JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 21, No. 4, pp. 811-816, Apr. 2020

Check for updates

인공지능을 적용한 소형 모바일 로봇을 위한 물체 및 환경 인식 연구

김 현 태[1] · 김 영 식[2*]
[1]한밭대학교 전자제어공학과 학부과정
[2]한밭대학교 기계공학과 교수

Object and environment recognition for a small mobile robot applying artificial intelligence

Hyuntae Kim[1] · Youngshik Kim[2*]
[1]Bachelor's Course, Department of Electronic Control Engineering, Hanbat National University, Daejeon, Korea
[2]Professor, Department of Mechanical Engineering, Hanbat National University, Daejeon, Korea

[요 약]

본 연구에서는 제한된 H/W 성능을 가진 소형 모바일 로봇의 성공적인 자율주행을 위하여 인공지능을 적용하여 주변의 환경을 효과적으로 인식하기 위한 연구를 수행하였다. 이를 위하여 먼저 비정형 환경에서 구동 바퀴의 형태를 자유롭게 변형하여 험지를 효과적으로 극복할 수 있는 소형 휠-레그(wheel-leg) 모바일 로봇을 설계하였다. 험지를 효율적으로 주행하기 위해 로봇의 바퀴에 기어구조를 적용하여 휠-레그의 크기가 최대 2.5배까지 변형할 수 있도록 설계하였다. 주변 환경의 인식을 위해 GPU가 내장된 저가형 임베디드 보드와 카메라를 사용하였다. 본 연구에서 사용한 딥러닝 모델은 YOLOv3를 사용하였고 학습한 모델의 성능을 mAP, GIoU 등으로 확인하였다. 본 연구를 통하여 제안된 객체 인식 알고리즘을 실제 로봇에 적용하여 로봇이 비정형 환경에서 효과적으로 장애물을 인식하고 이를 극복하는 것을 확인하였다.

[Abstract]

In this research we study artificial intelligence-based environment recognition for autonomous navigation of a small mobile robot with limited hardware resources. Toward this goal, we designed a wheel-leg mobile robot that could transform its circular wheel configuration into wheel-leg configuration for rugged terrain locomotion. Size of a wheel-leg can be increased up to 2.5 times larger when compared to circular wheel configuration applying gear-driven mechanisms. A web camera and less expensive embedded board integrated with a GPU are used in the case. We used YOLOv3 as a deep learning model to recognize obstacles in real time. Performance of learned model is verified using mAP and GIoU. Our proposed object recognition algorithm is implemented in the real wheel-leg mobile robot and its performance is verified in unstructured environments.

색인어 : 딥러닝, 욜로, 휠-레그 모바일 로봇, 험지극복, 환경인식

Key word : Deep Learning, YOLO, Wheel-Leg Mobile Robot, Obstacle Overcoming, Environment Recognition

http://dx.doi.org/10.9728/dcs.2020.21.4.811

Received 01 February 2020; **Revised** 15 April 2020
Accepted 25 April 2020

***Corresponding Author; Youngshik Kim**
Tel: +※ 개인정보 표시제한
E-mail: youngshik@hanbat.ac.kr

Ⅰ. 서 론

비정형 환경에서 로봇을 자율 주행하기 위해서는 로봇 주변의 환경과 장애물을 인지(perception)하고 판단(decision)하는 것이 매우 중요하다. 객체를 인식하는 알고리즘에 관한 연구는 이전부터 활발히 진행되었다. 특히 사람의 얼굴인식, 움직이는 물체를 인식하여 추적하는 시스템과 자율주행 자동차 등의 실시간 물체 인식 등의 분야에서 많은 연구가 진행되었다[1]. 본 연구에서는 기존 연구들과 달리 딥러닝 알고리즘을 하드웨어 성능이 제한된 소형 모바일 로봇에 적용하여 실시간 장애물 인식 알고리즘을 구현하고, 이를 바탕으로 바퀴의 형태를 지능적으로 변형하여 장애물 환경에서 로봇을 효과적으로 주행하는 것을 목표로 한다. 이를 위해 CNN (Convolutional Neural Networks [2]) 알고리즘을 정형 및 비정형 환경을 효과적으로 이동할 수 있는 소형 모바일 로봇에 적용하였다.

