Journal of Institute of Control, Robotics and Systems (20xx) 29(x):000-000
https://doi.org/10.5302/J.ICROS.20xx.xx.xxxx ISSN:1976-5622 eISSN:2233-4335

온디바이스 비주얼-언어 매핑과 내비게이션을 위한 쿼리 기반 물체 인지 지도 작성

Query-based Object-aware Mapping for On-device Visual Language Mapping and Navigation

윤 준 영, 김 필 은*
(Jun Young Yun[1], and Pileun Kim[2,*])
[1]Department of Artificial Intelligent, Korea Aerospace University
[2]Department of Autonomous Vehicle Engineering, Korea Aerospace University

Abstract: Recent advancements in artificial intelligence, particularly in multimodal models like LLM and VLM, robots can now perform zero-shot learning, allowing them to efficiently complete a variety of tasks. This paper introduces DMAP: a query-based object-aware mapping method utilizing VLMs to produce lightweight and effective navigation maps for edge robots. By dividing, encoding and saving keyframes with observed voxels, our method minimizes computational complexity, which allows real-time object aware mapping and language-driven navigation on low-powered devices like the Nvidia Jetson Orin Nano. This approach significantly reduces both map creation time and storage requirements, supporting zero-shot navigation without the need for prior object learning.

Keywords: AI-based Mapping, Object-aware Mapping, VLM, Robot Navigation, Zero-shot Object Navigation

I. 서론

최근 LLM(Large Language Models)과 같은 인공 지능의 발전으로 챗봇 등 디지털 환경에서 제로-샷(zero-shot) 성능을 가지게 되고 여러 방면의 작업을 추가적인 학습 없이 수행할 수 있다. 그에 따라 인공 지능을 이용하여 실제 물리 환경에 영향을 미칠 수 있는 지능형 로봇공학이 새로운 관심사로 떠오르고 있다. 지능적인 로봇이 여러 분야에 활용되기 위해선, 고성능 로봇뿐만 아니라 모바일 로봇 등의 작은 로봇들도 이러한 인공지능 모델을 이용하여 실제 물리 환경에서 처음 보는 임무에 대해서도 기존의 지식을 응용하는 제로-샷 수준의 성능을 가져야 한다.

기존 물체 인지 지도 작성 및 내비게이션 연구[7,9,11,12]에서는 복셀(voxel)이나 그리드(grid)에 VLM(Visual Language Model)의 임베딩(embedding)을 저장하는 방법 등으로 구현하였지만, 시간적 공간적 복잡도(complexity)가 크다는 단점이 있었다. 본 연구는 이런 지능적인 로봇이 임무를 수행하는 과정에서 언어를 통해 내비게이션할 수 있도록 VLM을 활용하여 지도를 작성하는 시스템을 구축하고, 해당 시스템이 모바일 로봇에서 수행할 수 있도록 요구 컴퓨팅 파워를 줄이는 것을 목표로 하였다.

본 연구에서 제안하는 모델은 프레임 분할(frame division)을 이용한 쿼리 기반의 물체 인지 지도 DMAP으로, SLAM(동시적 위치추정 및 지도작성)과 더불어 수행되며 지도를 작성하는 동안 얻어진 카메라 프레임 중 중요한 키프레임을 선별하고, VLM인 CLIP[1]의 이미지 인코더(image encoder)를 통해 키프레임을 임베딩한다. 임베딩에 복셀이나 그리드를 저장한 후 사용자 쿼리(query)를 통해 요청받은 문자에 해당하는 복셀를 특정하는 방법으로 지도 작성 시간과 저장되는 지도 용량을 획기적으로 단축시켰다. 이 과정에서 손실된 정확도는 키프레임 분할 방식을 이용하여 데이터를 증강(augmentation)하여 보완하였다. 이를 통해 모바일 로봇의 성능만으로도 실시간으로 물체의 정보를 포함하고 있는 지도를 작성할 수 있었다.

DMAP은 VLM으로 작성한 지도의 장점인 관측한 물체를 이해하는 지도를 작성하면서도, 이전 연구들에 비해 가볍기 때문에 다양한 환경에서 여러 작업을 수행하기 위해 사용할 수 있으며, 상대적으로 컴퓨팅 성능이 낮은 모바일 로봇으로도 작성이 가능하다. 또한 언어로 내비게이션이 가능하기 때문에 최근 지능형 로봇을 구동할 때 사용되는 LLM을 미세조정(fine-tuning)하여 내비게이션하기에도 유용하다.

본 논문은 다음과 같이 구성되어 있다. 제2장에서는 관련 연구와 본 연구진들의 선행 연구를 분석하여 해당 연구의 장단점들을 살펴보고, 본 연구에서 발전시키고자 하는 부분을 서술하였다. 제3장에서는 프레임 분할을 이용한 쿼리 기반 물체 인지 지도(DMAP)의 구조를 전체적으로 서술하고, 해당 지도가 가지는 3가지의 특징인 키프레임 선별, 쿼리 방식, 프레임 분할 방식에 대해 각각 자세히 설명한다. 제4장은 제안되는 지도 작성 방법을 카메라와 2D LiDAR가 장착된 Nvidia Jetson Orin Nano로 수행한 결과에 대해 토의한다. 마지막으로 제5장에서 결론을 서술하였다.

* Corresponding Author
Manuscript received July 10, 2024; revised August 16, 2024; accepted September 7, 2024.
윤준영: 한국항공대학교 인공지능학과 대학원생(a_o@kau.kr, ORCID 0009-0000-5140-3553)
김필은: 한국항공대학교 AI자율주행시스템공학과 교수(pkim@kau.ac.kr ORCID 0000-0001-8750-4524)
※ 본 연구는 2024년도 정부(과학기술정보통신부)의 재원으로 한국 연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2022R1F1A1073799)

II. 관련 연구

1. 동시적 위치추정 및 지도작성(SLAM)

SLAM(Simultaneous Localization And Mapping)은 로봇이 카메라나 LiDAR와 같은 센서를 통해 주위 환경에서 습득한 정보를 토대로 로봇의 현재 위치를 추정하면서 지도를 작성하는 기술이다[2][3]. ORB-SLAM[4]과 같은 VSLAM이나 LIO-SAM[5]같은 LiDAR SLAM의 경우 실시간으로 지도 작성이 가능하다는 장점이 있지만 작성된 지도에서 해당 특징점이나 해당 점 또는 복셀의 의미를 알기 어려워 지능적인 내비게이션을 위한 지도로 사용하기에는 한계가 있다. 이런 한계를 극복하기 위해 이미지 객체 인식이나 3D 객체 인식 같은 기술을 이용[6]하여 후처리는 가능하지만 사용하는 센서에 따라 다른 기술을 적용하여야 하고, 각각의 기술들도 학습이 필요하여, 본 연구에서 원하는 일반적인 지식 수준의 제로-샷 내비게이션 목표점 생성에는 한계가 있었다.

