셀이 존재하는 경우, 그 중 가장 신뢰도가 높은 픽셀의 패턴 정보를 현재 픽셀의 패턴 ID 값으로 사용한다.

패턴 디코딩 과정은 값을 찾는 과정에서 많은 조건문과 이전 값을 참조하는 연산을 포함하고 있다. 높은 고속화 효율을 보여주는 CUDA의 경우 조건문이나 이전 값을 참조하는 연산에는 취약한 효율을 보여준다. 하나의 명령이 전체적으로 동시에 수행되는 구조이다. 때문에 조건문을 통해 특정 경우에 특정 연산을 더 수행해야 하는 경우, 전체의 연산이 한 부분의 연산이 끝날 때까지 미뤄지게 된다. 또한, 실험에 사용된 패턴은 가로로 776의 길이를 가지는 패턴으로 이뤄져있다. 한 라인에 대해서 초기 패턴 ID 값을 구하기 때문에 영상의 가로줄에 대한 연산이 나누어 지는 것은 좋지 못하다. CUDA를 사용했을 때 고속화의 효율이 좋지 못한 부분으로 OpenMP를 사용하여 고속화를 수행하였다. 실험에서 사용한 영상이 1280×800 크기를 갖는 영상이기 때문에 이를 세로로 200씩 나누어 4등분하여 각 쓰레드를 이용하여 병렬화를 수행하였다. 그 결과 CUDA에 비해서 더 높은 고속화된 결과를 얻을 수 있었다.

4. 깊이 추정 (Depth estimation)

가시 구조광과 능동 스테레오의 통합 방법은 프로젝터와 카메라, 카메라와 카메라 사이의 위치 관계를 이용하여 깊이 정보를 획득하는 방법이다. 이 방법은 가시 구조광 기법으로부터 신뢰할 수 있는 세 시야간의 대응 관계를 얻고 가시 구조광으로 구하지 못한 영역에 대해서만 추가적으로 스테레오 매칭을 통해 대응 관계를 추정한다[13].

먼저 시차(disparity) 추정을 수행한다. 이 과정은 디코딩 과정과 마찬가지로 가로로 검색하면서 조건부 연산으로 이루어진 연산이다. 때문에 이전과 마찬가지로 세로로 4등분하여 OpenMP로 수행 하였다.

다음으로 깊이지도는 앞에서 구한 시차 값을 통하여 구하게 된다. 실험 이전에 구한 카메라 투영 행렬(camera projection matrix)값을 이용하여 각각의 깊이 값을 구하며 이때 최대 깊이 값을 구한다. 최대 깊이 값은 최종적으로 화면에 나타나게 될 영상의 정규화를 위해 쓰이게 된다. 이때의 연산은 실험적으로 OpenMP를 사용했을 때 더 효과적으로 나타났으며 결과는 그림 7과 같다.

가시 구조광과 능동 스테레오의 통합 방법은 한쪽 영상에만 포함된 영역 및 경계에서 패턴의 길이가 부족하여 디코딩이 되지 못한 영역은 잡음으로 나타나게 된다. 또한 그

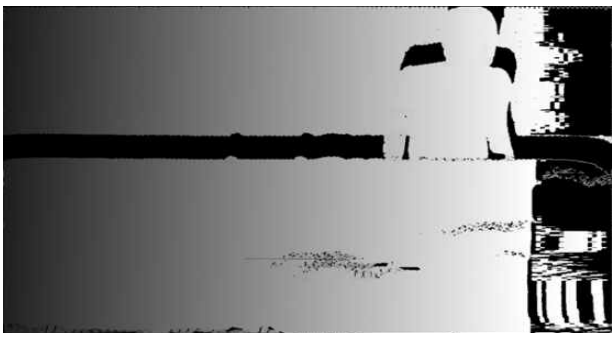

그림 6. 디코딩 영상
Fig. 6. Decoding image

그림 7. 최종적으로 얻은 깊이지도 영상
Fig. 7. Final depth map image

림자 영역의 경우 가시 구조광이 제대로 보이지 못하기 때문에 마찬가지로 잡음으로 나타나게 된다. 그림 6과 7의 영상에서 오른쪽에 나타난 잡음은 두 대의 카메라가 촬영하는 영역이 다르기 때문에 생기는 부분으로 두 대의 카메라의 간격을 줄이면 해당 잡음 범위를 줄일 수 있다.

Ⅳ. 실험 결과

여기서는 구현 결과에 대해 설명 한다. 앞에서 언급한대로 각각의 연산에 더 효율적인 병렬처리를 찾아 OpenMP와 CUDA를 사용하여 구현 하였으며, 1280×800 해상도를 가지는 영상을 사용하였다. 실험에 사용된 시스템의 환경은 표 1과 같다.

