논문 16-41-11-24 The Journal of Korean Institute of Communications and Information Sciences '16-11 Vol.41 No.11
http://dx.doi.org/10.7840/kics.2016.41.11.1490

병렬 분산 처리를 이용한 영상 기반 실내 위치인식 시스템의 프레임워크 구현

권 범◆, 전 동 현*, 김 종 유*, 김 정 환*, 김 도 영*, 송 혜 원*, 이 상 훈°

Framework Implementation of Image-Based Indoor Localization System Using Parallel Distributed Computing

Beom Kwon◆, Donghyun Jeon*, Jongyoo Kim*, Junghwan Kim*,
Doyoung Kim*, Hyewon Song*, Sanghoon Lee°

요 약

본 논문에서는 인메모리(In-memory) 병렬 분산 처리 시스템 Apache Spark(이하 Spark)를 활용하여 사용자에게 실시간 측위 정보를 제공할 수 있는 영상 기반 실내 위치인식 시스템을 제안한다. 제안하는 시스템에서는 사용자에게 실시간 측위 정보를 제공하기 위해서, Spark를 이용한 영상 특징점 추출 알고리즘의 병렬 분산화를 통해 알고리즘 연산 시간을 단축시킨다. 하지만 기존의 Spark 플랫폼에서는 영상 처리를 위한 인터페이스가 존재하지 않아, 영상 처리와 관련된 연산을 수행하는 것이 불가능하였다. 이에 본 논문에서는 Spark 영상 입출력 인터페이스를 구현하여 측위 연산을 위한 영상 처리를 Spark에서 수행 가능하게 하였다. 또한 무손실 압축(lossless compression)기법을 이용하여 특징점 기술자(descriptor)를 압축된 형태로 데이터베이스에 저장하여, 대용량의 실내 지도 데이터를 효율적으로 저장 및 관리하는 방법을 소개한다. 측위 실험은 실제 실내 환경에서 수행하였으며, 싱글 코어(Single-core) 시스템과의 성능 비교를 통해 제안하는 시스템이 최대 약 3.6배 단축된 시간으로 사용자에게 측위 정보를 제공 할 수 있다는 것을 입증하였다.

Key Words : Image-based localization, Apache Spark, In-memory parallel distributed computing, Lossless compression, Lempel-Ziv

ABSTRACT

In this paper, we propose an image-based indoor localization system using parallel distributed computing. In order to reduce computation time for indoor localization, an scale invariant feature transform (SIFT) algorithm is performed in parallel by using Apache Spark. Toward this goal, we propose a novel image processing interface of Apache Spark. The experimental results show that the speed of the proposed system is about 3.6 times better than that of the conventional system.

※ 이 연구는 방위사업청 및 국방과학연구소의 재원에 의해 설립된 신호정보 특화연구센터 사업의 지원을 받아 수행되었음.
◆ First Author : Yonsei University, Department of Electrical and Electronic Engineering, hsm260@yonsei.ac.kr, 학생회원
° Corresponding Author : Yonsei University, Department of Electrical and Electronic Engineering, slee@yonsei.ac.kr, 종신회원
* Yonsei University, Department of Electrical and Electronic Engineering, {jdh3577, jongky, junghwan.kim, tnyffx, shw3164}@yonsei.ac.kr,
논문번호 : KICS2016-08-201, Received August 17, 2016; Revised October 8, 2016; Accepted October 17, 2016

Ⅰ. 서 론

최근 위치 측위가 가능한 단말의 보급 확산과 스마트폰의 위치 측위 플랫폼 개방으로 위치기반 서비스에 대한 연구가 활발히 진행되고 있다. 또한 영상 촬영 기능이 탑재된 다양한 모바일 기기의 등장으로 언제 어디서나 손쉬운 촬영이 가능하게 됨에 따라, 영상으로부터 위치 정보를 추정하여 사용자에게 유용한 정보를 제공하는 서비스에 대한 수요가 급격하게 증가하는 추세이다[1-3].

대부분의 영상 기반의 실내 위치인식 시스템에서는 사용자가 촬영한 영상에서 추출한 특징점과 영상 측위 데이터베이스에 저장된 특징점 사이의 매칭(matching) 과정을 통해 사용자의 위치 정보를 추정한다[4]. 이때, 영상 측위 데이터베이스는 고가의 영상 수집 장치를 이용하여 구축되는데, 실내 지도 정보와 영상 수집 장치에 의해 촬영된 영상의 특징점들을 이용한 3차원 구조 복원 결과가 하나의 쌍(pair)으로 함께 저장된다. 영상 기반 실내 측위의 정확도는 영상 측위 데이터베이스를 구성하는 특징점들에 영향을 받기 때문에, 정확한 영상 기반 측위를 위해서는 대규모로 실내 영상을 촬영하고 이를 이용하여 측위 데이터베이스를 구축하여야 한다[5,6].

그밖에도 영상기반 위치 인식을 위해서는 기본적으로 측위 데이터베이스 및 사용자가 촬영하는 영상들에 대해 공간 및 시간 변화 등에 강인한 특징점을 추출하는 알고리즘, 다수의 영상들에서 추출된 특징점들 간 유사점을 찾는 매칭 알고리즘, 매칭점을 이용해 여러 뷰(view)에 대한 3차원 모델을 구축하는 알고리즘, 카메라의 센터(center) 및 사용자의 위치를 계산하는 알고리즘 등이 필요하다. 따라서 영상 정보를 활용한 위치 인식의 경우, 기존의 무선 신호의 세기를 이용하는 위치 인식 시스템[7]보다 알고리즘 복잡도가 높아 연산 시간이 많이 소모 되는 단점이 있다.

그러므로 영상기반 실내 측위 시스템에서는 측위의 정확도도 중요하지만, 서버에서 수행하는 측위에 대한 연산 속도 또한 중요하다. 측위 과정 중에서 영상 특징점 추출 부분에서 가장 많은 시간이 소비되기 때문에, 대용량 영상에 대해 측위를 수행할 경우 병목현상이 발생하여 사용자에게 측위 정보 제공에 있어 지연이 발생할 수 있다. 이에 본 논문에서는 측위 연산 서버를 Apache Spark(이하 Spark)[8]를 활용하여 여러 대의 컴퓨터로 구성된 분산 시스템으로 구성하여, 영상 특징점 추출 알고리즘을 분산 처리 할 수 있게 하였다.

Spark는 대용량의 데이터를 분산 클러스터에서 고속으로 처리할 수 있도록 도와주는 빅데이터 분산 컴퓨팅 플랫폼 중의 하나로, 인메모리(In-memory) 처리를 통해 디스크 I/O 병목을 줄여주기 때문에 Hadoop[9]보다 연산이 빠르고 지연 속도가 낮은 장점이 있다. 하지만 기존의 Spark에서는 JPEG(joint photographic experts group), PNG(portable network graphic)와 같은 표준 영상 데이터를 다루는 인터페이스가 없어 영상 파일을 텍스트(text) 파일로 변환하여 사용한다[10]. 따라서 다수의 대용량 영상에 대해 처리를 하게 되는 경우, 영상 파일을 텍스트 파일로 변환하는데 오랜 시간이 걸리기 때문에 매우 비효율적이다. 그리고 영상 파일에서 텍스트 파일로 포맷이 변환되기 때문에 기존의 영상처리 관련 알고리즘을 텍스트 파일 포맷과의 호환성을 고려하여 재설계하여야하기 때문에 구현적인 관점에서도 불편함이 존재한다.

