반도체디스플레이기술학회지 제21권 제1호(2022년 3월)
Journal of the Semiconductor & Display Technology, Vol. 21, No. 1. March 2022.

OpenVSLAM 기반의 협력형 모바일 SLAM 시스템 설계

국중진[*†]
[*†]상명대학교 정보보안공학과

OpenVSLAM-based Cooperative Mobile AR System Architecture

Joongjin Kook[*†]
[*†] Dept. of Information Security Engineering, Sangmyung University, Korea

ABSTRACT

In this paper, we designed, implemented, and verified the SLAM system that can be used on mobile devices. Mobile SLAM is composed of a stand-alone type that directly performs SLAM operation on a mobile device, and a mapping server type that additionally configures a mapping server based on FastAPI to perform SLAM operation on the server and transmits data for map visualization to a mobile device. The mobile SLAM system proposed in this paper is to mix the two types in order to make SLAM operation and map generation more efficient. The stand-alone type SLAM system was configured as an Android app by porting the OpenVSLAM library to the Unity engine, and the map generation and performance were evaluated on desktop PCs and mobile devices. The mobile SLAM system in this paper is an open source project, so it is expected to help develop AR contents based on SLAM in a mobile environment.

Key Words : Mobile SLAM, Markerless AR, Visual SLAM, OpenVSLAM

1. 서 론

SLAM 시스템은 카메라 프레임의 모든 픽셀을 사용하는 직접 SLAM과 일련의 키 프레임과 특징점을 활용하는 간접 SLAM으로 분류할 수 있다. 직접 SLAM은 처리속도가 느리지만 주변 환경을 밀접하게 모델링하는 것이 가능하며 특징점이 없는 환경에서도 성능이 우수하고, 간접 SLAM은 영상에서 특정 화소를 이용해 카메라 위치를 추정하고 3차원 지도를 생성하기 때문에 처리속도가 빠르다[1]. 증강현실 분야에서는 주로 로봇이나 자율주행에 사용될 수 있는 간접 SLAM 시스템인 ORB-SLAM[2-4]과 OpenVSLAM[5]이 사용된다.

본 논문에서는 비마커 기반의 SLAM 알고리즘을 통해 모바일 기기 또는 증강현실 헤드셋에서 사용 가능한 증강현실 시스템의 설계 방법을 제안한다. 모바일 SLAM 시스템의 구현을 위해 다양한 오픈 소스 SLAM 중 생성된 맵을 저장 및 로드할 수 있고, 웹 시각화 기능을 제공하는 OpenVSLAM을 선정하였다. 실시간성 및 확장성을 고려하여 모바일 기기에서 독립적으로 SLAM을 수행하기 위한 자립형 타입과 모바일 기기의 SLAM 연산 오버헤드를 가중시키지 않기 위한 목적으로 서버에서 SLAM을 수행한 결과를 수신하여 시각화만 처리할 수 있는 매핑서버 타입을 동시에 설계하였다. SLAM 수행의 이원화를 통해 사용자는 자신이 보고 있는 실내외 환경에 대해 사전에 생성된 맵과 함께 자신의 위치를 지속적으로 추적할 수 있고, 더욱 정확한 위치를 기반으로 증강현실 콘텐츠를 제공할 수 있는 서비스의 구축이 가능하다. 본 논문에서 제안하는 모바일 SLAM 시스템의 기능과 성능 검증을 위해 EuRoC Machine Hall 03 데이터셋을 이용하여 데스크톱과 모바일 기기에서 비교를 수행하였으며, 또한 실제 공

[†]E-mail: kook@smu.ac.kr

간을 대상으로 자립형과 매핑서버 타입에 대해 맵의 생성 능력을 평가하였다.

