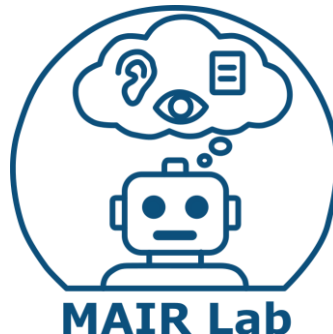


운영체제 실제 (ROS2) Introduction

안인규 (Inkyu An)

<https://mairlab-km.github.io/courses/operating-systems-practice-2025fall>



수업계획

- 진행
 - PPT 강의교재
(<https://mairlab-km.github.io/courses/operating-systems-practice-2025fall>)
 - 이론수업 → 실습수업
- 준비사항
 - 노트북 (Ubuntu 22.04 설치)
- 실습로봇: Turtlebot4 (CLEARPATH)



수업계획

- TA (류재우, 이성빈)
 - 실습 중 어려운 부분을 질문
 - TA hours: 화, 목 13:00~15:30, 자율주행 자동차 스튜디오



수업계획

•Week 1

- Sep 2: ROS2 Introduction: Why ROS2?
- Sep 4: ROS2 Introduction: Setting up the development environment

•Week 2

- Sep 9: ROS2 Nodes and Data Communication
- Sep 11: Practice – ROS2 Nodes and Data Communication

•Week 3

- Sep 16: ROS2 Transformation System (TF2)
- Sep 18: Gazebo Practice: Robot Creation (Part 1)

•Week 4

- Sep 23: Gazebo Practice: Robot Creation (Part 2)
- Sep 25: ROS2 Sensing

•Week 5

- Sep 30: Gazebo Practice: 2D Lidar
- Oct 2: Gazebo Practice: Depth Camera

•Week 6

- Oct 7: SLAM Theory
- Oct 9: Gazebo Practice: SLAM

•Week 7

- Oct 14: Path Planning Theory
- Oct 16: Gazebo Practice: Path Planning

•Week 8

- Oct 21 & Oct 23: Midterm Exam

•Week 9

- Oct 28: ROS2 Programming – Build and Package Files
- Oct 30: ROS2 Programming – Package Design

•Week 10

- Nov 4: Real Robot Practice – Sensing (Part 1)
- Nov 6: Real Robot Practice – Sensing (Part 2)

•Week 11

- Nov 11: Real Robot Practice – YOLOv8
- Nov 13: ROS2 Topic Programming (Python)

•Week 12

- Nov 18: ROS2 Service Programming (Python)
- Nov 20: Real Robot Practice – SLAM (Part 1)

•Week 13

- Nov 25: Real Robot Practice – SLAM (Part 2)
- Nov 27: Real Robot Practice – Path Planning (Part 1)

•Week 14

- Dec 2: Real Robot Practice – Path Planning (Part 2)
- Dec 4: Real Robot Practice – Path Planning (Part 3)

•Week 15

- Dec 9: Final Project
- Dec 11: Final Project

수업계획

•Week 1

- Sep 2: ROS2 Introduction: Why ROS2?
- Sep 4: ROS2 Introduction: Setting up the development environment

•Week 2

ROS2 functions

- Sep 9: ROS2 Nodes and Data Communication
- Sep 11: Practice – ROS2 Nodes and Data Communication

•Week 3

- Sep 16: ROS2 Transformation System (TF2)
- Sep 18: Gazebo Practice: Robot Creation (Part 1)

•Week 4

- Sep 23: Gazebo Practice: Robot Creation (Part 2)
- Sep 25: ROS2 Sensing

ROS2 Simulation

•Week 5

- Sep 30: Gazebo Practice: 2D Lidar
- Oct 2: Gazebo Practice: Depth Camera

•Week 6

ROS2 applications

- Oct 7: SLAM Theory
- Oct 9: Gazebo Practice: SLAM

•Week 7

- Oct 14: Path Planning Theory
- Oct 16: Gazebo Practice: Path Planning

•Week 8

- Oct 21 & Oct 23: Midterm Exam

•Week 9

Robotics Programming

- Oct 28: ROS2 Programming – Build and Package Files
- Oct 30: ROS2 Programming – Package Design