Spark의 이러한 단점을 극복하기 위해서, [11]에서는 Hadoop의 InputFormat, RecordReader class를 기반으로 Java용 영상처리 라이브러리인 JavaCV와 비디오 디코딩 라이브러리인 FFMPEG를 사용하여 Spark에서 비디오 데이터 프로세싱을 위한 인터페이스를 제안하였지만, 오픈소스로 공개되지 않았으며 비디오 영상이 아닌 대용량 이미지에 대해서는 일반화가 되지 않았다. 그리고 [12]은 대용량 영상 데이터를 처리하기 위해 Hadoop Image Processing Interface (HIPI)를 개발하여 대용량 영상에 대한 입출력 프레임워크를 제안하였지만, Spark와의 호환성 문제가 아직 해결되지 않아 Spark에서는 HIPI를 사용할 수 없다는 한계점이 존재한다. 또한 [13]은 Spark와 Caffe[14]를 연동하여 영상에 대해 분산 기계 학습이 가능하도록 하는 프레임워크를 제안하였는데, 입력 영상을 Caffe를 통해 받은 기계 학습용 데이터 포맷으로만 활용이 가능하여 어플리케이션이 이미지 분류로 제한되는 한계점이 존재한다.

이에 본 논문에서는 영상 특징점 추출 알고리즘의 병렬 분산 처리를 위해, Spark에서 대규모 영상 데이터를 효율적으로 다루기 위한 영상 처리 인터페이스를 구현하였다. 구현한 인터페이스는 Hadoop의 입출력 함수를 활용하여 대용량 영상 이미지를 하나의 bundle 형식의 포맷으로 만들고, 이것을 Spark에서 각 영상에 대해 병렬적으로 디코딩하는 과정을 포함한다.

본 논문의 공헌점은 다음과 같이 두 가지로 요약할 수 있다.

- 영상기반 위치인식 시스템에서 가장 많은 시간이 소비되는 특징점 추출 알고리즘을 병렬 분산 처리

할 수 있는 Spark 기반 실내 위치인식 시스템을 제안한다.

- JPEG, PNG 등과 같은 표준 영상 데이터를 다룰 수 있는 Spark 영상 처리 인터페이스를 제안한다. 제안하는 인터페이스는 OpenCV 인터페이스를 기반으로 하고 있어 높은 확장성을 제공한다.

Ⅱ. 영상기반 실내 위치인식 고속화 프레임워크

2.1 제안하는 전체 프레임워크

본 논문에서 제안하는 영상 기반 실내 위치인식 시스템은 그림 1과 같이 구성되어 있다. 오프라인 단계(Off-line phase)는 데이터베이스 구축을 목적으로 한다. 우선 영상 데이터베이스 수집 장치 카메라를 통해 촬영된 데이터베이스용 영상들을 측위 서버로 전송한다. 서버는 전송받은 데이터베이스용 수집 영상들에 대해 특징점을 추출하고, Multi-view SFM(Structure From Motion) 알고리즘[15][16]의 3차원 구조 재복원 과정[17]에 의해 만들어진 3차원 점과 그에 대응되는 특징점 기술자들(descriptors)을 데이터베이스에 실내 지도와 함께 저장함으로써 영상 측위 데이터베이스를 구축한다. 이때, 데이터베이스의 효율적인 저장 용량 관리를 위해서 무손실 압축(lossless compression)기법을 이용하여 각 특징점 기술자들을 압축된 형태로 데이터베이스에 저장한다. 본 논문에서 데이터베이스 구축을 위해 사용된 무손실 압축 기법은 사전(dictionary) 부호화 기반의 Lempel-Ziv(LZ) 알고리즘과 그에 대한 변종 알고리즘들이다. 무손실 압축 알고리즘에 따른 데이터베이스의 저장 용량 효율성은 4절 실험결과에서 자세히 다룬다.

온라인 단계(On-line phase)에서는 사용자가 측위를 위해 촬영한 영상을 서버에 전송하면서 시작된다. 서버에서는 사용자로부터 받은 영상에 대해, 신뢰도가 높은 특징점들을 추출한다. 그리고 추출된 특징점들을 데이터베이스에 있는 특징점들과 매칭 과정을 통해 대응하는 3차원 점을 찾는다. 찾아진 3차원 점 정보로부터 카메라의 위치 즉, 사용자의 위치정보를 추정하고 추정 결과를 사용자 단말기에 제공한다.

2.2 오프라인 단계: 실내측위 데이터베이스 생성

2.2.1 영상 특징점 추출

본 논문에서는 영상에서 포인트(point) 기반의 특징점 추출하기 위해서 SIFT(Scale Invariant Feature Transform)[18]를 이용하였다. SIFT는 영상의 크기, 조명, 평행이동, 회전에 대해 강인한 특징점을 추출할 수 있다는 장점을 갖고 있다.

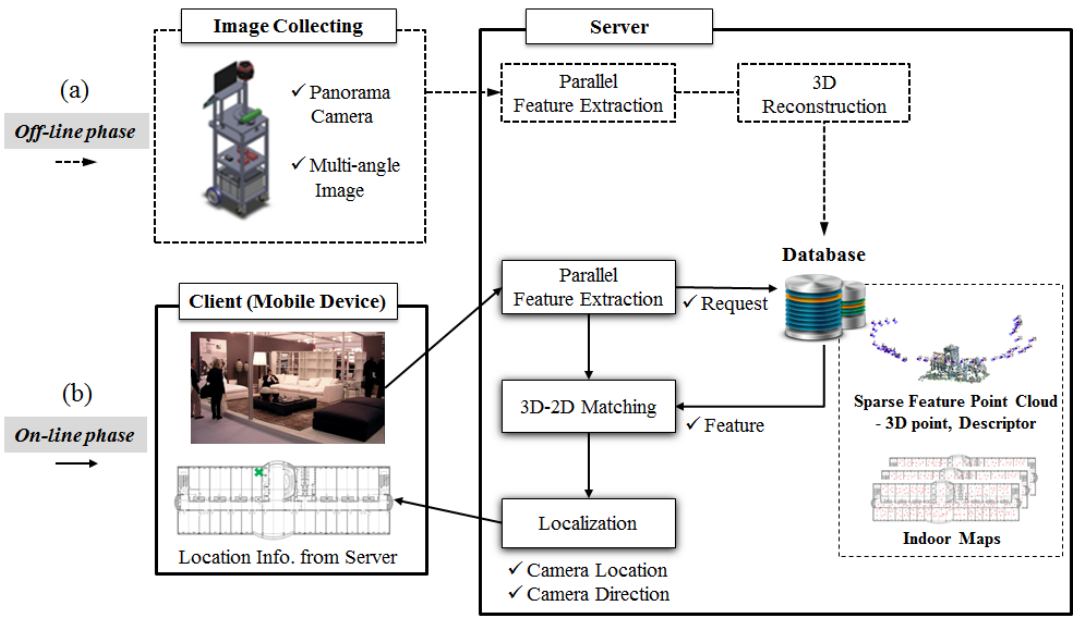


그림 1. 전체 프레임워크; (a) 오프라인 단계: 스파크 기반 실내 측위 데이터베이스 생성, (b) 온라인 단계: 스파크 기반 사용자 위치 인식.
Fig. 1. Overall framework; (a) off-line phase: Spark based indoor localization database generation, (b) on-line phase: Spark based user localization.