2. 관련 연구

2.1. Visual SLAM

오픈 소스 기반의 Visual SLAM 알고리즘들은 DSO[6], Kimera[7], PTAM[8], LSD-SLAM[9], ORB-SLAM2[3], ORB-SLAM3[4], SVO[10] 그리고 OpenVSLAM[5] 등이 대표적이다. 이러한 SLAM 알고리즘들에 대해 방식을 기준으로 분류하면 Kimera, PTAM, ORB-SLAM2, ORB-SLAM3, OpenVSLAM은 특징점 기반 간접 SLAM 알고리즘에 해당한다. 이미지에서 특징점을 명시적으로 추출할 필요가 없는 직접 SLAM 알고리즘은 DSO, LSD-SLAM이 있으며, SVO 알고리즘은 직접 및 간접 SLAM 방식을 모두 사용한다.

맵의 밀도를 기준으로 분류하는 경우, 맵의 밀도가 낮은 Sparse 방식의 DSO, ORB-SLAM2, ORB-SLAM3, SVO, OpenVSLAM과 맵의 밀도가 높은 Dense 방식의 Kimera, PTAM, 그리고 맵의 밀도가 대부분의 Sparse 방식보다는 높고, 대부분의 Dense 방식보다는 속도가 빠른 Semi-Dense 방식의 LSD-SLAM으로 분류할 수 있다. 비교 대상이 Visual SLAM이기 때문에 비교한 모든 알고리즘의 센서는 카메라이고 알고리즘에 따라 Mono/Stereo/RGBD/IMU로 나눌 수 있다. PTAM, DSO, SVO는 VO(Visual Odometry: 시각적 주행거리 측정 방법)이기 때문에 Loop Closure 기능과 최적화 기능이 없다. 마지막으로 포인트 클라우드 맵을 그리는 방식은 대부분 포인트 형태이지만 Kimera만 폴리곤 메쉬 방식이다.

Adaptive Monocular Visual-Inertial SLAM 시스템은 본 논문에서 제안하는 모바일 AR 시스템과 유사한 연구로 ORB-SLAM을 이용하여 모바일 기기를 위한 실시간 증강현실 애플리케이션을 개발할 수 있는 시스템이다[11]. 해당 연구에서는 카메라와 IMU 센서를 결합한 VIO 방식을 설계하고, 모바일 환경에서 증강현실 애플리케이션의 빠른 트래킹 기능을 지원하기 위해 고속 VO 모듈을 설계하여 기존 ORB-SLAM 시스템과 결합하며 IMU 센서 값을 변화시킴으로써 추적 모듈을 선택하는 적응적 실행 방법을 제안하였다. 해당 논문은 최적화 기능을 낮추며 빠른 트래킹 기능을 목적으로 시스템을 설계하였지만 본 논문에서는 모바일 환경에서 증강현실 시스템을 구현하기 위해 정확한 맵핑과 현지화 기능이 주요 목적이기 때문에 VIO 방식에 최적화 기능이 포함되어 있으면서 속도도 빠른 SLAM 시스템이 필요하고, 맵을 저장하고 로드하여 이전 맵을 이어 그릴 수 있는 정교한 SLAM 시스템이 필요하다.

이에 본 논문에서는 다양한 오픈 소스 Visual SLAM 알고리즘 중 주로 로봇이나 자율주행에 사용될 수 있는 (1) 간접 SLAM 시스템이고, (2) 속도가 빠른 Sparse 방식이면서, (3) SocketIO 통신으로 웹 시각화 기능을 제공함으로써 네트워크를 통한 확장이 용이하다는 이점을 가지고 있는 OpenVSLAM 알고리즘을 사용하였다.