2. 비전 언어 모델을 이용한 내비게이션

SLAM의 이러한 단점을 보완하기 위해 의미적 분할 모델과 강화학습을 이용하여 지도를 작성하거나[7], 혹은 더욱 강력한 제로-샷 물체 성능을 위하여 VLM을 이용하여 내비게이션하거나[12] 지도를 작성[9]하려는 시도가 있었다. VLM은 CLIP[1], FLAVA[8]와 같이 인터넷 스케일로 사전 학습(pretrained)된 모델로, 이미지-언어 쌍을 학습하여 이미지와 문자를 특정 벡터 공간으로 인코딩하고, 인코딩된 이미지 임베딩과 문자 임베딩은 같은 벡터 공간에 존재하게 되어, 간단한 내적 계산만으로도 이미지와 문자의 유사도를 계산할 수 있다. VLM을 이용하여 지도를 작성하거나 내비게이션하는 모델로는 VLMaps[9], NLMap[10], LOC-ZSON[11], CoWs[12] 등의 모델이 있는데, 이는 강력한 일반화 능력과 물체를 인지하는(object-aware) 지도를 작성할 수 있고, 언어로 내비게이션할 수 있다는 장점이 있었지만, 학습이 따로 필요하거나[11] 컴퓨팅 파워가 많이 필요하여 시간이 오래 걸린다는 단점[9,10,11]이 있어 본 연구에서 목표하는 모바일 로봇에서 사용하기엔 부적절했다. 예를 들면 VLMaps[9]의 경우 의미적 분할(semantic segmentation) VLM인 LSeg[13] 모델을 사용하여 3D 복원(reconstruction) 후 2D로 투영시킨 그리드마다 임베딩을 저장하였는데, 30초 정도의 1000프레임짜리 RGBD 시퀀스를 처리하는 데에 Nvidia A100으로 15분 정도가 소요되었다. NLMap[10]의 경우 RPN(Region Proposal Network)을 통해 물체의 위치와 크기를 특정하고 두 개의 VLM을 이용하여 최대 앙상블(maximum ensemble) 방법으로 물체의 정보를 얻었다. 하지만 RPN과 같은 새로운 신경망을 추가하거나, 두 VLM을 동시에 사용하는 방법 모두 컴퓨팅 파워가 많이 필요하다. 또한 LOC-ZSON[11]의 경우 Object-Centric Image Encoder를 사용하여 얻은 이미지 임베딩을 사용하였지만 이 또한 이미지 내 물체의 위치를 얻기 위한 인코더를 추가적으로 학습하여야 한다는 단점이 있었다. 그래서 본 연구에서는 해당 연구들의 단점을 보완하여 강력한 일반화 능력을 유지하면서 적은 컴퓨팅 파워로 지도를 작성하는 것을 목표로 하였다.

3. 선행 연구

해당 단점을 보완하기 위해 본 연구진이 수행한 이전 연구인 CMAP[14]에서는 2D LiDAR와 카메라 정보를 통합하여 관측된 프레임을 VLM으로 인코딩한 이미지 임베딩 벡터(image embedding vector)와 SLAM으로 계산된 자세를 저장하는 방법으로 지도를 작성하였다. 이러한 방식으로 언어를 통한 제로-샷 내비게이션은 가능하였지만, 물체의 위치가 어디인지 알고 내비게이션하는 것이 아닌, 물체가 찍힌 위치로 내비게이션하는 방법에 가까웠다. 그렇기 때문에 프레임 안에서 어느 부분이 해당 물체인지는 알 수 없어 정확한 내비게이션이 어렵고, 해당 자세로 이동하기 때문에 물체가 찍히지 않은 다른 각도로은 이동할 수 없다는 단점이 있었다. 따라서 본 연구에서는 이런 단점을 보완하여 물체가 찍힌 위치가 아닌, 물체의 정확한 위치를 얻는 것을 목표로 하였다.

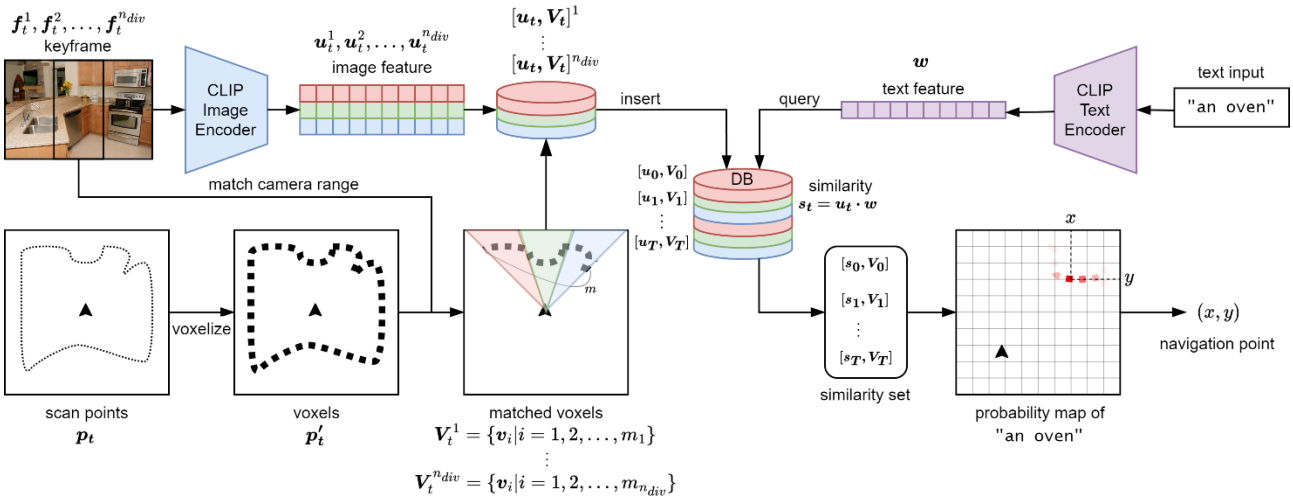


그림 1. DMAP 모델 구조: 쿼리 기반 물체 인지 지도

Fig. 1. DMAP model description: Query-based Object-aware Map

III. 프레임 분할을 이용한 쿼리 기반 물체 인지 지도

1. 모델 구조

모바일 로봇이나 서비스 로봇같은 저사양 컴퓨터가 탑재된 로봇에서 VLM을 사용하기 위해선 프로세스를 줄이고 연산시간 및 반응속도를 향상시킬 필요가 있었다. 또한 해당 물체의 정확한 위치를 파악하기 위한 과정도 필요했다. 이를 위해 본 연구에서는 이전 연구에서 사용한 키프레임 선별 알고리즘의 속도를 개선하고, 임베딩 표현 방식을 변경하고 쿼리 방식을 도입하여 연산시간과 반응속도를 향상시켰다. 또한, 프레임 분할 방식을 통해 물체의 정확한 위치를 특정하였다. 전체적인 모델의 구조는 그림 1과 같다.