•Week 10

Real robot practice

- Nov 4: Real Robot Practice – Sensing (Part 1)
- Nov 6: Real Robot Practice – Sensing (Part 2)

•Week 11

- Nov 11: Real Robot Practice – YOLOv8
- Nov 13: ROS2 Topic Programming (Python)

•Week 12

- Nov 18: ROS2 Service Programming (Python)
- Nov 20: Real Robot Practice – SLAM (Part 1)

•Week 13

- Nov 25: Real Robot Practice – SLAM (Part 2)
- Nov 27: Real Robot Practice – Path Planning (Part 1)

•Week 14

- Dec 2: Real Robot Practice – Path Planning (Part 2)
- Dec 4: Real Robot Practice – Path Planning (Part 3)

•Week 15

- Dec 9: Final Project
- Dec 11: Final Project

수업계획

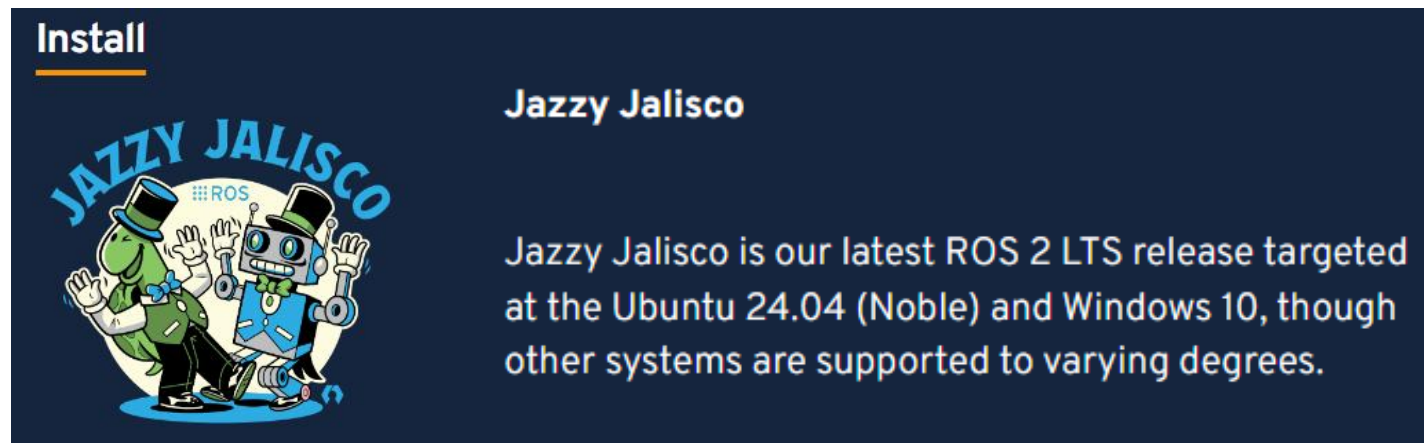
- 평가
 - 중간고사 (40%)
 - 프로젝트 (40%) – 중간고사 이후 공지
 - 과제 (10%)
 - 출석 (5%)
 - 보너스 점수 (5%) – 수업참여(질문, 대답)+0.5, 학우를 도와주면+0.5
- 성취기반평가
 - 정확한 기준은 중간고사 이후에 결정 및 공지
 - 상대평가가 아닌 절대평가

ROS2를 배우는 이유?

