다시점 촬영 시스템을 위한 카메라 동기화 구현

*박정탁, *박병서, *서영호
*광운대학교
*littlepine97@kw.ac.kr, *bspark@kw.ac.kr, *yhseo@kw.ac.kr

Implementation of camera synchronization for multi-view capturing system

*Jung Tak Park, *Byung Seo Park, *Young-Ho Seo
*Kwangwoon University

요약

본 논문에서는 RGB이미지와 Depth 이미지를 촬영할 수 있는 촬영 장비인 Azure Kinect를 사용해 다시점 촬영 시스템 구성을 위한 카메라 동기화 시스템을 제안한다. 제안한 시스템에는 8대의 Azure Kinect 카메라를 사용하고 있으며 각 카메라는 3.5-mm 오디오 케이블로 연결되어 외부동기화 신호를 전달한다. 그리고 이미지를 저장할 때 발생하는 메모리에서의 병목현상을 최소화하기 위해 촬영 시스템의 동작을 16개의 버퍼로 나누어 병렬 컴퓨팅으로 진행한다. 이후 동기화 여부에 따른 차리를 디바이스 타임스탬프를 기준으로 하여 비교한다.

1. 서론

촬영 시스템에서 더욱 정확한 캡처를 진행하기 위해서는 카메라 간의 동기화가 필수적이다. 인간의 눈과 달리 카메라는 입력되는 미세한 차이에도 변화를 나타내기 때문에 정확한 타이밍에 모든 카메라가 동작하는 것이 중요하다.

본 연구에서는 Azure Kinect 카메라를 3.5-mm 오디오 케이블로 연결해 외부 동기화 신호를 전달하는 방식으로 동기화를 구현했다. 그리고 하나의 PC에 여러 대의 카메라를 연결하게 되면 캡처 사이에 이미지를 저장하는 과정에서 메모리 병목현상에 의한 지연이 발생할 수 있으므로 정확한 캡처가 어려워질 수 있다. 이를 방지하기 위해 8대의 Azure Kinect를 위한 16개의 버퍼를 생성하여 각 버퍼는 한 버퍼가 데이터를 입력받는 동안 다른 버퍼는 데이터를 저장하는 동작을 수행하는 시스템을 제안한다.

2. 카메라 시스템의 구성

본 논문에서는 앞서 설명한 바와 같이 8대의 RGB-D 카메라를 이용하여 3D 포인트 클라우드 모델의 캡처가 가능한 촬영 시스템을 구성하였다. 구성한 시스템의 구조는 그림 1과 같다. 그림 1(a)는 동기화 케이블을 연결하기 전의 구조이고, 그림 1(b)는 동기화 케이블을 연결한 이후의 구조이다.

제안한 시스템을 적용하기 위해, 먼저 다수의 Azure Kinect를 동기화하는 작업이 수행되어야 한다. 각 Azure Kinect는 Daisy-chain Configuration 방식으로 3.5-mm 오디오 케이블을 이용해 연결된다. Master device는 Sync out 신호를 내보내며, 다른 Subordinate device는 이 신호를 Sync in 단자로 받아 Sync out 단자로 오디오 케이블을 통해 다음 Subordinate device로 전달한다. 다음으로 각 카메라의 촬영 딜레이와 노출시간을 조정해야 한다. IR 신호가 간섭을 일으켜 정확한 Depth capture에 방해가 될 수 있으므로 각 디바이스간 캡처는 160us의 간격을 두고 이루어지며, 이때 노출시간은 12725us로 설정한다[1].