SIFT를 이용한 특징점 추출은 먼저 영상의 특징점 후보를 찾는다. 특징의 후보 점을 검출하는 방법은 입력 영상의 스케일(scale)을 줄여가면서 정해진 분산 값의 가우시안 블러 필터(Gaussian blur filter)를 적용하여 가우시안 피라미드의 연속 영상을 만들고, 스케일별로 두 개의 가우시안 블러 영상의 차인 DoG(differential of Gaussian) D를 식 (1)과 같이 계산한다.

$$D(x,y,\sigma) = (G(x,y,k\sigma) - G(x,y,\sigma)) * I(x,y) \\ = L(x,y,k\sigma) - L(x,y,\sigma) \quad (1)$$

여기서 G는 가우시안 블러 필터, σ는 크기 인자(scale factor), $I(x,y)$는 입력 영상 그리고 L은 스케일 공간 영상을 의미한다. 식 (1)을 통해 DoG를 계산한 후, 현재의 DoG의 픽셀과 이웃한 두 개의 DoG 사이에서 3×3×3영역의 26개 주변 픽셀을 비교하여 극댓값 또는 극솟값을 구한다. 이렇게 구해진 극댓값 또는 극솟값이 특징점의 후보가 된다.

정해진 특징점의 방향과 크기를 결정하기 위해 특징점 주변 픽셀의 기울기 방향(Gradient orientation)에 따른 히스토그램(histogram)을 10°씩 36단계로 구한다. 36단계 중 임계값을 넘는 방향을 특징점의 기준 방향으로 결정한다. 특징점이 속해 있는 각 이미지 샘플 L에서 Gradient 벡터는 식 (2)로 표현할 수 있다.

$$Gradient = \begin{bmatrix} L(x+1,y) - L(x-1,y) \\ L(x,y+1) - L(x,y-1) \end{bmatrix} \quad (2)$$

Gradient 벡터의 크기 $m(x,y)$와 방향 $\theta(x,y)$는 식 (3)과 (4)로부터 각각 구할 수 있다.

$$m(x,y) = \sqrt{\begin{matrix}(L(x+1,y) - L(x-1,y))^2 \\ +(L(x,y+1) - L(x,y-1))^2\end{matrix}} \quad (3)$$

$$\theta(x,y) = \tan^{-1}\left(\frac{L(x+1,y) - L(x-1,y)}{L(x,y+1) - L(x,y-1)}\right) \quad (4)$$

특징점의 방향에 맞추어 결정된 영역에 있는 픽셀들을 4×4배열에 할당한 후 각 배열을 8방향으로 재구성한다. 그림 2의 예시에서 볼 수 있듯이 재구성된 128개의 값(128bin)이 상기 특징점의 기술자 v가 된다. 이러한 일련의 과정이 한 이미지의 모든 특징점들에 대해서 기술자 벡터 $v = \{v_1, v_2, \cdots\}$를 구하는 작업이 수행되기 때문에 연산의 복잡도가 높다.

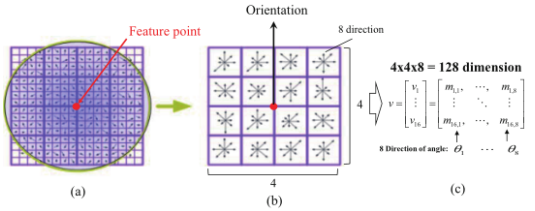


그림 2. SIFT 기술자의 구조; (a) 기울기 방향과 특징점들 주변의 크기값, (b) 표준 방향에 대한 4x4 그리고 8-레벨의 양자화, (c) 각 특징점들에 대한 128 차원의 기술자.
Fig. 2. Structure of SIFT descriptor; (a) gradient orientation and amplitude of around of feature points, (b) 4×4 range and 8-level quantization with respect of standard orientation, (c) descriptor of 128 dimension for each feature points.

제안하는 특징점 추출 알고리즘 SIFT의 병렬 분산 처리는 본 논문에서 구현한 Spark 영상 처리 프레임워크를 통해 병렬화 되어 그림 3과 같이 수행된다. 총 N개의 실내 이미지들이 있다고 가정했을 때, N개의 이미지들을 Spark Image Sequence File <Image file name, Byte of Image file> 형태로 입력받는다. 여기서 Image file name을 Key값, 각 영상들의 binary 포맷을 Value 값을 의미한다. 그리고 Spark driver 노드는 Spark worker 노드들에게 영상 데이터들을 균등하게 나누어 데이터 병렬화를 시킨다. 예를 들면, 총 3대의 Spark worker 노드로 분산 클러스터가 구성되어 있다면 각 Spark worker 노드당 n=N/3개씩 영상을 분배 받는다.

각 Spark worker 노드는 분배된 영상 파일들에 대해 픽셀 배열단위로 처리 가능할 수 있게 영상 binary를 디코딩 한다. Spark는 함수형 프로그래밍 언어인 스칼라(Scala)를 기반으로 구현되어 있기 때문에 함수 자체를 객체화 시켜서 각 Spark worker 노드의 코어별로 함수를 넘겨주어 실행하는 것이 가능하다. 본 논문에서는 Spark에서 제공하는 map 함수 중 하나인 flatMap를 이용하여 각각의 Spark worker 노드로 분배된 영상 데이터 셋에 대해 SIFT를 Key값(image file name) 기준으로 호출할 수 있게 하였다.

flatMap은 디코딩된 이미지 픽셀 배열 타입의 Value 데이터 셋을 인풋(Input)으로 받고, SIFT를 수행 한 후 특징점 기술자 벡터들을 반환한다. 결과적으로 각 Spark worker 노드에서 병렬적으로 연산된 최종 결과 값들은 각 이미지들의 <Image file name, Feature descriptors>로 즉, <Key, Value> 포맷으로 반환되고 각 이미지별 특징점들로 데이터베이스에 저장된다. 그리고 본 논문에서는 효율적으로 데이터베이스의 저장 용량을 관리하기 위해서 LZ-알고리즘 기