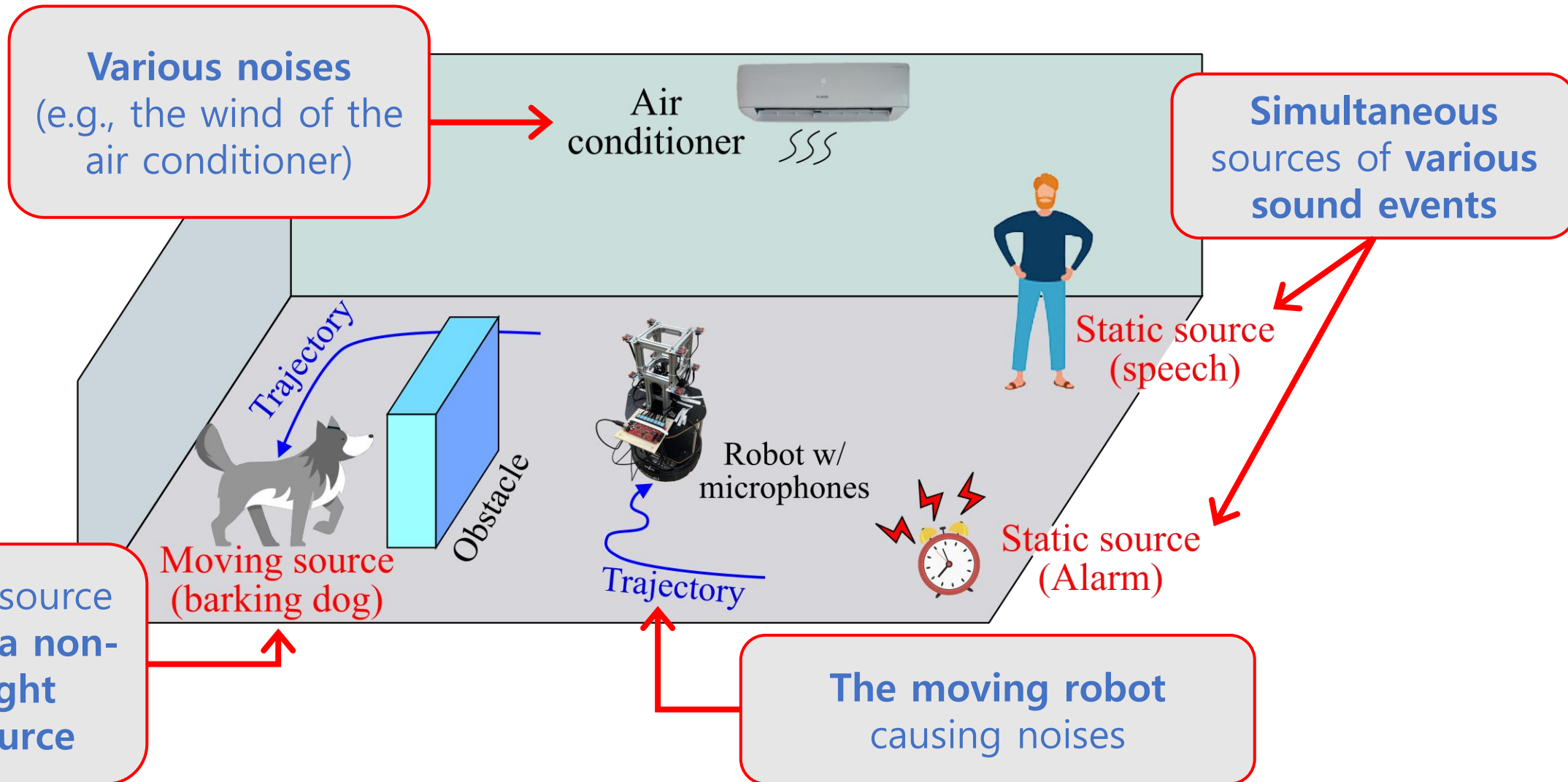
- ROS: Do not reinvent the wheel!
 - 모듈화 (Modularity): 로봇 시스템을 센서, 제어기, 알고리즘 등 작은 node로 나누어 개발하고 연결할 수 있도록 함
 - 표준화된 통신: Publisher-Subscriber 구조를 통해 노드 간 데이터를 안정적으로 주고받게 함.
 - 재사용과 공유: 한 번 만든 소프트웨어 package를 다른 로봇이나 프로젝트에서 쉽게 재사용할 수 있도록 하고, 커뮤니티에서 자유롭게 공유할 수 있음.
- 로봇 소프트웨어 개발을 위한 공통 프레임워크 제공
 - 센서, 액추에이터, 제어 알고리즘, 인공지능 모듈 등을 쉽게 연결하고 재사용할 수 있도록 지원함.

Computer Science도 로봇 어렵지 않아요!