일반적인 LiDAR SLAM 또는 VSLAM과 더불어 카메라를 통해 얻어진 프레임을 키프레임 선별 알고리즘을 이용하여 키프레임을 얻은 후, 키프레임 f_t에 대해 VLM을 이용하여 인코딩하여 이미지 임베딩 벡터 u_t를 얻는다. 프레임 선별 알고리즘은 이전 키프레임과 비슷한 정보를 가지고 있는 프레임은 계산하지 않고, 이전 키프레임보다 많은 정보를 가지고 있는 프레임을 키프레임으로 선별한다. 키프레임은 설정된 프레임 분할수 n_{div}으로 분할되어 $f_t^1, f_t^2, \dots, f_t^{n_{div}}$의 분할된 이미지에 대해 이미지 임베딩 벡터 $u_t^1, u_t^2, \dots, u_t^{n_{div}}$로 인코딩되고 각각의 이미지 임베딩 벡터는 해당 프레임 분할 분에서 관측된 복셀(혹은 그리드) $V_t^1, V_t^2, \dots, V_t^{n_{div}}$과 함께 쌍으로 데이터베이스에 저장된다. 사용자 문자 쿼리가 입력되면 VLM의 텍스트 인코더를 통해 문자 임베딩 벡터를 구한 뒤, 저장된 이미지 임베딩 벡터와의 내적 계산을 통해 유사도를 계산한다. 계산된 유사도와 함께 저장된 복셀을 통해 사용자 입력 문자에 대한 지도에서의 복셀 확률 지도를 얻고 가장 확률이 높은 복셀을 특정하여 내비게이션한다.

2. 키프레임 선별

VLM은 강력한 제로-샷 성능을 가지고 있지만 처리량이 큰 모델이기 때문에 컴퓨팅 성능이 낮은 에지 로봇에서는 모든 프레임을 실시간으로 계산할 수 없었다. 선행 연구[14]에서는 키프레임 선별 알고리즘을 도입하여 해결하였다. 키프레임 선별 알고리즘은 수식 (1)과 같이 현재 프레임의 정보(point cloud) 중 이전 키프레임에서 관측하지 않은 새로운 정보를 수치화하고 이전 키프레임보다 일정 수준(k_{th}) 이상의 정보를 가진 프레임을 키프레임으로 선별하는 방법이다.

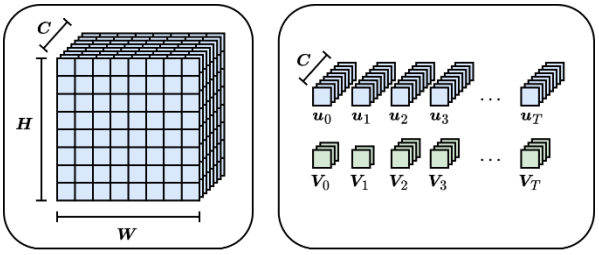
$$k_{t=kf}:\ f_t(\boldsymbol{P_t}) - f_{kf-1}(\boldsymbol{P_t} \cap \boldsymbol{P_{kf-1}}) > k_{th} \tag{1}$$

$$f_t(p) = w_o(d_{t,p}) = de^{\frac{d}{o}+1-\log o} \tag{2}$$

카메라에 촬영된 유의미한 정보량을 얻기 위해 한 점 p가 가지는 정보량을 나타내는 $f(p)$는 수식 (2)와 같이 나타내었다. $d_{t,p}$는 현재 위치 t와 점 p 사이의 거리이고, 정의된 o는 최적 촬영 거리를 나타낸다. $f_t(p)$의 최적 거리 o에서 최댓값인 1을 가지고, o보다 가까워지거나 멀어지면 0으로 근접하게 된다. 본 연구에서는 위 방식에서 속도를 개선하기 위하여 미리 계산된 w_o 테이블을 사용하였다. w_o 테이블을 사용하여 1000 픽셀 당 계산 시간을 4.148ms에서 0.566ms로 7.33배 단축시켰다.

3. 임베딩 표현 방식 변경 및 쿼리를 통한 확률 지도 생성

VLM을 이용해 지도를 작성하는 다른 알고리즘[9]은 LSeg[13]같은 의미적 분할 VLM을 이용하여 그리드마다 이미지 임베딩을 저장하는 방식을 사용한 후 미리 정해진 라벨 세트에 대해 지도를 작성하였다. 이는 정확도를 향상시킬 수는 있었지만 의미적 분할 VLM은 각 픽셀 별 임베딩을 계산하기 때문에 연산량이 많아 시간이 오래 걸릴 뿐만 아니라, 모든 그리드가 이미지 임베딩을 가져야 하므로 아니라 저장 공간 효율도 좋지 못했다. 본 연구에서는 쿼리 방식을 도입하고 이에 따라 그림 2 (b)와 같이 임베딩 표현 방식을 변경하여 이런 문제를 해결하였다.



(a) VLMaps (b) Ours

그림 2. 임베딩 표현 방식 비교

Fig. 2. Comparison on embedding representations.

임베딩을 모든 복셀(혹은 그리드)에 대해 저장하게 되면 공간의 크기($W \times H$)가 커질수록 지도를 표현하는 데이터의 크기가 기하급수적으로 증가하게 된다. 문자 임베딩을 통해 지도를 작성하는 과정에서도 모든 그리드에 대해 유사도를 계산하게 되어 그리드의 크기가 100 × 100인 작은 공간에서도 하나의 문자 임베딩마다 1만 회의 내적을 수행하여야 한다. 따라서 본 연구에서는 그림 2 (b)와 같이 키프레임에서 관측된 임베딩과 동시에 관측된 그리드를 저장하고, 그림 1의 오른쪽 부분과 같이 사용자 쿼리가 있을 때에만 쿼리로부터 받은 유사도와 해당 프레임에서 관측된 그리드를 통해 확률 지도를 작성하여 그리드를 특정하였다. 이렇게 되면 같은 양의 계산으로 1만 개의 키프레임의 유사도를 계산할 수 있다.