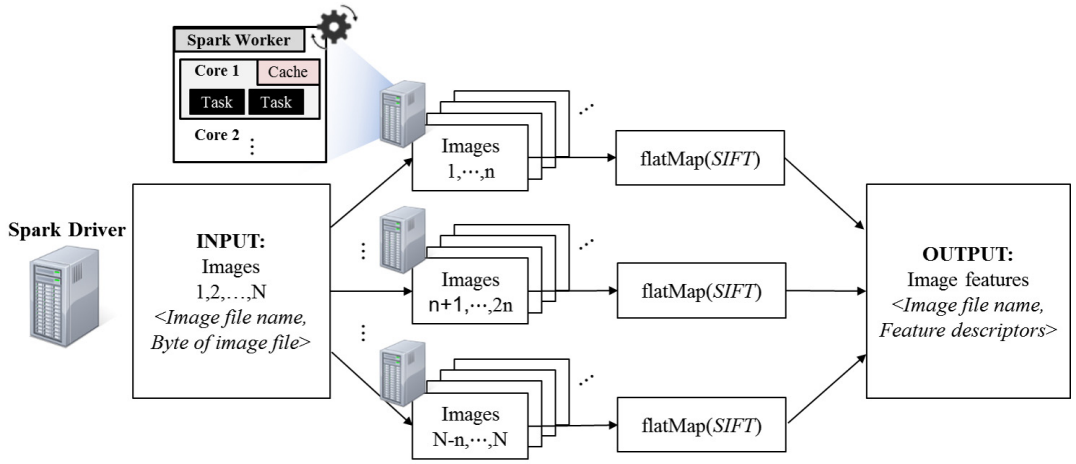


그림 3. 제안하는 스파크 영상 처리 인터페이스를 통한 병렬 분산 SIFT 알고리즘.
Fig. 3. Parallel distributed SIFT algorithm via the proposed Spark image processing interface.

반의 무손실 압축 기법을 이용하여 각 이미지별 특징점들을 압축하고, 압축된 특징점 데이터들을 데이터베이스에 저장한다.

2.2.2 영상 특징점을 활용한 3차원 점 기반의 측위 데이터베이스 생성

앞서 데이터베이스용으로 촬영된 영상에 대해 병렬적으로 특징점을 추출한 후, 유사한 특징점 기술자를 가진 특징점들을 찾기 위해 그림 4와 같이 영상들 간의 매칭 작업을 수행한다. 이를 기반으로 3차원 좌표점을 생성하여 실내 공간 데이터베이스를 구축한다. 본 논문에서는 하나의 카메라로 얻은 다수 영상으로부터 하나의 3차원 구조를 복원하기 위해 SFM 알고리즘[17]을 사용하였다.

3차원 구조를 복원하기 위해서는 우선, 그림 4와 같이 서로 다른 위치에서 단안 카메라를 통해 두개 이상의 영상들을 얻고, 영상들 사이의 매칭점을 구한다. 그리고 매칭점들 사이의 epipolar geometry[15]를 이용해 모션 정보 P를 구하고, 이에 대응되는 영상의 특징점 x를 이용하여 식 (4)을 기반으로 3차원 cloud point X로 복원함으로써 측위 데이터베이스를 생성한다.

$$x = PX,\ where\ P = K[R|t] \tag{4}$$

여기서 K는 Calibration matrix로 카메라의 초점거리, 카메라의 왜곡정도, 촬영된 이미지의 중심점을 나타내는 카메라 내부 파라미터를 포함한다. R은 카메라의 회전 정보를 나타내는 rotation matrix를 의미하고, t는 카메라의 이동 정보를 나타내는 translation matrix를 의미한다[19]. 그림 5는 SFM 알고리즘을 기반으로 복원된 3차원 point cloud 데이터베이스의 예시이다.

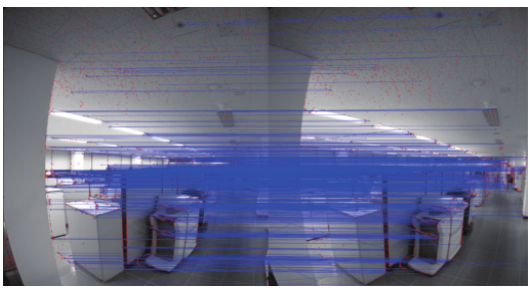


그림 4. 측위 데이터베이스 생성을 위한 카메라 움직임에 따른 SIFT 특징점 매칭에 관한 예제.
Fig. 4. Example of SIFT feature matching according to camera motion for making localization database.

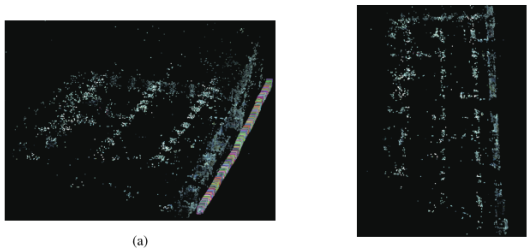


그림 5. 오프라인 단계의 3차원으로 복원된 포인트-클라우드 예제; (a) 카메라들과 3D 모델, (b) 카메라들을 표현하지 않은 상태에서 3D 모델의 탑뷰.
Fig. 5. Example of 3D reconstructed point-cloud of off-line phase; (a) 3D model with cameras, (b) top-view of 3D model without cameras.

2.3 온라인 단계: 사용자 위치 인식

카메라 센터 또는 사용자의 위치는 영상 기반 실내 측위 데이터베이스 상의 3차원 모델을 사용자가 촬영한 2차원 평면에 투영하여 3차원 점과 2차원 영상 위의 특징점의 연장선이 만나는 지점을 통해 추정할 수 있다. 즉, 사용자가 촬영한 측위용 영상에서 특징점 추출을 한 후, 측위 데이터베이스와의 매칭 과정을 통해 3D-2D correspondence set를 구하고, 식 (5)를 통해 Projection matrix를 계산하여 사용자의 위치 $[X_c, Y_c, Z_c]$를 추정할 수 있다[20].

$$-R^T t = [X_c,\ Y_c,\ Z_c]^T \tag{5}$$

사용자의 영상기반 측위 과정에는 사용자의 측위용 영상에 대한 특징점 추출, 측위용 촬영 사진에서 추출된 특징점과 측위 데이터베이스 내 특징점 간의 매칭을 위한 3D-2D 매칭, 그리고 매칭 이후에 식 (5) 기반 사용자 위치 $[X_c, Y_c, Z_c]$ 계산[21]으로 크게 세 가지 연산들이 수행되는데, 온라인 단계 측위 연산 과정에 있어 각각 83%, 15%, 2%의 연산 비율을 차지한다. 따라서 오프라인 단계뿐만 아니라 온라인 단계에서도 측위 시간의 대부분은 영상 특징점 추출 알고리즘 연산 부분에서 소비되기 때문에, 그림 3에서 제안하는 Spark 영상 처리 프레임워크를 활용하여 여러 사용자에 대한 SIFT 알고리즘 분산 병렬화를 적용하였다.