- 많은 Computer Science 학생들이 로봇 (하드웨어)를 다룬다는 것에 두려움이 있음
- 사실 로봇의 하드웨어 장치 (모터, 센서 등)를 직접 다룰 필요없음
 - Open Source Platform이 잘 제공됨 (Robot Operating System, ROS)
 - 하드웨어 장치 지원과 관련된 많은 Open Source가 있음 (다운받아서 설치 후 바로 사용 가능!)
 - 로봇을 움직이는 부분 (제어와 관련)도 Open Source로 되어서 쉽게 사용 가능!
- 즉, CS 학생들도 하드웨어 지식이 없어도 로봇을 다루기 어렵지 않음



나는 ROS/ROS2를 이용해 어떤 것을 했나?



나는 ROS/ROS2를 이용해 어떤 것을 했나?

- 로봇 청각 기반 기술 연구 (실세계의 어려움 극복)
 - 음원의 위치를 찾는 연구 (Sound Source Localization) → **SSL node**
 - 중첩된 음성 신호를 분리하는 연구 (Speech Separation) → **Speech Separation node**
- 멀티모달 센싱 통합
 - 로봇의 mapping (지도 생성) 연구 → **Mapping node / Navigation node**
- 실시간 로봇 응용
 - 드론을 위한 경로생성 연구 → **Path-Planning node**

Sound Source Localization



Navigation

Sec. V-G: Navigating to the NLOS source (clapping sound)

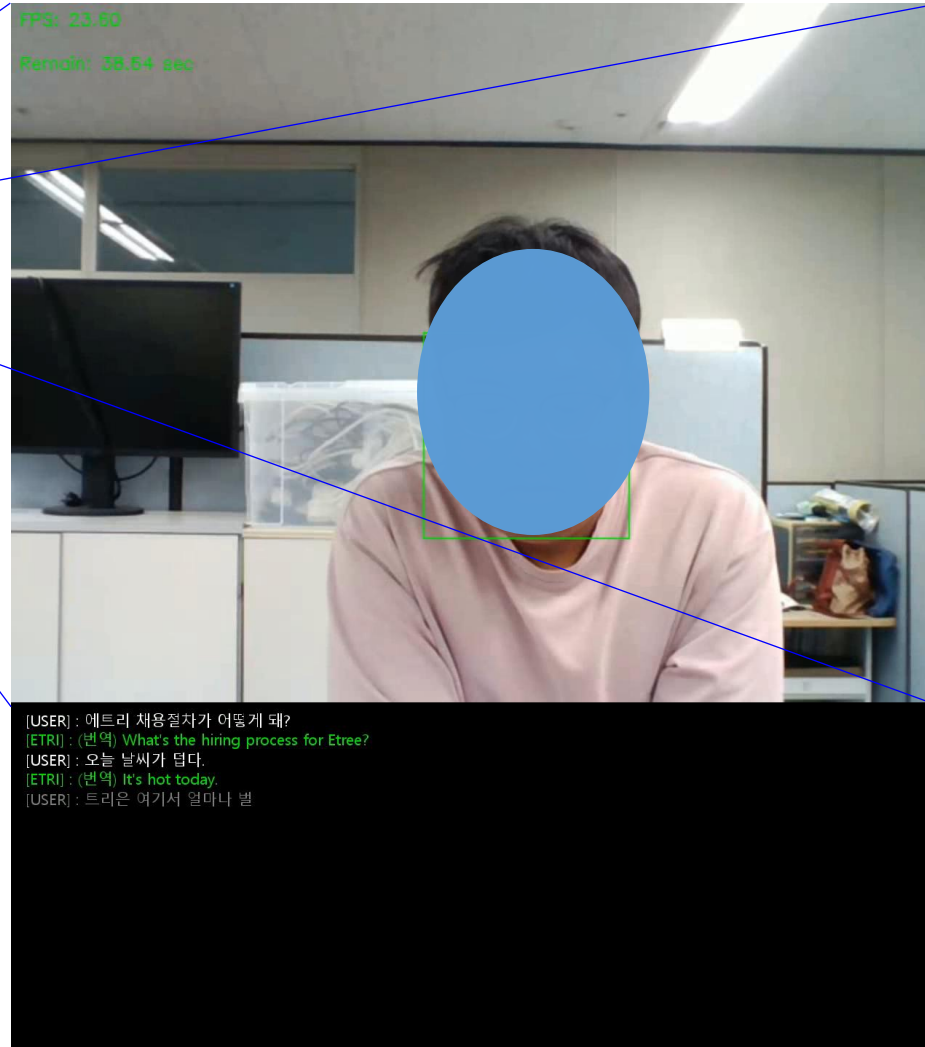
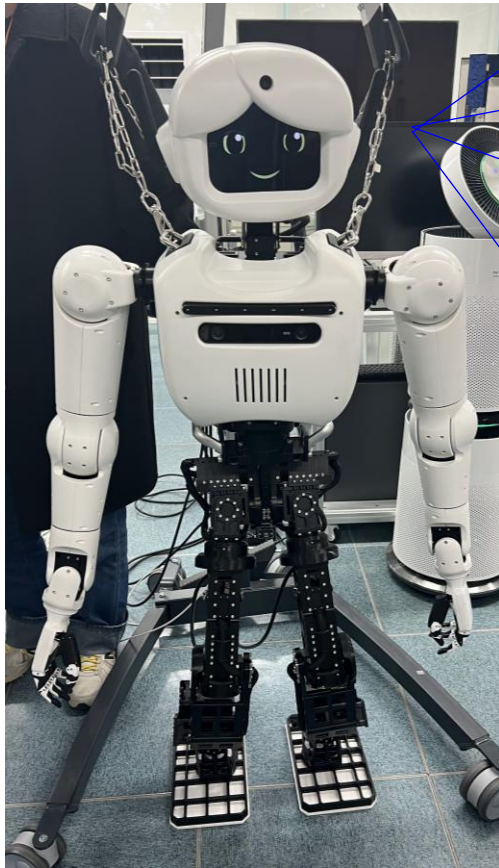