Algorithm 1 Generating probability map

Inputs: features $\boldsymbol{u_t}$, voxels $\boldsymbol{V_t}$, text_feature $\boldsymbol{w}$
Initialize: $\forall c_{x,y}, \forall d_{x,y}, \forall p_{x,y}, d_{max} \leftarrow 0$
$s_i \leftarrow \boldsymbol{u_i} \cdot \boldsymbol{w},\ i = 0, 1, \dots, T$
for i=0 to T **do**
  $S_i \leftarrow \frac{e^{s_i}}{\sum_{j=0}^{T} e^{s_j}}$
  for j=0 to m_i **do**
    $c_j \leftarrow \frac{c_j \times d_j + S_i}{d_j + 1}$
    $d_j \leftarrow d_j + 1$
    if $d_{max} < d_j$ **then** $d_{max} \leftarrow d_j$
    $p_j \leftarrow c_j + \frac{d_j}{d_{max}}$
  end for
end for
return $\boldsymbol{p}$

확률 지도 작성 방법은 알고리즘 1과 같이 각 키프레임에서의 유사도를 계산하고, 해당 키프레임에서 관측된 복셀에

대해 누적 평균 유사도와 관측된 빈도를 업데이트한다. 최종적으로 각 복셀이 가지고 있는 점수는 해당 복셀 v_j가 관측된 키프레임의 개수 d_j와 해당 키프레임들의 평균 유사도 c_j가 된다. 마지막으로 복셀 v_j의 확률 점수 p_j는 평균 유사도 c_j와 0과 1사이로 정규화된 d_j의 합이다.

쿼리 방식을 이용하여 의미적 분할 모델을 사용하지 않으면서 미리 정해진 라벨 세트 없이 객체 정보를 저장할 수 있었고, 키프레임 선별과 더불어 연산량을 크게 줄여 실시간으로 수행할 수 있었다. 또한, 복셀마다 임베딩을 저장하지 않았기 때문에 공간적 복잡도도 크게 줄일 수 있었다.

4. 프레임 분할 방식

쿼리 기반으로 복셀을 특정하는 방법은 모든 복셀에 임베딩을 저장하지 않고, 키프레임에 해당하는 임베딩과 그 때에 관측된 복셀을 저장하기 때문에, 입력 쿼리 문자에 해당하는 복셀을 삼각 측량과 유사한 방법으로 정확하게 특정하기 위해선 여러 화각에서 찍힌 물체가 필요했다. 더 많은 화각에서 찍힌 물체를 얻기 위해 DMAP은 그림 3처럼 한 프레임을 n개로 분할하여 처리함으로써 데이터를 n_{div}배 증강시켜 처리하는 방식을 통해 해결했다. n_{div}배로 나누어진 키프레임은 배치를 이용하여 이미지 임베딩 벡터로 인코딩 되기 때문에 한 번에 처리가 가능하며, 연산 속도는 소폭 증가하게 되었지만 확률의 편차를 크게 줄여 정밀도를 크게 향상시킬 수 있었다. 하지만 n_{div} 값을 너무 크게 하면 임베딩되는 분할 분의 화각이 작아져 큰 물체를 식별하지 못할 가능성도 있다.

위 세가지 방법을 통해 시간, 공간적인 복잡도를 크게 줄이면서도 추가적인 학습 없이 제로-샷 물체 내비게이션이 가능한 지도를 작성할 수 있었다. 방법론적으로는 복셀, 그리드, 특징점을 저장하는 방식이기 때문에, 카메라만 사용하거나 2D LiDAR, 3D LiDAR를 카메라와 함께 사용하는 등의 센서의 제약이 적은 편이며, 지도와 함께 추가적인 데이터베이스를 저장하는 방식이므로 위치 추정 및 지도 작성의 기반으로 사용되는 SLAM 프레임워크에 대해서도 제약이 없다. 또한 쿼리 과정에서의 연산은 간단한 내적과 정렬 연산 등으로 이루어져 있기 때문에 제안된 지도 작성 모델로 작성된 지도와 작성하는데 사용한 VLM의 텍스트 인코더로 미리 인코딩된 가능한 문자 세트(possible text set)만 있다면 Raspberry Pi같은 초소형 컴퓨터가 장착된 로봇으로도 사용이 가능하다.

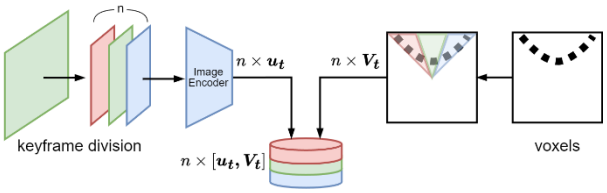


그림 3. 키프레임 분할 방식을 통한 데이터 증강

Fig. 3. Data augmentation with keyframe division method

IV. 실험

먼저 실험 1은 제안된 모델을 검증하기 위해 여러 VLM 모델과 각각 다른 프레임 분할수 n_{div}로 지도를 작성하였고, 작성하는 동안 사용된 키프레임 개수(Total keyframes), 분할된 키프레임 개수(Total divisions), 평균 FPS(Avg FPS; frame per second), 평균 DPS(AVG DPS; division per second), 작성된 지도의 용량(Map size), 단어 쿼리에 대한 확률 지도 생성 시간(Inference time)을 측정하였다. 실험 2는 제안된 모델과 다른 연구에서의 모델의 성능을 비교하기 위해 VLMaps[9]와 서로 다른 GPU를 탑재한 시스템으로 실험하였다. 두 모델의 연산 속도(Process time), 추론 속도(Inference time), 지도의 용량(Map size)를 측정하였다.

표 1. VLM 모델과 n_{div} 분할에 따른 DMAP 모델의 실험 결과.

Table 1. Description of VLM models and experiment results of various division of DMAP.

Model (pretrained)	Params (M)	FLOPs (B)	n	Total keyframes	Total divisions	Avg FPS	Avg DPS	Map size (MB)	Inference time (s)
ViT-B/32 (datacomp_xl_s13b_b90k)	151.28	14.78	1	1384	1384	5.51	5.51	6.8	0.129
			2	1240(89.6%)	2480	4.67	9.34	11.8	0.166
			3	1211(87.5%)	3633	4.69	14.07	17.1	0.201
			4	1176(85.0%)	4704	4.33	17.32	22.1	0.256
			5	1094(79.0%)	5470	4.05	20.25	25.6	0.231
ViT-B/32-256 (datacomp_s34b_b86k)	151.29	17.46	1	1346	1346	5.21	5.21	6.6	0.123
			2	1308(97.2%)	2616	4.87	9.74	12.4	0.170
			3	1262(93.8%)	3786	4.54	13.62	17.7	0.206
			4	1240(92.1%)	4960	4.60	18.40	23.2	0.248
			5	1041(77.3%)	5205	3.84	19.20	24.3	0.276
ViT-B/16-SigLIP (webli)	203.16	46.44	1	1352	1352	5.17	5.17	9.7	0.149
			2	1258(93.0%)	2516	4.67	9.34	17.8	0.205
			3	1006(74.4%)	3018	4.09	12.27	21.2	0.215
			4	1039(76.8%)	4156	3.82	15.28	29.1	0.251
			5	980(72.5%)	4900	3.60	18.00	34.2	0.278
ViT-L/14-quickgelu (dfn2b)	427.62	175.33	1	1011	1011	3.69	3.69	7.3	0.127
			2	806(79.7%)	1612	2.92	5.84	11.4	0.175
			3	698(69.0%)	2094	2.52	7.56	14.7	0.196
			4	606(59.9%)	2424	2.20	8.80	17.0	0.201
			5	533(52.7%)	2665	1.93	9.65	18.6	0.225