Ⅲ. 제안하는 Spark 영상 처리 인터페이스

기존의 Hadoop은 HDFS(Hadoop Distributed File System)를 통해 스토리지(storage)를 경유하기 때문에, 병렬 연산 후 디스크 I/O 병목으로 인해 느리다는 단점이 있다. 이에 비해 Spark는 분산 클러스터에서 각 노드의 캐싱을 활용한 인메모리 처리를 기반으로 하는 시스템이기 때문에 좀 더 빠르고 지연 속도가 낮은 처리가 가능하다[8]. Spark의 경우, Hadoop 프레임워크를 기반으로 인메모리 분산 프로세싱 기술을 적용 및 발전시킨 것으로 대용량 데이터를 연산하는 데에 있어 뛰어난 성능을 보여준다. 그러나 Spark는 영상 처리에 적합한 API(Application Programming Interface)를 갖고 있지 않아, 영상 데이터를 연산하는 것에는 어려움이 있다.

이에 본 논문에서는 Spark가 갖는 영상 처리에 대한 한계점을 극복하기 위해, 그림 6과 같이 새로운

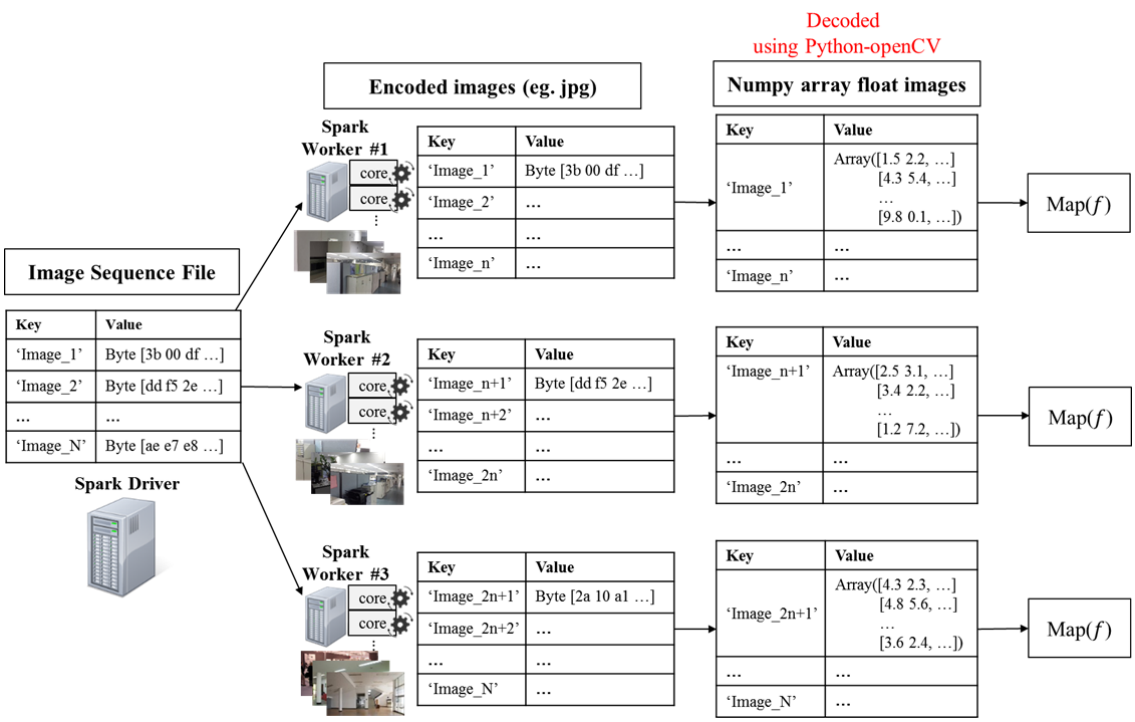


그림 6. 제안하는 스파크 영상 처리 인터페이스
Fig. 6. The proposed Spark image processing interface.

Spark 영상 처리 인터페이스를 제안한다. 그리고 Spark에서 대규모 이미지들을 입력으로 읽을 수 있도록 각 이미지들에 대해 Hadoop I/O함수인 SequnceFile.Writer 인터페이스를 활용해 <Key, Value> 데이터의 리스트 형식으로 구성된 Image Sequence File 포맷을 만들었다. 제안하는 Image Sequence File 포맷에서 Key 값은 이미지의 식별자로써 파일 이름 [Text] 형태로 저장된다. 그리고 Value 값은 해당 이미지에 대한 binary 포맷인 [ByteWritable] 형태로 저장된다.

Image Sequence File을 통해 읽은 N개의 인코딩된 표준 이미지들은 Spark driver에서 한꺼번에 load되고, 각 Spark worker 노드에 균등하게 분배된다. 그리고 각 Spark worker 노드에서는 분배받은 이미지들에 대해 병렬적으로 Python-OpenCV 함수로 디코딩을 수행하고, Numpy array package를 통해 픽셀 단위로 이미지를 제어할 수 있도록 float 타입의 이미지 배열 형태로 변환한다. 기존의 모든 이미지는 표준 형식, 예를 들어 JPEG, PNG 등으로 저장되지만 제안하는 프레임워크는 입력 이미지를 float 타입의 배열 형태로 디코딩하므로 이미지의 픽셀 값에 대한 계산 등이 자유롭게 수행될 수 있다. 따라서 디코딩된 이후의 데이터는 OpenCV 라이브러리를 사용해 영상 처리 알고리즘들을 적용할 수 있는 구조가 된다. 또한 Spark 내 병렬화 함수에 영상 처리 알고리즘을 넘겨줌으로써 이미지들에 대한 영상 처리 알고리즘을 병렬적으로 수행 할 수 있다. 그리고 Spark 내 병렬화 함수는 각 Spark worker 노드가 가지고 있는 코어들을 활용해 각 코어 당 하나의 thread를 생성하여 다수의 병렬 job들을 수행하는 멀티 코어 프로세싱을 지원한다.

Ⅳ. 실험결과

4.1 실험환경

본 논문에서는 영상 기반 실내 측위의 분산 처리를 위해 Spark driver 노드 1대, Spark worker 노드 3대 (총 4대)의 컴퓨터를 이용하였으며, 컴퓨터의 사양 및 실험용 소프트웨어 버전들은 표 1과 같다.

Spark driver 노드는 Intel core i7 processor 3.30GHz를 장착한 CPU를 사용하였고, 8개의 코어를 가지고 있다. 그리고 Spark worker 노드 3대는 모두 동일한 사양으로 사용하였고, 4개의 코어를 가지고 있다. 운영체제(Operating system; OS)는 Spark driver 노드와 Spark worker 노드 모두 동일하게 CentOS 6.6 리눅스로 실험하였다.

표 1. 실험 환경.
Table 1. Simulation environment.