본 연구에서 사용하는 모바일 로봇은 이전 연구에서 개발한 휠-레그 로봇(Wheel-Leg Robot)을 사용한다[3]. 모바일 로봇은 Fig. 1과 같이 바퀴의 모양을 변형하여 비정형 환경에서 장애물을 넘기 쉽게 설계되어 있다. 로봇의 제어를 위한 하드웨어는 오픈 소스 하드웨어인 Arduino Mega를 사용하였고, 실시간 딥러닝 연산처리를 하기 위해 GPU를 탑재한 저가형 임베디드(embedded) 보드인 Nvidia JetsonNano를 사용하여 시스템을 구성하였다.

딥러닝 기술을 이용한 실시간 객체 인식 기술로는 대표적으로 R-CNN(Regions with Convolutional Neural Networks)알고리즘이 있다. R-CNN의 경우엔 높은 mAP(mean Average Precision)를 제공하지만, 연산이 많아 초당 이미지 처리 속도가 느린 단점이 있다[4]. 하지만 YOLO(You Only Look Once)의 경우 R-CNN보다 mAP는 낮은 편이지만 단순한 처리로 속도가 매우 빠르다[5]. YOLO는 이미지 전체를 한 번에 바라보는 방식으로 Class를 분별한다. 이 방법은 기존의 다른 비전 기술과 비교할 때 2배 정도의 높은 성능을 보인다. 본 논문에서는 실시간으로 주변 환경을 인식하고 장애물을 판단하는 객체 인식을 진행하므로 초당 이미지 처리 속도가 비교적 빠른 YOLO를 사용한다. 최근 YOLOv3는 기존 YOLO에 비해 상당히 성능향상과 속도가 개선되었다[6]. 이에 본 연구에서는 YOLOv3를 채택하여 연구를 수행하였다.

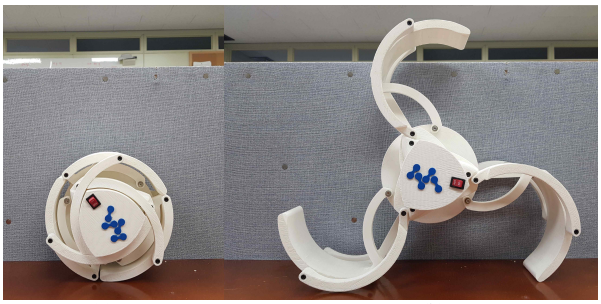
본 연구에서는 딥러닝 학습을 기반으로 소형 모바일 로봇이 지형/장애물을 인지/판단하고 장애물을 극복할 수 있는 알고리즘과 이를 구현하기 위한 하드웨어 시스템 설계를 연구하였다. 로봇은 먼저 제안된 객체 인식 알고리즘을 통해 장애물을 인식하고 판단한다. 평탄한 계단 같은 장애물을 만나게 되면 로봇은 객체 인식을 통하여 극복할 수 있는 장애물로 판단하여 원형 바퀴를 휠-레그 모드로 변형한 뒤 장애물을 넘어간다. 하지만 극복할 수 없는 장애물로 판단되는 경우는 로봇은 다른 대안 경로를 찾는다. 이처럼 로봇은 물체 인식 결과에 따라 휠-레그 바퀴의 형태를 자율적으로 변화하여 효과적인 주행을 할 수 있다.

그림 1. 휠-레그 로봇
Fig. 1. Wheel-Leg Robot

Ⅱ. 시스템설계와 알고리즘

2-1 모바일 로봇 시스템

본 연구에서 사용되는 모바일 로봇의 휠-레그 바퀴는 Fig. 2와 같다. 기어구조를 활용한 휠-레그 메커니즘은 그 크기를 환경에 따라 적절히 제어하여 장애물을 극복하는데 용이한 구조로 설계하였다. 휠-레그는 다리를 최대로 접혔을 때 최소 지름은 220mm이고, 완전히 다리를 펼쳤을 때 최대 다리의 지름은 500mm이다.