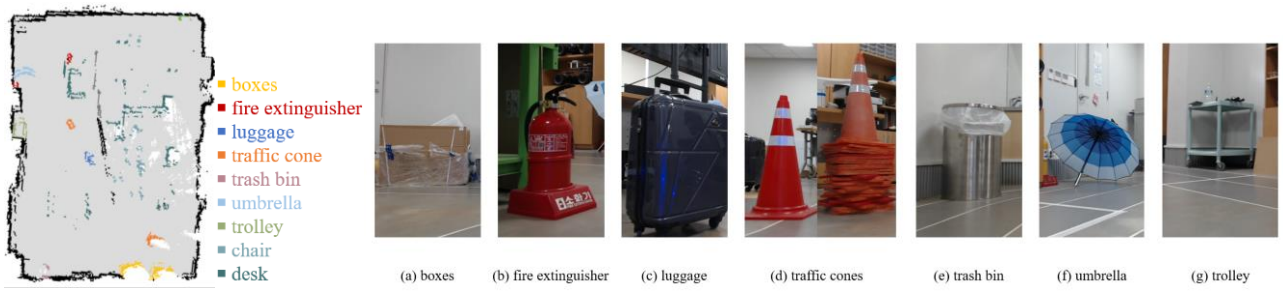


그림 4. 실험에 사용된 물체 사진과 실측 지도

Fig. 4. Ground truth map and picture of objects of experiment environment

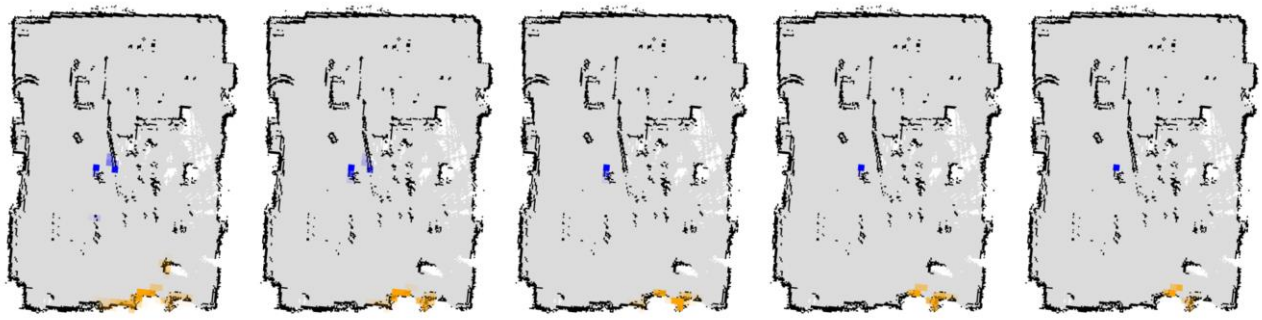


그림 5. ‘boxes’, ‘luggage’ 입력 쿼리에 대한 ViT-B/16-SigLIP 모델의 확률 지도 ($n_{div} = 1,2,3,4,5$)

Fig. 5. Probability map of ViT-B/16-SigLIP model with input query ‘boxes’, ‘luggage’ ($n_{div} = 1,2,3,4,5$)

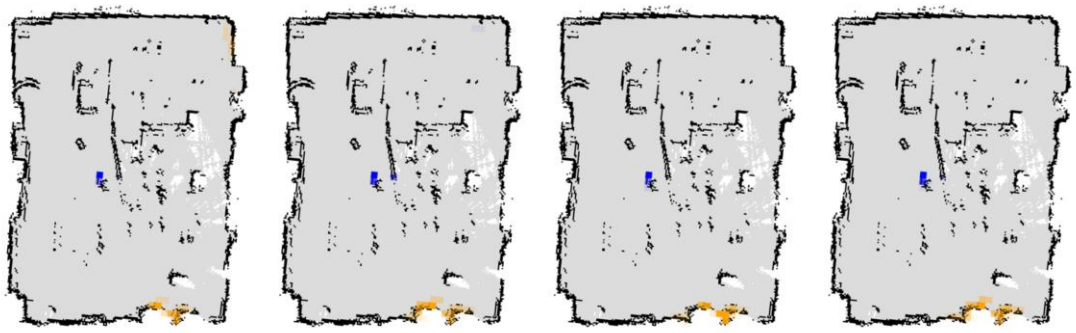


그림 6. ‘boxes’, ‘luggage’ 입력 쿼리에 대한 네 모델의 확률 지도 ($n_{div} = 3$)

Fig. 6. Probability map of 4 VLMs with input query ‘boxes’, ‘luggage’ ($n_{div} = 3$)

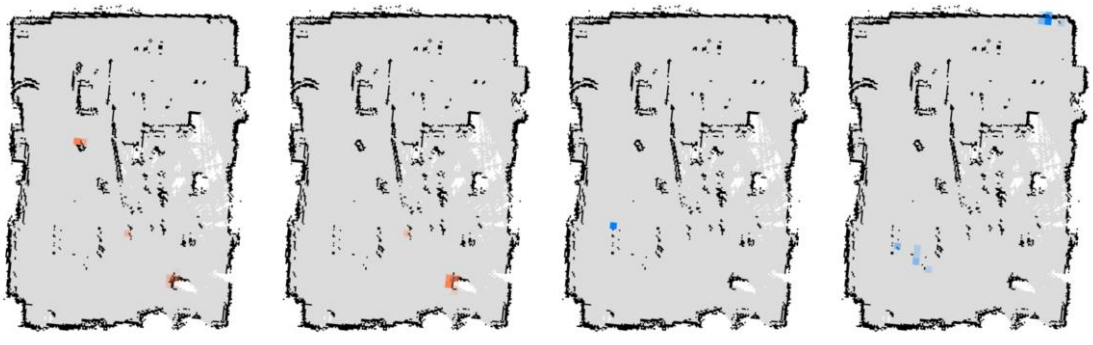


그림 7. ‘a single traffic cone’, ‘pile of traffic cones’, ‘chair’, ‘folded chair’ 입력 쿼리에 대한 ViT-B/16-SigLIP 모델의 확률 지도 ($n_{div} = 3$)