Category	Development Tools
PC Environment	CPU: Intel core i7 processor 3.30 GHz
RAM	Spark Driver node: 8 GB, Spark Worker node: 6.4 GB
OS	Linux, CentOS 6.6
Spark	1.4.0
Python	2.7
JVM	1.8.0

실내 측위를 위한 실내 환경은 그림 7과 같이 가로 40m, 세로 40m 넓이의 건물 내부를 대상으로 하였으며, 실내 환경의 구조별 성능을 분석하기 위해 크게 3곳으로 나누어 실험하였다. 사이트(site) 1과 사이트 2는 방 구조이고, 사이트 3은 복도 구조이다. 그림 8은 각 사이트에서 영상 특징점 추출에 대한 예를 보여준다. 사이트 3의 복도 구조의 경우 단조로운 벽과 창문들 때문에 특징점이 상대적으로 적게 뽑히는 특성을 가지고 있다. 추출되는 특징점의 개수에 따라 측위 오차 성능과 측위까지 소요되는 시간이 변화 될 수 있기 때문에 사이트를 구분하였다. 사이트 1에 대해서는 69,221 개의 3차원 좌표 점과 282,897 개의 SIFT 기술자를 데이터베이스로 구축하였고, 사이트 2에 대해서는 30,842 개의 3차원 좌표 점과 141,356 개의

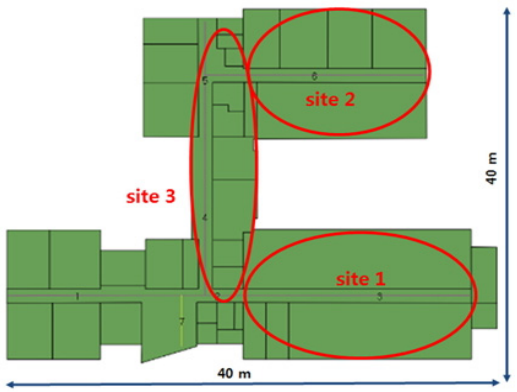


그림 7. 실내 측위 실험을 위한 건물의 실내 지도.
Fig. 7. Indoor map for indoor localization simulation.

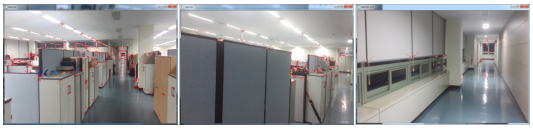


그림 8. 각 사이트에서 영상 특징점 추출에 대한 예제.
Fig. 8. Example of image feature extraction for indoor visual localization for each site.

SIFT 기술자를 데이터베이스로 구축하였다. 그리고 사이트 3에 대해서는 51,295 개의 3차원 좌표 점과 252,138 개의 SIFT 기술자를 데이터베이스로 구축하였다.

측위 데이터베이스 구축을 위해 촬영된 이미지의 해상도는 2048×2448이고, 실제 사용자 측위를 위한 카메라는 스마트폰에 내장된 카메라를 사용하여 4128×2322의 해상도를 가진다. 오프라인 단계에서 사용된 데이터베이스 구축용 이미지는 사이트 1에서 267개, 사이트 2에서 201개, 사이트 3에서 238개이다. 온라인 단계에서 사용된 측위용 이미지는 사이트 1에서 20개, 사이트 2에서 10개, 사이트 3에서 18개를 이용하여 사용자 측위를 수행하였다.

4.2 무손실 압축 기법에 따른 데이터베이스 저장 용량 효율 분석

측위 데이터베이스 구축에 사용된 사이트 1, 2, 3에서 추출된 SIFT 기술자들의 개수와 압축 전 용량은 표 2와 같다.

본 논문에서 제안하는 실내 위치인식 시스템이 보다 많은 건물들과 해당 건물들의 각 층에 대해서 측위 데이터베이스를 구축한다고 했을 때, 측위 데이터베이스에는 매우 큰 용량의 SIFT 기술자들이 저장되게 된다. 그러므로 측위 데이터베이스의 저장 용량 관리를 위해서는 데이터베이스에 저장되는 데이터들을 압축해서 저장하는 것이 효율적이다. 데이터 압축은 손실 압축과 무손실 압축으로 구분되는데, SIFT 기술자는 사용자의 실내 위치 정보를 계산하는데 사용되는 중요 정보로 압축 시 데이터 손실이 발생하면 안 된다. 따라서 본 논문에서는 무손실 압축 기법 중 사전 부호화 기반의 LZ-알고리즘들을 이용하여 SIFT 기술자들을 압축하여 저장하는 방식을 사용하였다. 본 실험에서 고려한 LZ-알고리즘은 LZ77, LZ78, LZSS, LZW으로, 총 4가지 무손실 압축 알고리즘을 통해 데이터베이스의 저장 용량 효율을 측정하였고, 그 결과는 표 2와 같다. 실험 결과 LZSS로 SIFT 기술자들을 압축하여 저장했을 때, 데이터베이스 저장 용량 효율이 가장 좋은 것으로 판명되었다.

표 2. 사이트에 따른 SIFT 기술자들의 용량.
Table 2. Volume of SIFT descriptors according to site.

	site 1	site 2	site 3
# of SIFT descriptors	282,897	141,356	252,138
압축 전 용량	101.9 MB	58.6 MB	85.6 MB
LZ77	56.5 MB	47.9 MB	33.3 MB
LZ78	55.5 MB	46.9 MB	32.3 MB
LZSS	47.4 MB	39.7 MB	25.3 MB
LZW	49.4 MB	41.8 MB	29.3 MB

4.3 영상 특징점 추출 성능 분석

4.3.1 영상 특징점 추출 연산 속도 성능 분석

영상의 특징점 추출 알고리즘 중에 본 논문에서 사용한 SIFT 알고리즘 이외에도 SURF(speed up robust features) 알고리즘[22]이 있다. SIFT와 SURF 두 알고리즘은 모두 영상의 크기, 회전, 시점 등의 변화에 불변하는 특징점과 그 기술자를 생성하는 공통적인 특성을 가진 알고리즘이다. SIFT는 128차원의 특징점 기술자를 갖는 반면, SURF는 128 보다 작은 차원의 특징점 기술자를 갖기 때문에 연산처리 속도 면에서도 빠르다는 장점을 갖고 있다. 하지만 특징점 기술자의 차원이 낮은 만큼 특징점의 강인함(robustness)이 떨어지는 단점을 갖고 있다. 본 논문에서는 측위 정확도를 위해 SIFT를 사용하였다.

그림 9는 사이트 1 데이터베이스를 기준으로, 실내 영상의 수에 따른 Spark-SURF, Spark-SIFT, SURF, SIFT의 총 연산 속도를 보여준다. 여기서 Spark-SURF, Spark-SIFT는 8개의 코어를 사용하여 Spark를 통해 SURF, SIFT를 각각 병렬 분산 처리한 경우이다. Spark-SIFT의 경우, SIFT와 비교하였을 때 평균 연산 속도가 약 3.584배 빨라지며, SURF와 비교하였을 때는 평균 처리 속도가 약 2.408배 빨라짐을 확인하였다. Spark-SURF의 경우, SURF와 비교하였

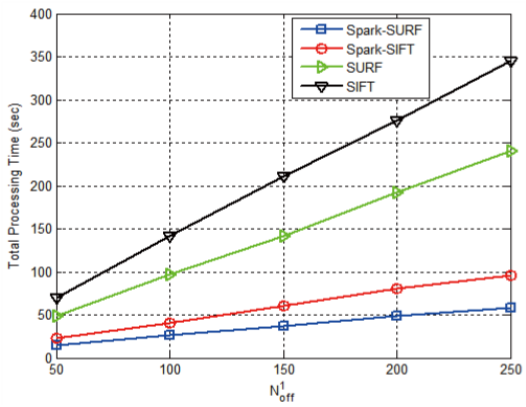


그림 9. 스파크-SURF, 스파크-SIFT, SURF, SIFT 의 연산 속도 비교.
Fig. 9. Comparison of processing time of Spark-SURF, Spark-SIFT, SURF, SIFT algorithms.