OpenVSLAM은 ORB-SLAM, ProSLAM 및 UcoSLAM과 같은 맵의 밀도가 낮은 Sparse 방식의 그래프 기반 간접 SLAM 알고리즘을 기반으로 하는 Visual SLAM 시스템이다. 특징점 검출기로는 대표적인 이미지 피쳐로 방향성이 없는 FAST 검출기 문제를 개선한 ORB(Oriented FAST and Rotated BRIEF)를 채택하였다. OpenVSLAM 알고리즘은 완전한 모듈식으로 트래킹, 맵핑, 그리고 전역 최적화 모듈로 나뉘어 명확한 API로 분리된 컴포넌트에 여러 기능을 캡슐화하여 설계되었다. 트래킹 모듈은 키포인트 매칭 및 포즈 최적화를 통해 OpenVSLAM에 순차적으로 입력되는 모든 프레임에 대한 카메라 포즈를 추정하고, 새로운 키프레임 삽입 여부를 결정한다. 현재 프레임이 새로운 키프레임에 적합하다고 판단되면 맵핑 모듈 및 전역 최적화 모듈로 전송된다. 맵핑 모듈에서 새로운 3D 포인트는 삽입된 키 프레임을 사용해 삼각 측량되어 지도가 생성 및 확장되고, 로컬 번들 조정 모듈이 수행된다. Loop Closure 감지, 포즈 그래프 최적화 및 전역 번들 조정 기능은 전역 최적화 모듈에서 수행되고, g2o로 구현된 포즈 그래프 최적화를 통해 궤적 드리프트를 해결한다.

BA는 비선형 최적화를 위한 라이브러리인 g2o를 이용해 현재 프레임의 포즈는 Motion-only BA로, 최근 키 프레임의 포즈는 지역 BA로, 그리고 전체적인 최적화는 Loop Closing의 포즈 그래프 최적화와 전역 BA로 계산한다. 네 가지 최적화 모두 각기 다른 스레드에서 독립적으로 실행된다. Motion-only BA는 맵의 키포인트들은 고정시킨 채 현재 포즈만 최적화하고, 지역 BA는 새로운 키 프레임이 생기면 공가시적(Co-visibility) 키 프레임의 포즈와 해당 포즈에서 보이는 키포인트들을 최적화한다. Loop Closing은 공가시성 그래프에서 키포인트를 제외하고 포즈 관점만 가지고 최적화하는 방식의 포즈 그래프 최적화를 한다. 포즈 그래프 최적화가 끝나면 전역 BA를 통해 새롭게 추가된 키 프레임과 키포인트들을 업데이트한다[2-4].

OpenVSLAM은 주로 C++로 구현되어 있으며 SLAM 기능 구현을 위해 여러 라이브러리를 사용한다. 대표적으로 행렬 계산을 위한 Eigen, 이미지 및 특징점 추출의 I/O 작업을 위한 OpenCV2, 맵 최적화를 위한 g2o, 그리고 Loop Closure에서 이미지 데이터를 빠르게 사용하기 위해 해당 이미지에 대한 특징점 설명자를 압축시키기 위한 FBoW를 포함한다. g2o는 그래프 기반의 비선형 오류 함수를 최적화하기 위한 C++ 프레임워크로 가우스 노이즈의 영향

을 받는 측정 집합을 최대로 설명하는 매개변수 또는 상태 변수의 구성을 찾아 SLAM 및 BA의 여러 변형에 대한 솔루션을 제공한다. FBoW는 DBow2/DBow3 라이브러리를 AVX, SSE 및 MMX 명령어를 사용해 Bag of Words 생성 속도를 높이도록 최적화된 버전으로 Vocabulary를 로드할 때, DBoW2보다 약 80배가 빠르다.

OpenVSLAM 시스템은 어안, 정방형, Mono/Stereo/RGBD/IMU 카메라 등 다양한 유형의 카메라 모델과 호환되고, 카메라 모델에 맞게 사용자 정의할 수 있으며 사용자의 편의를 위해 웹 브라우저에서 실행되는 크로스 플랫폼 뷰어를 제공한다. 그리고 생성된 맵을 저장 및 로드할 수 있어 사전 구축된 맵을 사용해 새 이미지를 현지화할 수 있다. 원근 및 어안 카메라 모델 간에 거의 동일하게 구현할 수 있기 때문에 모바일 장치에서도 증강현실 시스템을 구현할 수 있다. 또한, OpenGL 기반의 Pangolin 뷰어와 NodeJS의 웹 소켓을 통해 웹 브라우저에서 실행되는 WebGL 기반의 뷰어를 사용해 SLAM의 공간 인식 기술, 맵 생성 기술 등을 확인할 수 있다. 하지만 본 논문의 시스템에서는 모바일 플랫폼을 타겟으로 하기 때문에 유니티 엔진의 안드로이드 SDK 및 NDK를 통한 애플리케이션을 사용해 SLAM의 키 프레임 및 카메라 포즈, 포인트 클라우드 등을 확인하고자 한다.