Fig. 7. Probability map of ViT-B/16-SigLIP model with input queries ‘a single traffic cone’, ‘pile of traffic cones’, ‘chair’, ‘folded chair’ ($n_{div} = 3$)

실험 1. 각 모델과 프레임 분할수에 따른 성능 분석

실험은 Nvidia Jetson Orin Nano로 진행하였고, 사용된 VLM은 OpenCLIP[15] 라이브러리를 이용하여 여러 데이터셋으로 사전 학습된 ViT-B/32, ViT-B/32-256, ViT-B/16-SigLIP, ViT-L-14-quickgelu의 네 가지 모델을 사용하였다. 사용된 프레임 분할수는 $n_{div} = 1,2,3,4,5$이다. 지도를 작성하는 데 사용된 센서는 Logitech C920e 카메라와 SLAMTEC RPLiDAR A2M8 2D LiDAR를 사용하였다. SLAM 프레임워크는 SLAM Toolbox[16]를 사용하였다. 문자 쿼리는 'a single traffic cone', 'pile of traffic cones', 'boxes', 'chair', 'fire extinguisher', 'folded chair', 'luggage', 'trash bin', 'trolley', 'umbrella'로 실험 환경에 배치되어 위치를 특정할 수 있는 다중('chair', 'boxes') 혹은 단일 물체(나머지)에 대한 문자로 입력되었다. 지도 작성 시 이미지 임베딩 벡터에 저장되는 그리드의 크기는 0.25m로 하였다. 표 1은 네 모델의 사전 학습된 데이터셋(pretrained), 파라미터 수(Params), 연산 수(FLOPs; float operations)와 실험값을 표시한 결과이다. 실험 환경에서 작성된 지도에서 그리드 당 분류된 실제 물체 지도(ground truth map)는 그림 4와 같다.

그림 5는 ViT-B/16-SigLIP 모델을 이용하여 $n_{div} = 1,2,3,4,5$의 프레임 분할수로 작성된 지도에 'boxes', 'luggage'의 입력을 통해 생성된 확률 지도이고, 그림 6은 네 가지 모델에 대해 $n_{div} = 3$ 의 프레임 분할수로 작성된 지도에 'boxes', 'luggage'의 입력을 통해 생성된 확률 지도이다. 그림 7은 ViT-B/16-SigLIP, $n_{div} = 3$의 지도에 대해 'a single traffic cone', 'pile of traffic cones', 'chair', 'folded chair'의 입력 쿼리를 통해 생성된 확률 지도이다.

그림 5는 실험 환경에 배치되어 있는 물체 중 비교적 부피가 큰 'boxes'와 부피가 작은 'luggage'를 $n_{div} = 1,2,3,4,5$에 대해 확률 지도를 나타낸 것으로, 이를 통해 프레임 분할수에 따른 확률 지도의 정밀도를 확인할 수 있다. $n_{div} = 1$의 경우 키프레임을 그대로 인코딩한 것으로 프레임 분할을 하지 않고 선행 연구[14]에서 쿼리 방식만을 추가로 적용한 것과 같다. n_{div} 값이 증가할수록 확률 지도에서의 편차가 좁아 더욱 정밀하고 정확하게 물체를 특정할 수 있다. 그림 5 (a)를 보면 프레임 분할이 이루어지지 않았을 때에는 촬영된 주위 그리드들에도 확률이 존재하여 정확한 내비게이션이 어렵지만 $n_{div} = 3$ 이상에서는 정확하게 특정할 수 있었다. 큰 물체인 'boxes'의 경우 $n_{div} = 3,4,5$에서 모두 유사한 결과를 나타내었으며, 작은 물체인 'luggage'의 경우 $n_{div} = 3$보다 $n_{div} = 4,5$에서 확률이 더 정밀하게 나타났다. 제안하는 모델은 의미적 분할 모델을 사용하지 않고 주어진 분류 목록(label list)이 없는 제로-샷 모델이기 때문에 적절한 프레임 분할수가 필요하다.

그림 6은 네 가지 모델에 대해 $n_{div} = 3$으로 작성한 지도를 'boxes'와 'luggage'의 쿼리에 대한 확률 지도를 나타낸 것이다. 'boxes' 쿼리에서는 ViT-B/16-SigLIP 모델이 가장 정밀하게 나타났으며, 'luggage' 쿼리에서는 ViT-B/16-SigLIP 모델과 ViT-L/14-quickgelu 모델이 정밀하게 나타났다. 가장 파라미터가 많고 연산량이 많아 정확도가 높은 모델인 ViT-L/14-quickgelu 모델을 사용하여도, 처리 속도가 느려 많은 프레임을 연산하지 못하여 정확도가 낮아진 것으로 보인다. 표 1과 함께 보면 세 가지 ViT-B 모델은 실험 시퀀스에서 $n_{div} = 3$일 때 각각 3633, 3786, 3018개의 분할된 프레임을 처리하였지만, ViT-L 모델은 2094개의 분할된 프레임을 처리하여 많은 프레임을 연산하지 못하였고 이러한 점 때문에 정확도가 감소된 것으로 보인다.

그림 7은 같은 시퀀스에서 비슷한 물체 혹은 다중 물체가 관측되었을 경우에 대한 확률 지도이다. 그림 7의 (c)를 보면 'chair'라는 입력 쿼리에 의해 생성된 확률 지도는 여러 의자 중 하나의 확률이 높아, 의도하지 않을 수 있는 물체를 나타낼 수 있는 것을 확인할 수 있었지만, (d)를 보면 'folded chair'라는 입력처럼 더 정확한 설명을 통해 의도하는 물체의 위치를 특정할 수 있다. 또한, 그림 7 (a)와 (b)를 보면 실험 환경에 놓여 있는 하나의 라바콘과 라바콘 더미를 구분할 수 있었다. 따라서 VLM의 장점인 언어를 통해 확률을 계산할 수 있다는 특성을 활용하여 프롬프팅(prompting)을 통해 원하는 특정 물체의 위치를 알 수 있다.