(a) (b)

그림 2. 가변형 휠-레그 (a) 휠-모드 (b) 레그-모드
Fig. 2. Transformable wheel-leg (a) Wheel-mode (b) Leg-mode

사용자 device(스마트폰 등)를 포함한 전체 로봇 시스템은 Fig. 3과 같이 구성되었다. 로봇 몸체에 Arduino와 카메라로부터 얻은 이미지 데이터를 실시간 딥러닝 연산 처리하기 위한 JetsonNano가 설치되어 있다. 로봇 외부의 스마트폰과 같은 사용자 device는 Bluetooth를 통해 로봇과 데이터 통신이 가능하다. 휠-레그 바퀴를 변형하기 위한 Servo모터와 Open CM 제어기는 로봇 바퀴의 내부에 위치하고, Arduino와는 Xbee를 이용하여 무선으로 통신한다. 로봇에 부착된 센서로는 전방의 물체의 유무를 감지하기 위한 초음파센서와 모바일 로봇의 방위각

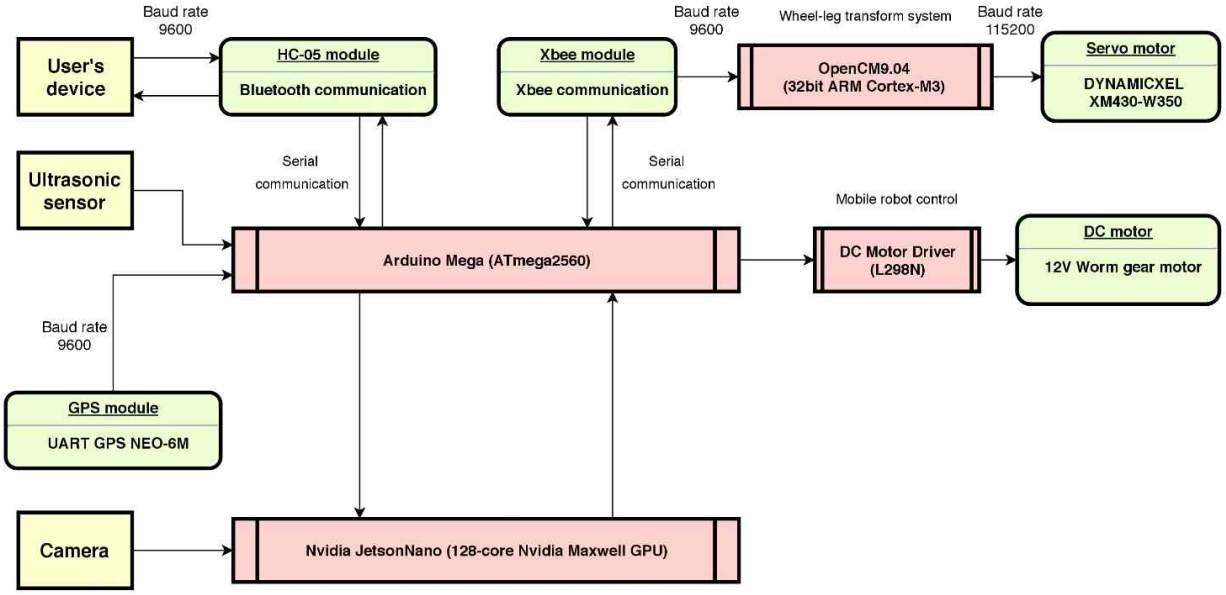


그림 3. 휠-레그 모바일 로봇 시스템 다이어그램
Fig. 3. System diagram for the wheel-leg mobile robot

을 정확하게 알아내기 위한 9축-IMU 센서가 있다. 또한 로봇의 현재 위치를 실시간으로 파악하기 위해 GPS 센서도 로봇의 상부에 설치되어 있다. 모바일 로봇의 구동을 위한 DC 모터와 모터 드라이버는 로봇의 내부에 위치한다. 로봇의 좌우 양 휠-레그는 DC모터에 의해 독립적으로 구동되고, 이 모터의 제어를 위해 2ch 모터 드라이버를 사용하였다. 모바일 로봇의 크기는 폭 305mm, 길이 667mm, 높이 200mm이다.