을 때 평균 연산 속도가 약 3.813배 빨라짐을 확인하였다.

4.3.2 영상 특징점 추출 정확도 분석

영상 특징점 추출의 정확도에 대한 성능 분석을 위해 본 논문에서는 보편적으로 많이 사용되고 있는 recall vs 1-precision을 이용하였다[23]. 이론적인 recall vs 1-precision 그래프는 precision 값이 증가 하더라도 일정한 recall 값을 갖지만, 실제 시스템에서는 다양한 요소에 의해 단조 증가하는 특성을 보인다. 그림 10은 1-precision에 따른 Spark-SURF, Spark-SIFT, SURF, SIFT의 recall을 보여준다. 그림 10에서 Spark-SURF와 SURF의 그래프가 서로 중첩되고, Spark-SIFR와 SIFR의 그래프 또한 서로 중첩되는 것을 확인할 수 있다. 이는 Spark 기반 병렬 분산 처리는 특징점 추출 과정에서만 이루어지는 것이기 때문에, 병렬 분산 처리 적용에 따른 실내 측위 정확도는 병렬 분산 처리를 적용하기 전과 변함이 없다는 것을 보여준다. 즉, 제안하는 병렬 분산 처리를 이용한 영상 기반 실내 위치인식 시스템은 Spark 기반 병렬 분산 처리를 통하여 실내 측위 정확도는 유지하면서 측위 정보 제공까지 걸리는 시간을 단축시킬 수 있다는 장점을 갖는다.

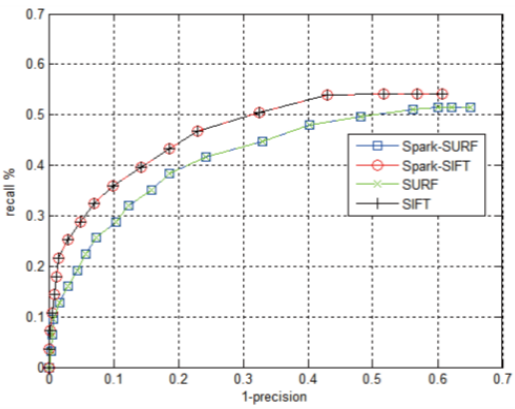


그림 10. 스파크-SURF, 스파크-SIFT, SURF, SIFT의 1-precision에 따른 recall 성능 비교
Fig. 10. Accuracy comparison using recall vs 1-precision for Spark-SURF, Spark-SIFT, SURF, SIFT.

4.4 코어 수에 따른 측위 속도 성능 분석

제안하는 Spark 기반 병렬 분산 처리를 적용하였을 때 코어 수에 따른 측위관련 성능을 비교하기 위해서 사이트 1, 2, 3에 대한 데이터베이스를 이용하였다. 그림 11-(a)은 오프라인 단계에서 코어 수에 따른 측위 데이터베이스 생성 시간을 나타낸 것이고, 그림 11-(b)는 온라인 단계에서 코어 수에 따른 사용자 측위 시간을 나타낸 것이다. 그림 11(a)에서 오프라인 단계에서 코어를 1개 사용하였을 때보다 코어를 8개 사용할 경우 약 4.106배 빠르게 측위 데이터베이스를 생성 할 수 있음을 보였다. 그림 11(b)에서는 온라인 단계에서 코어를 1개 사용하였을 때 보다 코어를 8개 사용할 경우 약 2.312배 빠르게 사용자의 측위 정보를 계산할 수 있음을 보였다.

그림 11(a), 11(b)를 통해서 병렬 분산 처리에 사용되는 코어 수가 증가할수록 SIFT 연산에 대한 처리 시간이 단축되는 것을 확인하였다. 본 논문에서는 Spark worker 노드 3대를 이용하였지만, 만약 이 보

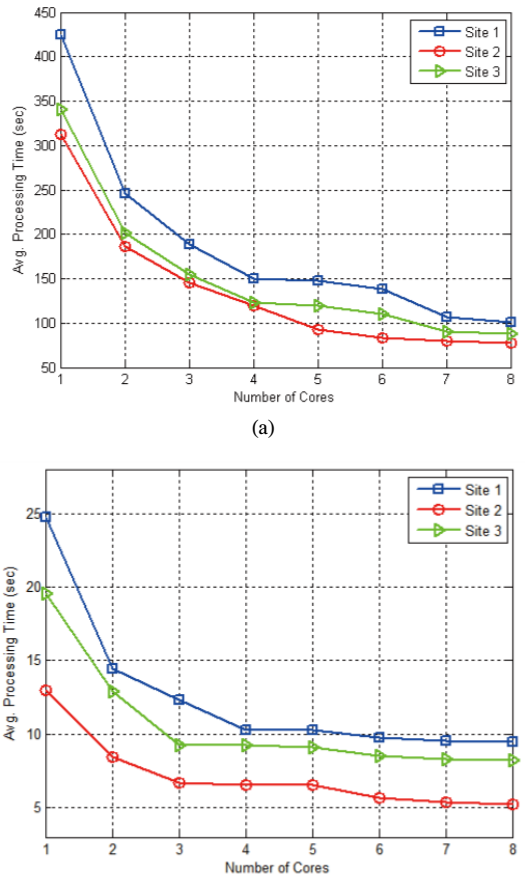


그림 11. 병렬화를 위해 사용한 코어의 개수에 따른 제안하는 시스템의 연산 처리 속도 비교; (a) 오프라인 단계의 측위 데이터베이스 생성 시간, (b) 온라인 단계의 사용자 위치인식 시간.
Fig. 11. Comparison of processing time of proposed system according to cores for parallelism; (a) localization database generation time of off-line phase, (b) user localization time of on-line phase.

다 더 많은 Spark worker 노드를 추가하고, Spark worker 노드에 내장된 코어들을 활용한다면 처리 시간이 더욱 줄어들 것이라는 예상을 할 수 있다.