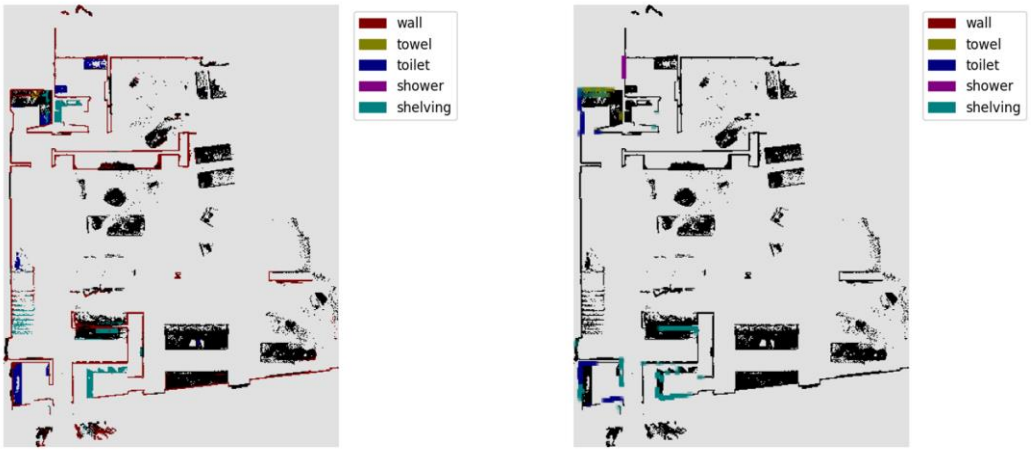
실험 2. 다른 연구와의 성능 비교

실험은 본 연구와 유사한 VLM을 사용한 지도 작성 모델인 VLMaps[9]를 동일한 MatterPort3D[17] RGBD 데이터셋을 사용하여 지도(임베딩 표현)를 작성하는 시간(Process time)을 측정하고, 지도에서 실제 확률 지도(VLMaps의 경우 Landmark indexing)를 작성하는 추론 시간(Inference time)을 측정하였다. 또한 작성된 지도의 크기를 측정하였다. 위 과정을 Jetson Orin Nano와 Nvidia Tesla T4를 장착한 시스템에서 각각 수행하였다. 사용한 VLM (혹은 의미적 분할 모델)은 같은 ViT-L/16 기반 모델을 사용하였고, 1159개의 프레임 시퀀스에 대해 키프레임 선별을 거치지 않고 $n_{div} = 3$의 프레임 분할수로 모든 프레임에 대해 계산하였다. 실험 결과는 표 2를 통해 나타내었다.

표 2. VLMaps[9]와의 연산, 추론 속도 및 맵 크기 비교.
Table 2. Comparison of process, inference speed and map size with VLMaps[9].

Model	**VLMaps[9] + LSeg[13]** (ViT-L/16)		**Ours + OpenCLIP[15]** (ViT-L/16)	
Device	Jetson	T4	Jetson	T4
Process	N/A	1278s	702s	410s
Inference	N/A	0.217s	0.378s	0.235s
Map Size	338.8MB		38.2MB	

실험 결과를 보면 VLMaps의 경우 Jetson Orin Nano에서 컴퓨팅 파워가 부족하여 구동되지 못하였고, 컴퓨팅 파워가 충분한 환경에서도 DMAP보다 월등히 느리게 구동되었다. T4 시스템 에서 DMAP은 같은 ViT-L/16 기반 VLM을 사용한 VLMaps에 비해 연산 속도는 약 3.1배 증가하였고, 저장 공간은 약 8.9배 적게 사용하는 것을 확인할 수 있었다. 그림 8은 두 모델을 통해 작성한 지도를 검증하기 위해 'wall', 'towel', 'toilet', 'shower', 'shelving'의 5개의 물체에 대해 물체 지도를 생성한 결과이다.



(a) VLMaps (b) DMAP

그림 8. ‘wall’, ‘towel’, ‘toilet’, ‘shower’, ‘shelving’ 물체 입력에 대한 두 지도 작성 모델의 지도

Fig. 8. Indexed map of two object-aware map with object input ‘wall’, ‘towel’, ‘toilet’, ‘shower’, ‘shelving’

VLMaps의 경우 입력된 문자와의 유사도를 비교하여 가장 적합한 물체(max argument)로 지도를 작성하는 방법을 사용하여 물체 지도 작성(Landmark Indexing) 과정에서 가능한 모든 물체를 입력하여야 정확한 지도 작성이 가능하여 ‘other’라는 물체를 추가하여 실험하였다. 반면, DMAP의 경우 확률 지도를 생성하기 때문에 환경에 존재하는 모든 물체를 작성하지 않아도 특정 물체로 이동이 가능하다는 장점이 있었다. 그림 8을 보면, ‘towel’의 경우 VLMaps는 위치를 알아내지 못하였지만, DMAP은 위치를 특정할 수 있었다. 하지만, 실험 1의 결과처럼 ‘wall’과 같이 항상 관측되는 물체에 대해서는 제대로 물체 확률 지도를 생성하지 못하는 것을 확인할 수 있었다.

V. 결론

본 논문은 기존의 VLM을 활용한 지도 작성 방식의 단점을 쿼리 방식을 도입하여 지도 작성 시간과 부하를 대폭 개선시켰다. 제안된 모델은 물체 인지 지도 작성 방식을 경량화시켜 모바일 로봇에서도 지능적으로 물체를 인식하여 내비게이션을 수행할 수 있게 하였다.

기존의 VLM을 활용한 지도 작성 방법[9]은 모든 복셀에 대해 임베딩 벡터를 계산하고 저장하는 과정에서 시간적, 공간적 복잡도가 크다는 단점이 있었다. 이러한 단점을 보완하기 위해, 쿼리 기반의 물체 인지 지도 모델인 DMAP을 제안하였다. DMAP은 모든 복셀에 대해 임베딩을 저장하지 않고 입력 쿼리에 대한 복셀을 특정하는 방법을 사용하고, 키프레임 분할을 통한 데이터 증강으로 정밀도를 보완하였다. 이를 통해 프로세스를 경량화하여 고가의 인공 지능용 그래픽카드를 사용하지 않고도 실시간으로 지도 작성을 수행할 수 있게 하였고, 저장되는 지도 용량을 대폭 감소시켰다. 제안하는 모델은 모바일 로봇에 주로 사용하는 Jetson Orin Nano의 성능만으로도 실시간으로 물체의 정보를 포함한 지도를 작성할 수 있었다.

이 모델은 관측한 물체를 따로 학습하지 않고도 이해하는 제로-샷 내비게이션이 가능한 지도를 작성하면서도 가벼운 모델이기 때문에 모바일 컴퓨터로도 온디바이스(on-device)로 지도를 작성할 수 있었으며, 언어로 내비게이션이 가능하므로 최근 지능형 로봇을 구동할 때 사용하는 LLM을 이용한 내비게이션에 응용할 수 있다는 장점이 있다. 또한 제안하는 모델에서 작성한 지도와 인코딩된 가능한 문자 목록을 제공하면 큰 네트워크를 사용하지 않고 간단한 내적 계산으로 확률 지도를 얻을 수 있기 때문에 Raspberry Pi 같은 초소형 컴퓨터로도 사용이 가능하다.