표 2. 실험 환경
Table 2. Test environment

항목	세부사항
CPU	Intel core i7 980 3.33GHz
GPU	GTX 590
Memory	24GB
OS	WINDOWS 7 64bit

표 2는 각 단계 별로 연산에 소모된 시간을 비교한 결과이다. 표의 CPU는 CPU의 단일 쓰레드(single-thread)를 이용했을 때의 결과이며, 멀티 쓰레드는 제안된 방법에 의한 최고의 고속화를 나타낸 멀티 쓰레딩을 적용했을 때의 결과이다.

표를 살펴보면 크게 두 가지로 분류가 된다. OpenMP를 사용한 영상 불러오기(image loading), 패턴 디코딩, 깊이 추정 단계와, CUDA를 사용하여 고속화를 수행하였던 영상 획득, 영역 분할 단계이다. 영상 획득 단계와 영역 분할 단계는 앞에서 말 한대로 병렬화를 하기에 아주 좋은 조건이다. 때문에 단일 쓰레드를 사용했을 때 5,585ms 에서 7.38ms 로 약 750배정도의 엄청난 고속화 수행이 가능하였다. 패턴 디코딩 과 깊이 추정 단계는 단일 쓰레드를 사용했을 때 477ms 에서 23.94ms 로 약 20배정도의 고속화를 수행할 수 있었다. 이 두과정은 CUDA를 사용하여 고속화를 수행했을 때 얻을 수 있었던 140ms 보다 약 7배 정도 향상된 고속화를 수행하였다. 결과적으로 전체 연산을 단일 쓰레드에서 수행했을 때 6,079ms 걸렸던 것을 멀티 쓰레드를 사용함으로써 34.61ms 로 약 175배의 고속화 성능을 보였다.

표 3. 각 단계별 실험 결과(ms)
Table 3. Result of step by step running time(ms)

단계	CPU	Multi-thread
Image loading	17	3.29
Texture image + Pattern segmentation	5,585	7.38
Pattern decoding	411	14.59
Disparity + Depth estimation	66	9.35
Total time	6,079	34.61

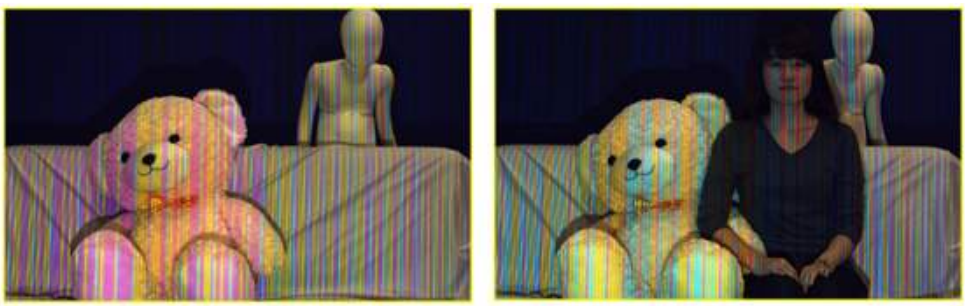

그림 8. 객체 수 변화에 대한 실험 영상
Fig. 8. The input images: (a) 2 objects, (b) 3objects

표 3은 그림 8의 서로 다른 실험 영상에서의 결과를 나타내었다. 두 영상은 같은 패턴크기와 같은 거리에서 실험 되었지만 화면에 위치하는 객체의 수(객체가 전체 화면에서 차지하는 비율)가 다른 영상이다. 실험 결과는 객체가 증가하면서 전체 실험에 소요된 시간을 미미하게 증가 시켰다. 영역 분할 과정에서 배경 영역은 ROI 영역에서 제외된다. 그림 8의 (a)와 (b)를 비교하면 사람이 추가되면서 배경의 영역이 줄어든다. 따라서 소모되는 시간이 증가하게 된다. 깊이 감을 가진 객체가 추가되면서 디코딩 단계에서 비용함수를 구하는 과정 역시 증가하게 되어 소요된 시간의 증가를 불러 일으켰다. 하지만 시간 증가는 매우 작은 폭으로 증가 하며 모든 영상에서 25 fps 이상의 속도를 얻었다.

표 4. 그림 8의 입력 영상에 따른 실험 결과(ms)
Table 4. Experiment result for other input image(ms)

단계	그림 8 (a)	그림 8 (b)
Image loading	3.66	3.86
Texture image + Pattern segmentation	5.89	7.96
Pattern decoding	11.20	12.05
Disparity + Depth estimation	10.58	9.77
Total time	31.33	33.64

V. 결 론

본 논문에서는 3D 방송을 위해서 구조광을 이용하여 깊이지도를 획득하는 최적화된 고속화 방법을 제안하였다. 따라서 제안된 기법은 깊이지도 정보를 실시간으로 획득하기 위하여 구조광의 패턴정보를 다루는 과정에서 발생하는 반복적인 연산에 대한 각각의 수행 동작에 맞도록 멀티 쓰레드를 사용하여 고속화 하는 것이다. 대부분의 GPU를 통한 병렬 처리 기법은 훌륭한 고속화 방법이다. 하지만 병렬 수행 단계에 따라서 GPU에 비해 현저히 낮은 쓰레드를 가진 CPU의 multi-core를 활용하는 것이 더 좋은 결과를 수행할 수 있다. 결과에서 볼 수 있듯이 병렬화를 수행하지 않았을 때에 비해서 약 175배의 속도 향상을 보였으며, 초당 최소 25 프레임 이상의 속도로 1280×800 해상도에서의 깊이지도를 획득 하였다. 따라서 실시간 3D 방송을 위해 충분한 처리 속도를 얻을 수 있다.