Testing environment

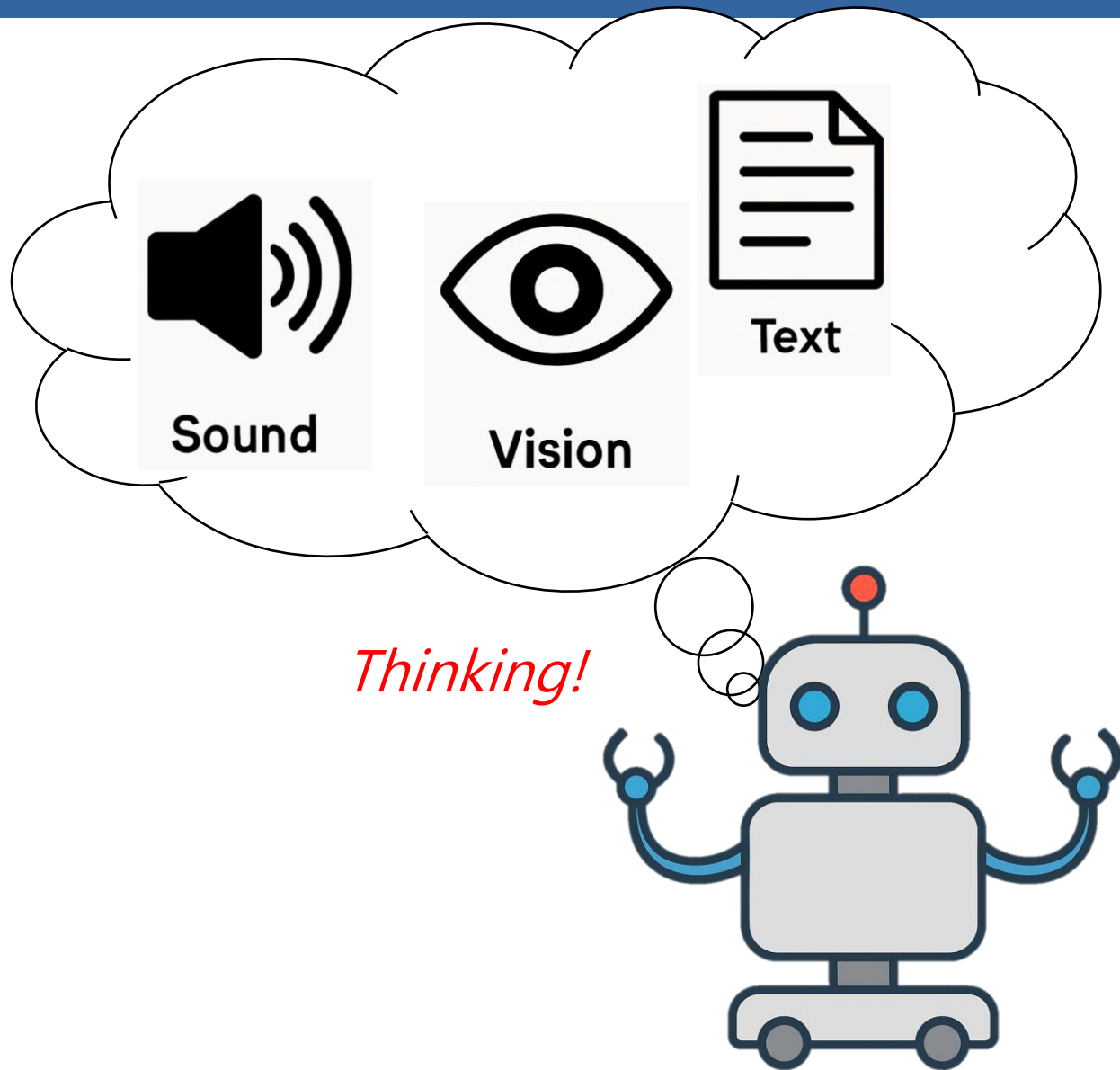
Path planning (Drone)