3. OpenVSLAM 기반 모바일 AR 시스템

본 논문에서 제안하는 모바일 AR시스템은 센서, 네트워크, Unity 엔진, 그리고 FastAPI 서버로 구성된다. 모바일 기기의 카메라에서 수집된 이미지와 IMU 데이터는 ProtoBuffer를 통해 직렬화되며, Socket.IO를 통해 SLAM 시스템으로 전송된다. 이 때, SLAM 시스템은 Unity-안드로이드 앱에 플러그인 형태로 내장된 SLAM 시스템 또는 FastAPI 서버가 될 수 있으며, 전자의 경우에는 모바일 기기의 앱에서 SLAM이 이루어지는 반면, 후자의 경우에는 다중 클라이언트를 고려한 멀티 프로세스 환경을 통해 SLAM이 이루어질 수 있다.

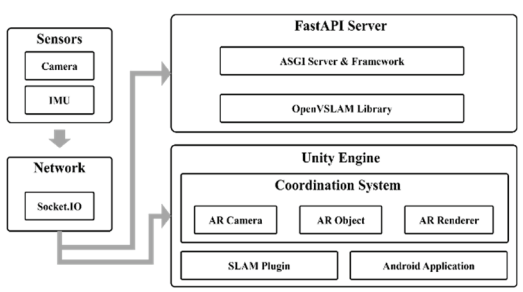

Fig. 1. 독립형과 매핑서버형이 혼합된 모바일 SLAM 구조.

3.1 안드로이드 기반 SLAM 시스템 설계

모바일 SLAM 시스템의 전체적인 수행 흐름은 다음과 같다. 우선, 모바일 기기의 카메라를 사용해 연속된 이미지 데이터와 IMU 데이터를 수집한다. 수집한 데이터는 Protobuffer를 통해 직렬화를 수행하고, Socket.IO를 통해 Unity 엔진 내부의 SLAM 시스템에 전달된다. Unity 엔진의 SLAM 시스템에서는 각 이미지 데이터에 대한 ORB 피쳐를 추출하고 가장 인접한 키 프레임과 비교하여 특징점의 위치를 계산함으로써 대략적인 카메라의 포즈를 추정한다. 카메라의 포즈 정보를 기반으로 실제 환경과의 상관관계를 판단하여 실제와 최대한 유사한 맵을 만들기 위해 키 프레임 사이의 관계를 나타내는 BA(Bundle Adjustment)의 공가시성 그래프를 사용하여 현재 카메라가 보고 있는 시점에서의 맵 데이터인 지역 맵을 그린다. 누적된 드리프트를 제거하기 위해 지역 맵에서 Loop Closure Detection을 수행하고, 모든 로컬 맵들을 가산하여 얻은 전역 맵에 대해 맵 포인트를 감소시키는 최적화를 거친 후 맵 포인트는 제외하고 오직 카메라 포즈로만 현지화를 한다. 번들 조정(BA: Bundle Adjustment)은 비선형 최적화를 위한 g2o 라이브러리를 이용해 현재 프레임의 포즈는 Motion-only BA로, 최근 키 프레임의 포즈는 Local BA로, 그리고 전체적인 최적화는 Loop Closing의 포즈 그래프 최적화와 Full BA로 계산하며, 네 가지의 최적화 태스크는 각각의 스레드를 통해 독립적으로 수행된다. Motion-only BA는 맵의 키포인트들을 고정시킨 채 현재 포즈만 최적화하고, Local BA는 새로운 키 프레임이 생기면 공가시적 키 프레임의 포즈와 해당 포즈에서 보이는 키포인트들을 최적화한다. Loop Closing은 공가시성 그래프에서 키포인트를 제외하고 포즈 관점만 가지고 최적화하는 방식의 포즈 그래프 최적화를 수행하고, 포즈 그래프 최적화가 끝나면 Full BA를 통해 추가된 키 프레임과 키포인트들을 업데이트한다[2-4].