앞으로 우리는 시간 소모가 가장 큰 패턴 디코딩에 대해서 좀 더 고속화 할 수 있는 방법을 알고리즘 적인 측면으로 접근하여 찾을 것이며, 잡음이 생겼던 획득한 깊이지도의 정확도 향상에 대한 연구를 진행 할 것이다.

참 고 문 헌

[1] K. Sato and S. Inokuchi, “Three-dimensional surface measurement by space encoding range imaging,” Journal of Robotic System, 2:27–39, 1985

[2] E. Horn and N. Kiryati, “Toward optimal structured light patterns,” Image and Vision Computing, 17, 1999

[3] Daniel Scharstein and Richard Szeliski, “High-accuracy stereo depth map using structured light,” IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Madison, WI, June, 2003

[4] S. Y. Chen, Y. F. Li, and J. W. Zhang "Vision processing for realtime 3-D data acquisition based on coded structured light", IEEE Trans. Image Process, vol.17, no. 2, pp. 167 - 176 , 2008

[5] Qiang Li, Moyuresh Biswas, Mark R. Pickering, and Michael R. Frater, “Dense depth estimation using adaptive structured light and cooperative algorithm,” Computer Vision and Pattern Recognition Workshops, June, 2011

[6] Song Zhang and Peisen S. Huang, “High-resolution, real-time three-dimensional shape measurement,” Optical Engineering, Vol.45, No.12, 2006

[7] S. Keerativittayanun, T. Kondo, P. Sira-uksorn, T. Phatrapornant, and M. Sato, “3D data acquisition using active stereo based on spatial neighbourhood technique,” Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, May, 17–19, 2011

[8] L. Zhang, B. Curless, and S. M. Seitz, “Rapid shape acquisition using color structured light and multi-pass dynamic programming,” 1st International Symposium on 3D Data processing, Visualization, and Transmission, Padova, Italy, June, 10-21, 2002

[9] Jordi Pages, joaquim Salvi, Christophe Collewet, and Joseq Forest, “Optimized De Bruijn pattern for one-shot shape acquisition,” Image and Vision Computing Vol.23, No.8, pp. 707-720, August, 2005

[10] N. H. Gabriel Falcao and J. Massich, “Plane-based calibration of a projector-camera system,” technical report, VIBOT, 2008.

[11] C. Loop and Z. Zhang, "Computing Rectifying Homographies for Stereo Vision," IEEE Conference Computer Vision and Pattern Recognition, Fort Collins, CO, USA, pp. 125-131, June, 1999

[12] Adrian Ford and Alan Roberts, "Color Space Conversions," technical report, Westminster University, 1998

[13] H. Saito and T. Kanade, "Shape reconstruction in projective grid space from large number of images," IEEE Conference Computer Vision and Pattern recognition, vol. 2, pp. 49-54, 1999

저 자 소 개

조 칠 석

- 2010년 : 대진대학교 통신공학과 학사
- 2010년 ~ 현재 : 한양대학교 전자컴퓨터통신공학과 석사과정
- 주관심분야 : 3차원 영상처리, 컴퓨터 비젼, 증강현실

전 지 인

- 2011년 : 한양대학교 전자통신컴퓨터공학부 학사
- 2011년 ~ 현재 : 한양대학교 전자컴퓨터통신공학과 석사과정
- 주관심분야 : 파노라마, 영상인식, 영상정렬, 스테레오

추 현 곤

- 1998년 : 한양대학교 전자공학과 학사
- 2000년 : 한양대학교 전자공학과 석사
- 2005년 : 한양대학교 전자통신전파공학과 박사
- 2005년 ~ 현재 : 한국전자통신연구원 선임연구원
- 주관심분야 : 3DTV, 디지털방송기술, 컴퓨터비전

박 종 일

- 1987년 : 서울대학교 전자공학과 학사
- 1989년 : 서울대학교 전자공학과 석사
- 1995년 : 서울대학교 전자공학과 박사
- 1992년 ~ 1994년 : 일본 NHK 방송기술연구소 객원연구원
- 1995년 ~ 1996년 : 한국방송개발원 선임연구원
- 1996년 ~ 1999년 : 일본 ATR 지능영상통신연구소 연구원
- 1999년 ~ 현재 : 한양대학교 공과대학 컴퓨터공학부 교수
- 주관심분야 : 가상현실, 컴퓨터그래픽스/비전, 3차원 영상처리, 인간컴퓨터 상호작용