Human-Robot Interaction



ROS를 이용해 어떤 것을 할 수 있는가?



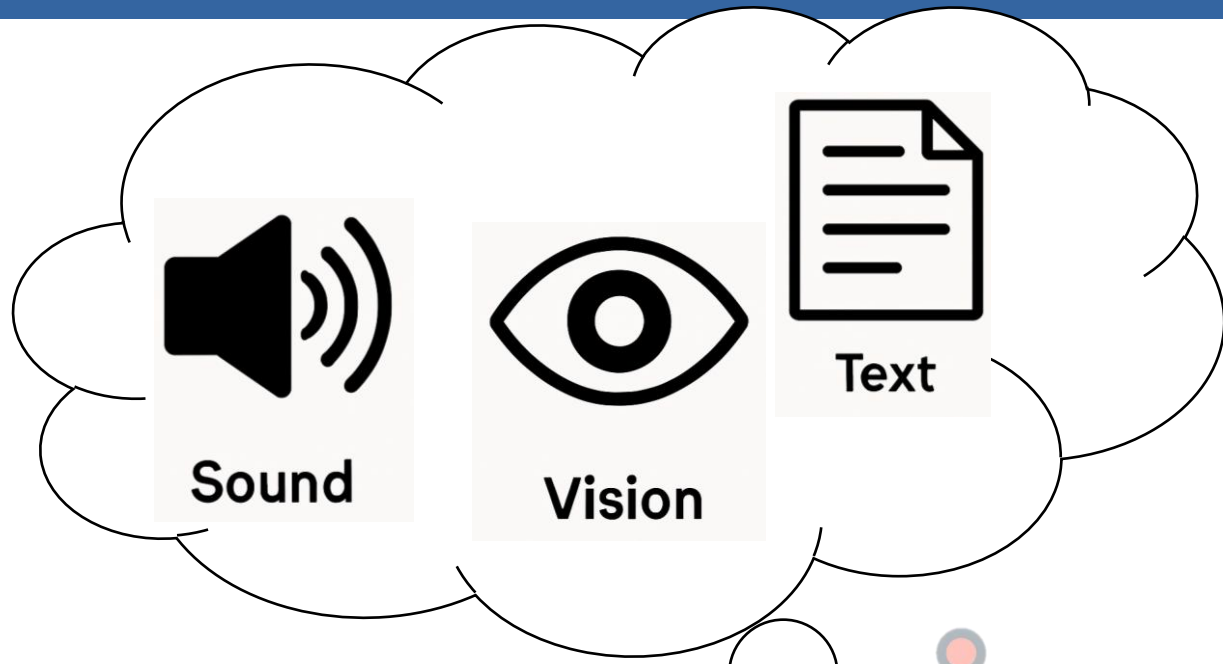
ROS를 이용해 어떤 것을

Research Topics

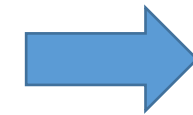
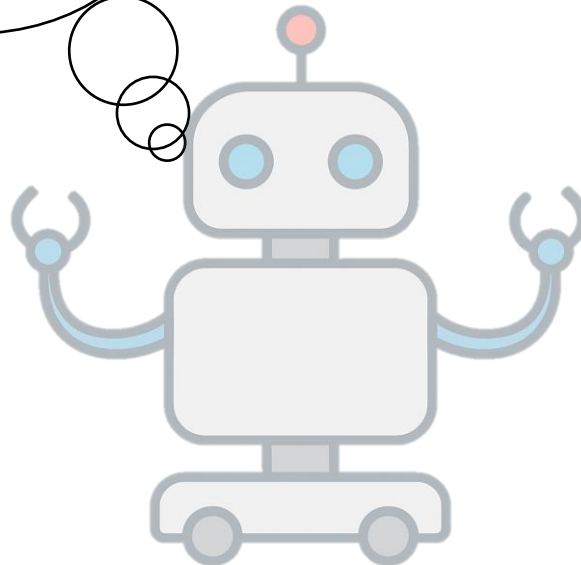
Vision-Sound Language Action
model

Sound-based Embodied AI

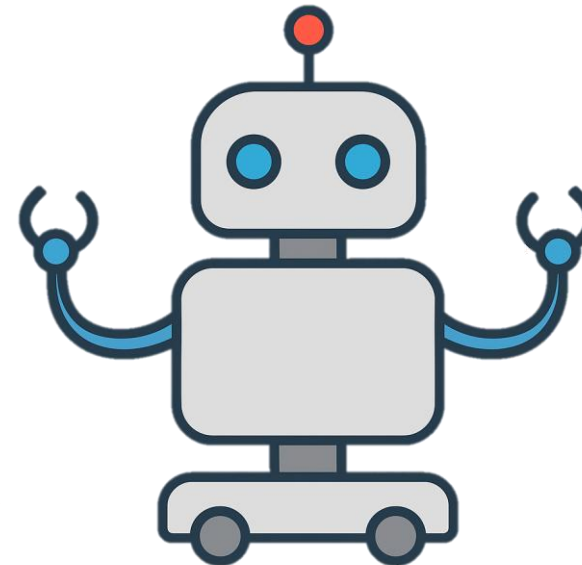
Robot Auditions in Real
Environments



Thinking!



*Do an
action!*



ROS를 이용해 어떤 것을 할 수 있는가?