동적 라이브러리만 지원 가능한 Unity 엔진에 SLAM 알고리즘을 적용시키기 위해서는 LLVM의 Clang 빌드를 통해 SLAM 시스템을 라이브러리화하여 Unity 플러그인으로 변환해야 한다. LLVM은 컴파일러의 기반으로 프로그램을 컴파일 타임, 링크 타임, 런타임 환경에서 작성 언어에 관계없이 쉽게 최적화를 가능하게 해 준다. Clang은 LLVM을 백엔드로 사용하는 C, C++, Objective-C, Objective-C++ 등의 프로그래밍 언어를 위한 컴파일러의 프론트엔드이고, GCC 컴파일러보다 빠른 컴파일 속도, 빠른 실행 속도, 그리고 더 정확한 에러 위치 및 에러 정보를 제공해줌으로써 개발의 편의성을 높여준다. 본 논문에서는 Clang으로 빌드하여 생성된 SLAM 플러그인을 Unity 엔진에서 사용하기 위해 Unity 엔진의 'DllImport("PluginName.dll")'

함수로 임포트 시키며, SLAM 시스템의 초기화 함수 및 설정값과 이미지 전송을 위한 함수 등을 사용하여 SLAM 시스템을 제어한다. 이러한 방식으로 Unity 엔진에 내장된 SLAM 플러그인을 통해 증강현실 기술을 구현하였다.

모바일 기기에서 수집된 연속적인 이미지 데이터로부터 포즈 데이터와 포인트 클라우드 데이터가 최종적으로 Unity 엔진으로 제공되도록 하기 위해 먼저 SLAM 라이브러리로 전달된다. 카메라 포즈 데이터는 4 x 4 행렬로 카메라의 움직임을 표현할 수 있으며, 포인트 클라우드 데이터는 1 x 3 벡터 데이터로 공간상의 위치를 표현한다. 이 두 가지 데이터를 통해 SLAM 내부에서 저장되고 있는 현재 카메라 포즈와 주변 환경에 대한 시각화가 가능하다[12].

3.2 포인트 클라우드 시각화

기존의 OpenVSLAM은 Pangolin을 사용하여 미리 준비된 데이터 세트를 토대로 SLAM의 키 프레임, 카메라 포즈, 그리고 포인트 클라우드를 시각화하여 SLAM이 구동되는 모습을 보여준다. Pangolin은 OpenGL 디스플레이를 관리하고 영상 입력을 추상화하기 위한 가볍고 빠른 3D 시각화 라이브러리로, 플랫폼별 상용구를 제거하고 데이터를 쉽게 시각화하기 위한 수단으로써 컴퓨터 비전 분야에서 널리 사용되고 있다. 본 논문에서는 안드로이드 플랫폼을 타겟으로 SLAM 시스템을 설계하기 위해 Pangolin 대신 Unity 엔진을 사용하였으며, Unity API를 사용해 카메라를 제어하고 증강현실 콘텐츠를 랜더링할 수 있도록 하였다.

EuRoC MH 03 데이터 세트는 MAV(Micro Aerial Vehicle)에서 수집된 시각적 관성 데이터 세트로 스테레오 이미지, 동기화된 IMU 측정, 정확한 모션 및 구조 실측 데이터가 포함되어 있으며 Asctec Firefly hex-rotor drone에 장착된 2개의 핀홀 카메라와 관성 센서를 이용하여 기록되었다[13]. 기존의 Pangolin 뷰어는 Ubuntu 데스크탑 환경에서 구동되었고, 본 논문에서 제안하는 Unity 기반의 모바일 뷰어는 갤럭시 노트 20 Ultra 5G에서 구동되었다.

Pangolin 뷰어와 모바일 뷰어는 동일한 데이터 세트를 사용하여 SLAM 시스템을 시각화했기 때문에 포인트 클라우드의 포인트 개수는 약 32,000개로 동일하다. EuRoC MH 03 데이터 세트에 대한 시각화 성능은 Table 1과 같다.