2-2 딥러닝 기반 비전 알고리즘 성능비교

YOLO 모델은 이미지를 일정한 그리드로 나누어 그리드의 신뢰도를 계산한다. 신뢰도는 그리드 내의 객체를 인식할 때 정확성을 반영한다. 다음 Fig. 4와 같이 처음에는 여러 경계 상자가 설정되지만, 신뢰도를 계산하여 가장 높은 정확성을 가지는 상자만 얻을 수 있다.

표 1에서는 YOLO와 다른 Detector들과의 Inference time을 비교하였다. Faster R-CNN [7]은 정확하다는 장점이 있다. 특히 겹쳐지거나 작은 사물에 대한 인식률이 비교적 높다. 하지만 느리다는 단점이 있다. SSD(Single Shot Detector [8])는 인식률과 속도가 상당히 개선되었다. 하지만 YOLO에 비해 사용하기 쉽지 않다. YOLO는 간단하고 학습하기 쉬우며 비교적 높은 속도와 정확도를 제공한다. 더 나아가 레이어의 수를 24개 대신 9개로 줄인 Fast YOLO의 경우는 정확도는 다소 낮아지지만, 속도는 3배 이상 증가한다. 최근 기존 YOLO의 속도와 정확성을 모두 개선한 YOLOv3개 개발되었고, SSD 대비 3배 이상 빠르며 더 정확한 것으로 보고되었다 [6]. 이에 본 연구에서는 실시간으로 물체를 인식하고 장애물을 극복하기 위해서 보다 빠른 속도가 필요하므로 기존의 YOLO보다 정확도와 속도가 모두 개선된 YOLOv3를 사용하여 연구를 진행하였다.

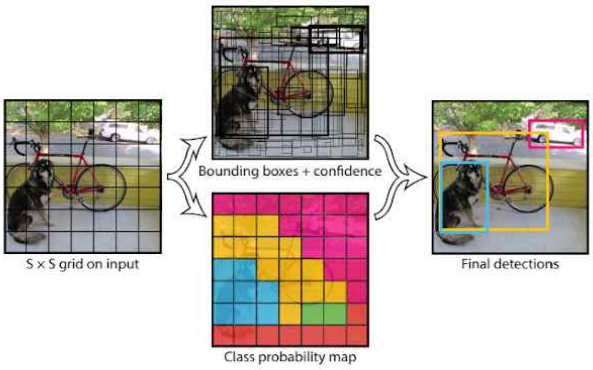


그림 4. YOLO의 객체검출 시스템
Fig. 4. YOLO detection system

표 1. 디텍터간의 성능비교
Table. 1. Inference time comparison

Method	mAP	FPS	Boxes
Faster R-CNN(VGG16)	73.2	7	300
Faster R-CNN(ZF)	62.1	17	300
YOLO	63.4	45	98
Fast YOLO	52.7	**155**	98
SSD300	72.1	58	7308
SSD500	**75.1**	23	20097

Ⅲ. YOLOv3 학습데이터 결과 분석

3-1 YOLOv3 학습

YOLOv3 학습을 위한 개발환경으로 Linux기반의 운영체제인 Ubuntu18.04를 사용하였다. GPU는 GTX 1080Ti를 사용하였다. YOLO_Mark를 통해 넘을 수 있는 장애물(계단)의 사진에 라벨링을 진행했다. Class는 1개로 설정하고 각 Box의 좌표 데이터를 text로 저장하였다. Convolutional layer는 Darknet19의 448x448 모델을 사용한다. 학습을 진행하면서 평균손실(avg loss)이 더 이상 줄지 않으면 중지하여야 한다. 과도하게 진행할 경우 Overfitting이 되어 이미지를 검출하는 데 어려움이 있으므로 과도하게 진행하지 않도록 한다.

본 연구에서는 100개의 데이터 세트로 3600회의 단일 Class 딥러닝 학습을 진행하였다. 학습을 중지하고 생성된 가중치 파일(Weights File)을 가지고 실험한 결과 Fig. 5와 같이 99% 계단인 것을 정확하게 검출하였다. JetsonNano는 데크스톱에 비해 연산 성능이 떨어지기 때문에 cfg 파일의 batch와 subdivisions를 1로 설정하고 width와 height를 256으로 파라미터값을 조정하였다.