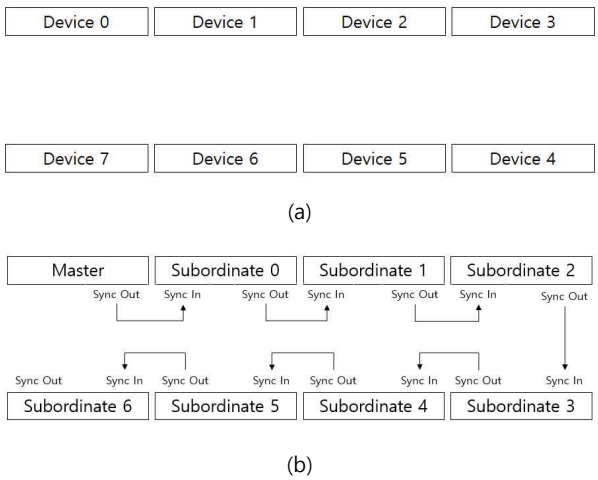

그림 1. 8대의 RGB-D 카메라를 이용한 3D 포인트 클라우드 촬영 시스템 (a) 동기화 케이블 연결 이전, (b) 동기화 케이블 연결 이후

다음으로 본 실험에서 사용한 시스템 구성을 설명한다. 과정을 설명하기에 앞서, 외부 동기화 설정에서 Azure Kinect를 사용할 때, 모든 Subordinate device는 Master device의 캡처를 신호로 받아 캡처를 시작하므로 모든 Subordinate device는 캡처 명령 입력 시 Master device의 캡처까지 대기상태를 유지한다. 그림 2(a)에서 이를 고려하여 병렬 컴퓨팅을 적용해 모든 Subordinate device는 동시에 캡처 대기상태로 진입하며 이후 Master device가 캡처와 동시에 캡처를 진행한다. 이와 동시에 이미 데이터가 저장된 버퍼의 데이터를 이미지 파일로 출력한다. 그림 2(b)에서는 시스템의 구조를 나타내고 있다. 우선 각 버퍼를 A버퍼, B버퍼라고 할 때, A버퍼에서 캡처를 진행하고, 다음 단계에서는 A버퍼에 저장된 데이터를 이미지 파일로 저장하면서 B 버퍼에서 캡처를 진행한다. 그리고 다시A 버퍼에서 캡처를 진행하면서 B 버퍼에 저장된 데이터를 이미지파일로 저장하는 과정을 반복한다.

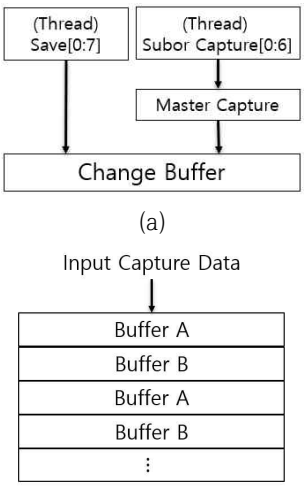

그림 2. 실험에서 사용한 시스템의 (a) 동작, (b) 버퍼 구조

3. 실험결과

본 장에서는 각각 동기화를 적용하지 않았을 때와 동기화를 적용한 후의 컬러 이미지 디바이스 타임 스탬프를 36프레임까지 그래프로 나타낸 결과이다. x축은 프레임을 나타내고, y축은 해당 프레임을 캡처하는 순간에 기록된 디바이스 타임 스탬프값을 나타낸다. 디바이스 타임 스탬프는 하드웨어가 캡처한 이미지의 노출시간의 중간시간을 기록한 값이다.

그림 3(a) 비동기 동작에서 타임 스탬프 그래프는 각 디바이스가 독립적으로 동작하기 때문에 디바이스의 타임 스탬프가 일치하지 않아 최대 40000us의 시차가 발생했고, 누적된 시차에 의한 딜레이로 FPS에 변화가 생겨 기울기가 변경된 것을 그래프상에서 확인할 수 있다.

반면 그림 3(b) 동기화 동작 그래프에서는 그림 3(a)보다 감소한 간격을 가져 최대 23000us의 시차 나타내며 일정한 FPS를 갖는 것을 확인할 수 있다.

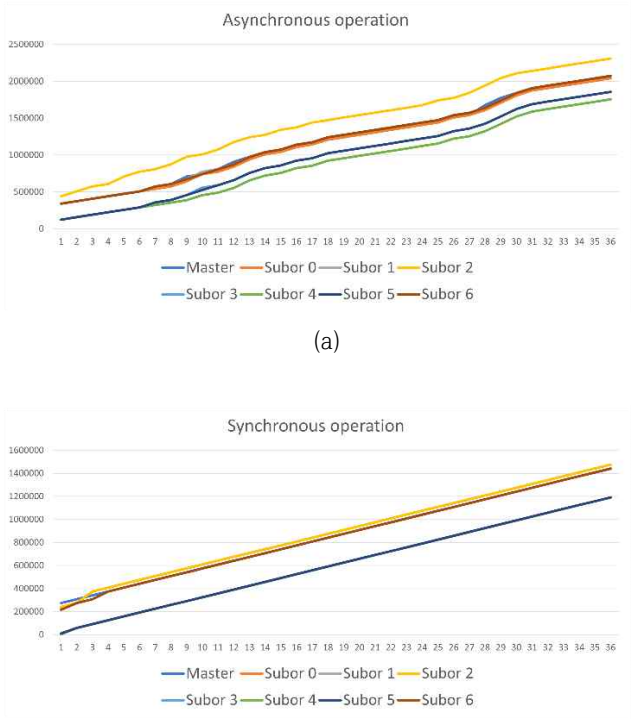

그림 3. 디바이스 타임스탬프 그래프 (a) 비동기 동작, (b) 동기화 동작

4. 결론

본 연구에서는 8대의 Azure Kinect 카메라의 동작에서 동기화 적용에 따른 디바이스 타임 스탬프 차이로 동기화 결과를 보였다. 비동기 동작에서 각 디바이스간 타임 스탬프 값 차이가 최대 35000us까지 발생한 것에 반해, 동기화 동작에서는 최대 20000us차이를 나타내며 동기화된 각 디바이스 사이의 타임 스탬프 차이는 3000us보다 낮은 값을 보였다.

감사의 글

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT 연구센터지원사업의 연구결과로 수행되었음 (IITP-2021-0-01846)

참고문헌

[1] Microsoft. About Azure Kinect DK. Available online: https://docs.microsoft.com/en-us/azure/kinect-dk/about-azure-kinectdk