Table 1. 데스크톱과 모바일 기기의 SLAM 시각화 성능 비교

Item	Desktop (Pangolin)	Mobile Device (Unity)
CPU Usage (%)	99	28
Memory Usage (%)	23.1	8.3
Screen Resolution	1920 x 1080	1280 x 720
Frame rate (FPS)	Avg. 30	Avg. 25

Pangolin 기반의 데스크톱 환경과 Unity 기반의 모바일 환경에서 OpenVSLAM을 이용한 SLAM 시각화 성능에 대해 CPU 사용량, 메모리 사용량, 해상도, 그리고 fps를 비교한 결과는 표 3과 같다. Pangolin 뷰어를 구동시킬 때 CPU 사용량이 99%, 메모리의 사용량이 23.1% 인 반면, Unity 기반의 모바일 환경에서는 CPU와 메모리의 사용량은 각각 28%와 8.3%로 더 적게 소비되었다.

이 실험을 통해 Unity기반의 모바일 환경에서도 Pangolin 기반의 데스크톱 환경에서 OpenVSLAM을 이용하여 시각화하는 것과 거의 대등한 SLAM 시각화가 가능한 것을 알 수 있으며, 데스크톱 환경과 달리 CPU에 대한 부하가 적어 멀티태스킹 관점에서도 효율적일 것으로 예상된다. 다만, frame rate이 데스크톱 환경보다 다소 낮게 나타나 실시간성은 떨어진다.

3.3 FastAPI Server

SocketIO는 브라우저와 서버 간의 양방향 이벤트 기반의 통신을 가능하게 하는 라이브러리로, 웹 소켓을 통해 안정성, 자동 재접속, 패킷 버퍼링, 멀티플렉싱 등의 기능을 제공한다. 클라이언트는 서버와 브라우저 간에 전이중 및 저지연 채널을 제공하는 통신 프로토콜인 웹 소켓 연결을 설정하고, 웹 소켓 연결을 할 수 없는 경우에는 HTTP 롱 폴링으로 대체한다. Protocol Buffer를 의미하는 Google의 Protobuffer는 구조화된 데이터를 직렬화하는 데 유용한 자료구조이며, 다양한 프로그래밍 언어에 대해 SDK를 제공한다. 데이터 직렬화는 유선 통신 또는 데이터 저장을 목적으로 하는 통신 프로그램을 개발할 때 주로 필요한 과정이다. Protobuffer를 사용하기 위해서는 .proto 파일을 컴파일하기 위해 필요한 프로토콜 컴파일러와 사용하고자 하는 프로그래밍 언어에 대한 Protobuffer 런타임을 설치해야 한다.

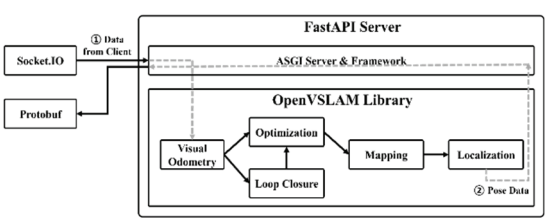

Fig. 2. FastAPI 기반의 매핑 서버에 의한 SLAM 수행 절차.

Fig. 2는 FastAPI 서버 시스템의 전체적인 작업흐름도이다. 서버 측에서는 Node.js에서 C++로 구현된 공간 데이터 처리, 시각화, 재구성 라이브러리를 사용하기보다는 Python 서버에서 구현하는 것이 더 편리하다. 따라서 본

논문에서는 OpenVSLAM 기반의 모바일 혼합현실 시스템을 플러그인으로 적용시키기 위해 Socket.IO 통신 모듈을 기존의 Node.js의 이벤트 핸들러 구조와 동일하게 구성해 FastAPI에서 사용하였다. 모바일에서 수집한 데이터를 Protobuffer를 통해 직렬화하고 FastAPI 서버와 SoketIO를 이용하여 모바일 애플리케이션과 서버의 SLAM 알고리즘 간의 데이터 통신을 구현하였다. Socket.IO를 통해 클라이언트로부터 이미지 및 IMU 데이터를 받아 Protobuffer로 역직렬화를 하고, 역직렬화한 데이터를 OpenVSLAM 알고리즘의 입력 데이터로 넣어 카메라의 포즈 데이터를 받아 다시 Socket.IO를 통해 클라이언트에 전송한다.