그림 5. YOLOv3를 사용하여 계단을 인식하는 모습
Fig. 5. YOLOv3 Objet detection

3-2 학습 결과 분석

객체 검출 알고리즘에서는 mAP(Mean Average precision) 지표를 가지고 성능을 평가한다. 정밀도(Precision)와 재현율(Recall)을 모두 측정하여 실제로 참인 객체가 임계값(Threshold) 내에 포함될 때 정밀도를 골라내어 평균을 낸 것이 평균정밀도 (AP; Average Precision)이다. 여기서 Precision과 Recall의 계산식은 다음과 같다.

$$precision = \frac{TRUE\,Positive}{TRUE Positive + FALSE Positive} \quad (1)$$

$$recall = \frac{TRUE Positive}{TRUE Positive + FALSE\,Negative} \quad (2)$$

mAP는 평균정밀도를 한 번 더 평균 낸 값이다. mAP가 1에 가깝다면 높은 정밀도를 가지고 있는 객체검출 알고리즘이라고 볼 수 있다.

객체를 검출할 때 IoU (Intersection over Union [9]) 또한 널리 사용되는 평가 지표이다. IoU는 알고리즘이 객체를 검출할 때 모델이 예측한 결과와 실제로 물체가 존재하는 영역, 두 Box 간의 교집합과 합집합을 통해 IoU를 계산한다. GIoU (Generalized Intersection over Union [10])는 기존의 IoU가 겹치지 않는 Box에서 발생하는 단점들을 보완하였다.

$$IoU = \frac{|Area of Overlap|}{|Area of\,Union|} \quad (3)$$

$$GIoU = IoU - \frac{\left|\frac{Enclose\ Union}{Area of\,Union}\right|}{|Enclose\ Union|} \quad (4)$$

본 연구에서는 Python code를 이용하여 앞서 학습을 진행한 YOLOv3모델의 데이터를 각 지표마다 얻을 수 있었다[11]. Fig. 6 의 그래프에서 x축은 학습 횟수(Epoch) 값이다. Epoch가 증가함에 따라 mAP는 100%에 가까워지는 것을 확인할 수 있으며 GIoU도 1에 가까워지는 것을 볼 수 있다. 표 2는 학습결과 데이터를 요약하여 보여준다. 학습된 모델의 데이터와 실시간 물체인식 알고리즘을 데스크톱 환경(GTX-1080Ti)에서 실행시켰을 때의 초당 프레임(FPS)은 45였고, 로봇에 적용된 저가용 보드인 JetsonNano에서 실행시켰을 때는 5.2 FPS을 얻었다.

Ⅳ. 결 론

실험 결과, 카메라와 GPU 가 탑재된 저가형 임베디드 보드를 통해 객체(장애물)를 인식하는 것이 가능하였다. 적용된 알고리즘을 통해 로봇이 장애물을 판단하여 바퀴를 변형하여 넘어가거나 GPS 센서와 IMU 센서를 활용하여 다른 경로를 찾는 것을 확인하였다. 하지만 저가형 임베디드 보드를 사용함으로써 데스크톱의 성능보다는 많이 떨어진 FPS를 보여주었다. 본 모바일 로봇에 적용된 바퀴와 알고리즘을 이용하여 인간이 진입하기 어려운 지역의 무인 정찰로봇으로 활용하거나, 정해진 경로의 GPS 기반의 자율주행 시 장애물 극복 판단 등 여러 분야에서 활용 가능할 것으로 예상된다.