V. 결 론

본 논문에서는 Spark 기반 병렬 분산 처리를 이용한 영상 기반 실내 위치인식 시스템을 제안하였다. 제안하는 시스템은 영상 특징점 추출 알고리즘을 Spark 기반의 병렬화를 통해, 데이터베이스 생성과 사용자 위치 추정까지 소요되는 시간을 단축시켰다. 또한 Spark 플랫폼에서 효율적으로 영상을 다룰 수 있게 하는 새로운 영상 처리 인터페이스를 제안하였다. 측위에 사용되는 영상 특징점 추출 알고리즘인 SIFT를 구현하여 제안하는 시스템에 적용함으로써 제안하는 Spark 영상 처리 인터페이스로의 다양한 영상처리 알고리즘 확장 가능성에 대해 보였다. 또한 사전 부호화 기반 LZ-무손실 압축 알고리즘을 이용하여 SIFT 기술자들을 압축하여 측위 데이터베이스에 저장함으로써, 데이터베이스 저장 용량의 효율성을 고려하였다. 그 결과 제안하는 영상 기반 실내 위치인식 시스템이 병렬 분산 처리 전과 동일한 측위 성능을 보이면서 병렬 분산 처리 기술을 적용하기 전 보다 단축된 시간으로 측위 정보를 사용자에게 제공할 수 있음을 실험 결과를 통해 입증하였다.

References

[1] S.-H. Jung, M.-W. Lee, K.-Y. Kim, and J.-S. Cha, “Position information acquisition method based on LED lights and smart device camera using 3-Axis moving distance measurement,” *J. KICS*, vol. 40, no. 1, pp. 226-232, Jan. 2015.

[2] M. K. Jung and D. M. Lee, “Performance analysis of the localization compensation algorithm for moving objects using the least-squares method,” *J. KICS*, vol. 39, no. 1, pp. 9-16, Jan. 2014.

[3] S. Son, T. Kim, Y. Jeon, and Y. Baek, “Smart camera technology to support high speed video processing in vehicular network,” *J. KICS*, vol. 40, no. 1, pp. 152-164, Jan. 2015.

[4] A. Irschara, et al., “From structure-from-motion point clouds to fast location recognition,” in *Proc. IEEE Conf. CVPR*, pp. 2599-2606, 2009.

[5] D. Robertson and R. Cipolla, “An image-based system for urban navigation,” in *Proc. British Machine Vision Conf.*, pp. 819-828, 2004.

[6] W. Zhang and J. Kosecka, “Image based localization in urban environments,” in *Proc. 3rd Int. Symp. 3D Data Process., Visualization, and Transmission*, pp. 33-40, 2006.

[7] B, Kwon, et al., “A target position decision algorithm based on analysis of path departure for an autonomous path keeping system,” *Wireless Pers. Commun.*, vol. 83, no. 3, pp. 1843-1865, 2015.

[8] M. Zaharia, et al., “Spark: cluster computing with working sets,” in *Proc. HotCloud*, pp. 10-16, 2010.

[9] J. Dean and S. Ghemawat, “Mapreduce: Simplified data processing on large clusters,” *Commun. ACM*, vol. 51, no. 1, pp. 107-113, 2004.

[10] M. Zaharia, et al., “Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing,” in *Proc. 9th USENIX Conf. Netw. Syst. Design and Implementation*, Apr. 2012.

[11] S. Yang and B. Wu, “Large scale video data analysis based on spark,” in *Proc. IEEE Int. Conf. Cloud Comput. and Big Data*, pp. 209-212, 2015.

[12] C. Sweeney, et al., *HIPI: A hadoop image processing interface for image-based mapreduce tasks*, University of Virginia, 2011.

[13] P. Moritz, et al., “SparkNet: training deep networks in Spark,” *arXiv preprint arXiv: 1511.06051*, 2015.

[14] Y. Jia, et al., “Caffe: convolutional architecture for fast feature embedding,” in *Proc. 22nd ACM Int. Conf. Multimedia*, pp. 675-678, Orlando, Florida, USA, Nov. 2014.

[15] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*, Cambridge University Press, 2003.

[16] D. Crandall, et al., “Discrete-continuous

optimization for large-scale structure from motion," in *Proc. IEEE CVPR*, pp. 3001-3008, 2011.

[17] J. M. Frahm, et al., "Building rome on a cloudless day," in *Proc. Eur. Conf. Computer Vision*, Springer Berlin Heidelberg, pp. 368-381, 2010.

[18] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Computer Vision*, vol. 60, no. 2, pp. 91-110, 2004.

[19] H. Liu, et al., "Robust and accurate mobile visual localization and its applications," *ACM Trans. Multimedia Comput. Commun., and Appl.*, vol. 9, no. 1s, p. 51, 2013.

[20] T. Sattler, et al., "Fast image-based localization using direct 2d-to-3d matching," in *Proc. IEEE ICCV*, pp. 667-674, Nov. 2011.

[21] Y. Li, et al., "Location recognition using prioritized feature matching," in *Proc. Eur. Conf. Computer Vision*, pp. 791-804, Springer Berlin Heidelberg, 2010.

[22] H. Bay, et al., "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346-359, 2008.

[23] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Trans. Pattern Anal. and Machine Intell.*, vol. 27, no. 10, pp. 1615-1630, 2005.

권 범 (Beom Kwon)

2012년 2월 : 숭실대학교 정보통신전자공학과 졸업
2012년 3월~현재 : 연세대학교 전기전자공학과 석박사통합과정
<관심 분야> 통신네트워크

전 동 현 (Donghyun Jeon)

2014년 2월 : 연세대학교 전기전자공학과 졸업
2014년 3월~현재 : 연세대학교 전기전자공학과 석사과정
<관심 분야> 실내 위치인식

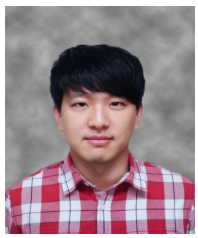
김 종 유 (Jongyoo Kim)

2011년 2월 : 연세대학교 전기전자공학과 졸업
2011년 3월~현재 : 연세대학교 전기전자공학과 석박사통합과정
<관심 분야> 화질평가

김 정 환 (Junghwan Kim)

2013년 2월 : 연세대학교 전기전자공학과 졸업
2013년 3월~현재 : 연세대학교 전기전자공학과 석박사통합과정
<관심 분야> 영상처리

김 도 영 (Doyoung Kim)



2014년 2월 : 연세대학교 전기전자공학과 졸업
2014년 3월~현재 : 연세대학교 전기전자공학과 석박사통합과정
<관심 분야> 행동인지

송 혜 원 (Hyewon Song)

2015년 2월 : 홍익대학교 전자전기공학과 졸업
2015년 9월~현재 : 연세대학교 전기전자공학과 석박사통합과정
<관심 분야> 영상 처리

이 상 훈 (Sanghoon Lee)

1989년 2월 : 연세대학교 전기전자공학과 학사
1991년 2월 : KAIST 전기전자공학과 석사
2000년 1월 : 텍사스 대학교 전기전자공학과 박사
2003년 2월~2007년 3월 : 연세대학교 전기전자공학과 조교수
2007년 4월~2012년 2월 : 연세대학교 전기전자공학과 부교수
2012년 3월~현재 : 연세대학교 전기전자공학과 정교수
<관심 분야> 이미지 프로세싱, 컴퓨터 비전, 화질 평가, 통신네트워크