Fig. 3은 OpenVSLAM 기반의 모바일 증강현실 시스템의 전체 작업흐름도이다. 사용자 클라이언트의 모바일 기기 카메라로부터 이미지 및 IMU 데이터를 수집하여 맵핑을 위한 Python FastAPI 서버로 전송한다. 맵핑 서버에서는 OpenVSLAM 알고리즘을 통해 맵 데이터와 카메라 포즈 데이터를 얻어 사용자 클라이언트의 Unity 엔진의 안드로이드 애플리케이션에 전송해 전역 맵과 위치를 시각화하여 보여준다.

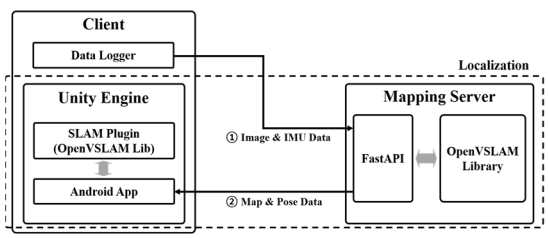

Fig. 3. Mapping 서버 기반 SLAM 수행을 위한 모바일 기기와 mapping 서버의 상호작용.

4. 실험 및 결과

본 논문에서 제안하는 모바일 SLAM 시스템의 기능과 성능 평가를 위해 실제 실내 공간에서 SLAM을 수행하여 시각화 결과 및 성능을 평가하였다.

4.1 실험 환경

본 논문을 통해 제안하는 모바일 SLAM 시스템을 통해 실제 환경에서 맵을 생성하고 현지화함으로써 검증을 수행하고자 하였으며, 이를 위해 실험 환경을 사전에 촬영하여 이미지 데이터와 IMU 데이터를 수집하였다.

Fig. 4는 해당 실험을 위해 사용된 실내(사무실) 실험 환경이다. 실내 공간의 면적은 약 65m^{2}이고, 갤럭시 노트 20 Ultra 5G의 카메라를 사용하여 촬영하였다.

Fig. 4. SLAM 시스템 평가를 위한 실내 공간.

4.2 실험 결과

촬영한 영상은 Frames.m4v으로 저장되고, 해당 영상을 타임 스탬프(Frames.txt)를 기준으로 이미지로 변환한 cam0, 위치 데이터를 저장하고 있는 GPS.txt, 가속도 데이터 Accel.txt, 각속도 데이터 Gyro.txt, 가속도 데이터와 각속도 데이터를 연산하여 얻은 관성 데이터를 저장하고 있는 IMU.txt, 그리고 카메라 설정값을 담고 있는 config.yaml 파일이 생성된다. 모바일 카메라를 통해 촬영한 이미지 데이터와 IMU 데이터로 유니티 엔진을 통해 sparse map (.bin)과 sparse point cloud (.ply) 파일들이 생성된다.

Fig. 5. 모바일 SLAM 시스템에 의한 맵 생성 결과.

Fig. 5는 Fig. 4의 실내 공간을 약 2분 30초 동안 사전 촬영하여 실제 실내 공간과 일치하는 포인트 클라우드 맵을 부분 캡처하여 비교한 결과이다.

해상도는 1280 x 720, 촬영한 실내 공간의 포인트 클라우드는 약 54,000개, fps는 평균 40으로 측정되었으며 배터리 소모량은 약 1% 이하로 매우 적게 나타났다.

Table 2. 모바일 SLAM 시스템 성능

Item	Description
Display Resolution	1280 x 720
Recording Time (s)	150
Number of Points	약 54,000 개
Frame rate (FPS)	평균 40

5. 결 론

본 논문에서는 연구자들의 불필요한 자체 연구를 줄이고 SLAM 분야의 보다 빠른 성장에 기여하고자 오픈소스를 목적으로 접근성이 좋은 모바일 환경에서 SLAM 기반의 증강현실 시스템을 설계하였다. 이를 위해 모바일 기기의 카메라를 이용해 이미지 데이터를 얻어 Tracking, Local 및 Global Mapping, Loop Closure Detection 그리고 Localization 기능을 유니티 엔진에서 사용하기 위해 dll 플러그인을 생성하고, SLAM 기능을 모바일 애플리케이션에서 확인할 수 있게 구현하였다.