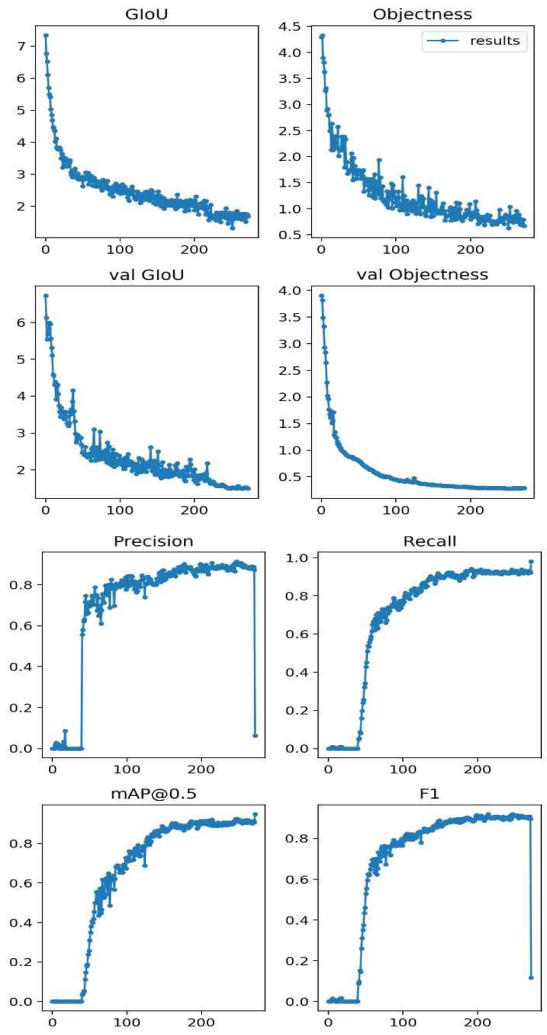


그림 6. YOLOv3 학습결과 데이터
Fig. 6. YOLOv3 learning results data

표 2. YOLOv3 학습결과 데이터 및 속도
Table. 2. YOLOv3 learning result data and speed

Precision	Recall	mAP	GIoU	FPS (GTX-1080Ti)	FPS (Jetson Nano)
0.117	0.949	0.981	1.69	45	5.2

감사의 글

이 연구는 2019학년도 한밭대학교 교내학술연구비의 지원을 받았음.

참고문헌

[1] Z. Zhao, P. Zheng, S. Xu and X. Wu, "Object Detection With Deep Learning: A Review," *in IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212-3232, Nov. 2019.

[2] Alex Krizhevsky and Ilya Sutskever and Geoffrey E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks", Advances in neural information processing systems, pp. 1097-1105, 2012

[3] S. Oh, Y. Yim, H. Kim, U. Kim, S. Kim, and Y. Kim, "Variable wheel-leg robot using gear structure." Proceedings of KIIT Conference, pp. 406-407, 2018

[4] RB Girshick, J Donahue, T Darrell, J Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation", Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580-58,. 2014.

[5] Joseph Redmon and Santosh Divvala and Ross Girshick and Ali Farhadi, "You Only Look Once: Unified, Real-Time Object Detection", Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779-788, 2016.

[6] Joseph Redmon and Ali Farhadi, "YOLOv3: An Incremental Improvement", arXiv:1804.02767, 2018.

[7] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", Advances in neural information processing systems, pp. 91-99, 2015

[8] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg, "SSD: Single Shot MultiBox Detector", European conference on computer vision, pp. 21-37, 2016.

[9] H. S. Lee, "Tutorials of Object Detection using Deep Learning How to measure performance of object detection", https://hoya012.github.io/blog/Tutorials-of-Object-Detection-Using-Deep-Learning-how-to-measure-performance-of-object-detection/

[10] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, Silvio Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 658-666, 2019.

[11] glenn-jocher, https://github.com/ultralytics/yolov3, 2019.

김현태(Hyuntae Kim)

2020년 : 한밭대학교 (학사)
2014년～2020년: 한밭대학교 전자제어공학과 학사
2014년～2020년: 한밭대학교 기계공학과 ICRS 학부생 연구원

※관심분야 : 회로 설계(Circuit design), 딥러닝(Deep learning), 제어공학(Control Engineering) 등

김영식(Youngshik Kim)

2003년 : University of Utah대학원(공학석사)
2008년 : University of Utah대학원(공학박사)
2008년～2009년: University of Utah
2009년～2009년: 방위사업청
2009년～2011년: DGIST
2011년～현 재: 한밭대학교 기계공학과 교수

※관심분야 : 스마트 액추에이터(smart actuator), 모션제어 (motion control), 생체모방로봇(bio-inspired robot), 센서융합(sensor fusion), 인공지능(AI) 등