저장된 데이터베이스와 입력 문자로 확률 지도를 생성하기 때문에 로봇이 지도가 작성된 공간 자체를 이해하지는 못하여, 지도 작성 환경에 놓여있는 중복된 물체가 관측되었다면 사용자가 의도한 물체로 내비게이션하기 어렵다는 단점이 있었다. 하지만 VLM의 장점인 언어를 이해한다는 특성을 이용하여 구체적인 설명(e.g. ‘접이식 의자’, ‘라바콘 더미’ 등)을 통한 프롬프팅으로 의도한 물체로 내비게이션할 수 있음을 실험을 통해 확인하였다.

추후 연구에서는 이러한 단점을 보완하여, 미세 조정된 LLM으로 지도 자체를 이해하며 사용자 요청에 맞는 물체로 내비게이션 할 수 있는 시스템을 구축할 예정이다. 더 나아가 명령에 따라 물체를 집거나 조작할 수 있는 매니퓰레이터(manipulator)를 장착하면, LLM의 사고 연쇄(chain-of-thought)를 통해 복잡한 임무를 세분화하여 순서대로 내비게이션하고 조작할 수 있는 더욱 발전된 지능형 로봇을 기대할 수 있다.

REFERENCES

[1] A. Radford, J.-W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, ..., and I. Sutskever, “Learning transferable visual models from natural language supervision,” *Proceedings of the 38th International Conference on Machine Learning*, vol. 139, pp. 8748-8763, July, 2022.
doi: https://doi.org/10.48550/arXiv.2103.00020

[2] X. Yue, Y. Zhang, J. Chen, X. Zhou, and M. He, “LiDAR-based SLAM for robotic mapping: state of the art and new frontiers.” *Industrial Robot*, vol. 51, no. 2, pp. 196-205, 2024.
doi: https://doi.org/10.1108/IR-09-2023-0225

[3] I. A. Kazerouni, L. Fitzgerald, G. Dooly and D. Toal, “A survey of state-of-the-art on visual SLAM,” *Expert Systems with Applications*, vol. 205, 117734. 2022.
doi: https://doi.org/10.1016/j.eswa.2022.117734

[4] R. Mur-Artal, J. M. M. Montiel and J. D. Tardos, “ORB-SLAM: a versatile and accurate monocular SLAM system,” *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147-1163, 2015.
doi: https://doi.org/10.1109/TRO.2015.2463671

[5] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti and D. Rus, “LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping,” *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp.5135-5142, October, 2020
doi: https://doi.org/10.1109/IROS45743.2020.9341176

[6] S. Yang and S. Scherer, "CubeSLAM: Monocular 3-D Object SLAM," *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 925-938, 2019
doi: https://doi.org/10.1109/TRO.2019.2909168

[7] J. Choi and I. Kim, "Goal Object Grounding and Multimodal Mapping for Multi-object Visual Navigation," *Journal of Institute of Control, Robotics and Systems (in Korean)*, vol. 30, no. 6, pp. 596-606, 2024
doi: https://doi.org/10.5302/J.ICROS.2024.23.0217

[8] A. Singh, R. Hu, V. Goswami, G. Couairon, W. Galuba, M. Rohrbach and D. Kiela, "FLAVA: A Foundational Language and Vision Alignment Model," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15638-15650, June, 2022
doi: https://doi.org/10.48550/arXiv.2112.04482

[9] C. Huang, O. Mees, A. Zeng and W. Burgard, "Visual Language Maps for Robot Navigation," *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10608-10615, May, 2023.
doi: https://doi.org/10.1109/ICRA48891.2023.10160969

[10] B. Chen, F. Xia, B. Ichter, K. Rao, K. Gopalakrishnan, M. S. Ryoo and D. Kappler, "Open-vocabulary Queryable Scene Representations for Real World Planning," *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11509-11522, May, 2023.
doi: https://doi.org/10.1109/ICRA48891.2023.10161534

[11] T. Guan, Y. Yang, H. Cheng, M. Lin, R. Kim, R. Madhivanan, ..., and D. Manocha, "LOC-ZSON: Language-driven Object-Centric Zero-Shot Object Retrieval and Navigation," *arXiv preprint arXiv:2405.05363*, 2024.
doi: https://doi.org/10.48550/arXiv.2405.05363

[12] S. Y. Gadre, M. Wortsman, G. Ilharco, L. Schmidt and S. Song, "CoWs on Pasture: Baselines and Benchmarks for Language-Driven Zero-Shot Object Navigation," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23171-23181, June, 2023.
doi: https://doi.org/10.48550/arXiv.2203.10421

[13] B. Li, K. Q. Weinberger, S. Belongie, V. Koltun and R. Ranftl, "Language-driven Semantic Segmentation," *arXiv preprint arXiv:2201.03546*, 2022.
doi: https://doi.org/10.48550/arXiv.2201.03546

[14] J. Y. Yun and P. Kim, “Building Real-time Image Embedded Map for Edge Robot Navigation,” *Proc. of ICROS Annual Conference 2024*, Daejeon, Korea, pp. 821-822, July, 2024.

[15] M. Cherti, R. Beaumont, R. Wightman, M. Wortsman, G. Ilharco, C. Gordon, ..., and J. Jitsev, "Reproducible Scaling Laws for Contrastive Language-Image Learning", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818-2829, June, 2023.
doi: https://doi.org/10.48550/arXiv.2212.07143

[16] S. Macenski and I. Jambrecic, "SLAM Toolbox: SLAM for the dynamic world," *Journal of Open Source Software*, vol. 6, no. 61, 2783, 2021.
doi: https://doi.org/10.21105/joss.02783

[17] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, …, and Y. Zhang, "Matterport3D: Learning from RGB-D Data in Indoor Environments," *arXiv preprint arXiv:1709.06158*, 2017.
doi: https://doi.org/10.48550/arXiv.1709.06158

윤 준 영

2023년 한국항공대학교 항공우주 및 기계공학부 졸업. 2023년~현재 한국항공대학교 대학원 인공지능학과 석사과정 재학 중. 관심분야는 인공지능 기반 인식 및 내비게이션, 자율주행.

김 필 은

2008년 인하대학교 기계공학부 졸업. 2017년 Georgia Institute of Technology, Mechanical Engineering 석사. 2020년 동 대학원 Civil Engineering 공학박사. 2021년~현재 한국항공대학교 AI자율주행시스템공학과 교수. 관심분야는 자율시스템, 모바일 로봇, 인공지능 기반 인식 및 제어.