추후에 본 시스템을 기반으로 서버를 구축해 여러 클라이언트 간의 Collaborative SLAM을 구현해 인건비, 개발에 소요되는 시간 등 개발 비용을 줄여 효율성을 높이고, SLAM 기능 중 Mapping, Optimization과 같이 연산량이 많은 기능들은 서버에서 구현시켜 모바일 애플리케이션에서 보여줌으로써 모바일 환경에서의 SLAM의 성능을 향상시키고자 한다. 또한, Mobile 카메라의 IMU 데이터를 얻어 정밀한 맵을 그림으로써 실시간으로 보다 정확한 현지화를 할 수 있을 것으로 기대한다.

참고문헌

1. Esparza-Jiménez, Jorge O., Michel Devy, and José L. Gordillo, "Visual EKF-SLAM from Heterogeneous Landmarks" *Sensors* 16, no. 4: 489, 2016.
2. H. C. Roh, C. H. Sung and M. J. Chung, "Rapid SLAM using simple map representation in indoor environment," *The 19th Korea-Japan Joint Workshop on Frontiers of Computer Vision*, 2013, pp. 225-229,
3. Jung, Sungwook, Duckyu Choi, Seungwon Song, and Hyun Myung, "Bridge Inspection Using Unmanned Aerial Vehicle Based on HG-SLAM: Hierarchical Graph-Based SLAM" *Remote Sensing* 12, no. 18: 3022, 2020.
4. Fan, T., Wang, H., Rubenstein, M., Murphey, T., "CPL-SLAM: Efficient and Certifiably Correct Planar Graph-Based SLAM Using the Complex Number Representation," TRO, vol. 36, pp. 1719-1737, 2020.
5. Myriam Servières, Valérie Renaudin, Alexis Dupuis, Nicolas Antigny, "Visual and Visual-Inertial SLAM: State of the Art, Classification, and Experimental Benchmarking", Journal of Sensors, vol. 2021, Article ID 2054828, 26 pages, 2021.
6. Engel, J.; Koltun, V.; "Cremers, D. Direct Sparse Odometry," TPAMI, vol. 40, pp. 611-625, 2018.
7. Rosinol, A.; Abate, M.; Chang, Y.; Carlone, L., "Kimera: an Open-Source Library for Real-Time Metric-Semantic Localization and Mapping, " IEEE Int. Conf. Robot. Autom. (ICRA), May. 2020, pp. 1689-1696.
8. G. Klein and D. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, 2007, pp. 225-234.
9. Engel J., Schöps T., Cremers D., "LSD-SLAM: Large-Scale Direct Monocular SLAM," Lecture Notes in Computer Science, vol. 8690. Springer, Cham pp. 834-849. 2014.
10. C. Forster, M. Pizzoli and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 15-22.
11. Piao, Jin-Chun, and Shin-Dug Kim, "Adaptive Monocular Visual–Inertial SLAM for Real-Time Augmented Reality Applications in Mobile Devices" *Sensors* 17, no. 11: 2567, 2017.
12. J. E. Song, J. Kook, "Building a Mobile AR System Based on Visual SLAM," Journal of the Semiconductor & Display Technology, vol. 20, no. 4. pp. 96-101, 2021.
13. Burri, M.; Nikolic, J.; Gohl, P.; Schneider, T.; Rehder, J.; Omari, S.; Achtelik, M.W.; Siegwart, R. The EuRoC micro aerial vehicle datasets. Int. J. Robot. Res. 2016, vol. 35, pp. 1157–1163.

접수일: 2022년 3월 12일, 심사일: 2022년 3월 18일, 게재확정일: 2022년 3월 25일