GO2 사족보행로봇

Industries that utilize this research topic



Industries that utilize this research topic



현대자동차 로봇틱스랩

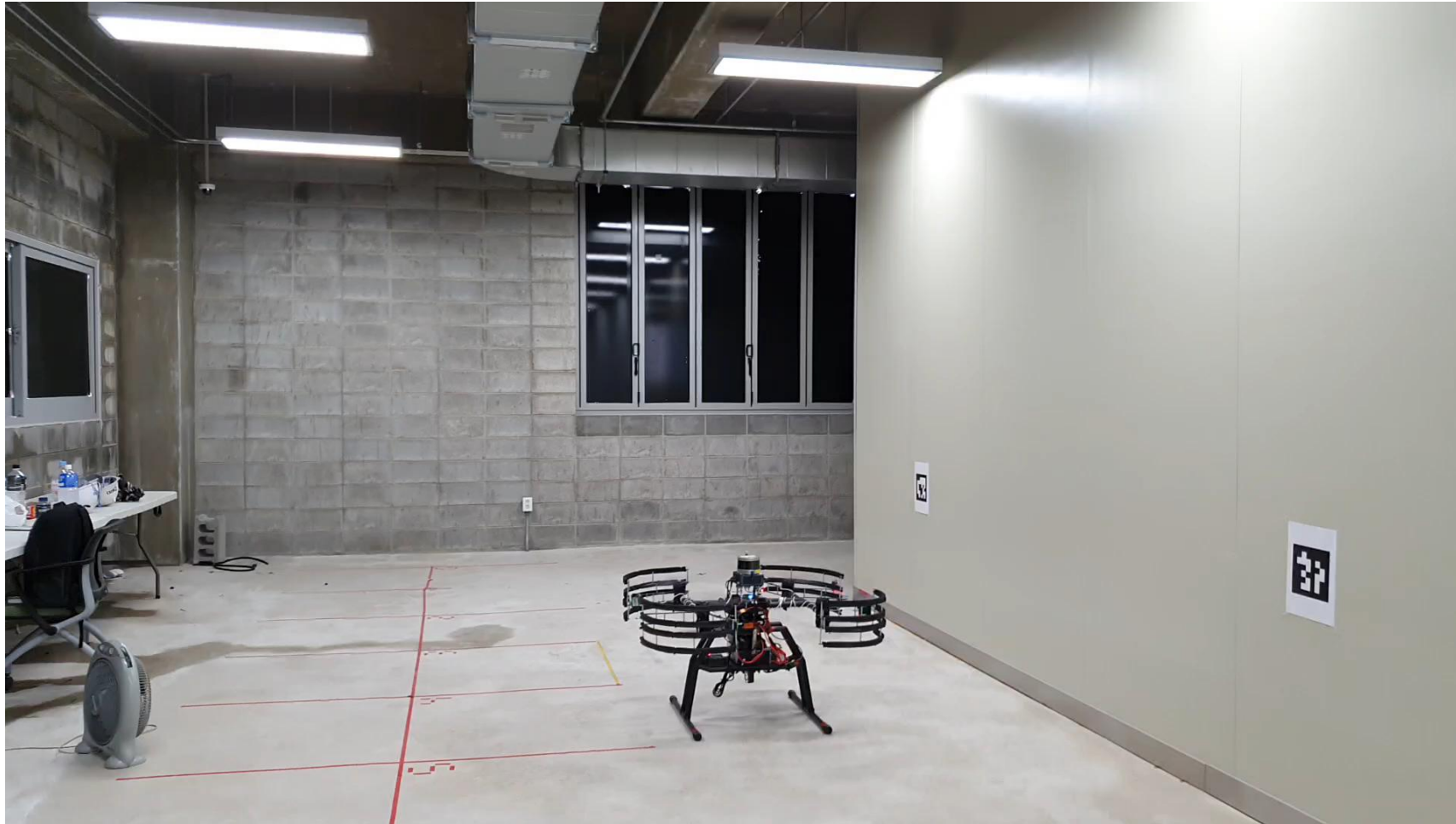


배달의민족 로봇배달

LG전자, 클로이

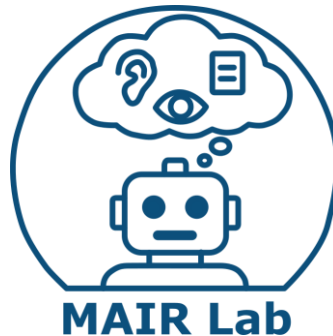


Don't be afraid of failure!



ROS Introduction

운영체제의 실제
안인규 (Inkyu An)



What is ROS?

- ROS is an open-source, meta-operating system for your robot!
 - **Open-source:** 소스 코드가 공개되어 누구나 자유롭게 사용, 수정, 배포할 수 있는 소프트웨어
 - **Meta-operating system:** 운영체제의 위에서 동작하면서 다수의 자원 (디바이스, 시스템, 사용자 등)을 통합 제어하고 최적화하는 시스템

For operating various robots, Sensors, ...



What is ROS?

- The Robot Operating System (ROS) is a collection of **tools**, **software libraries**, and **documentation** facilitating the development and sharing of robotics software
- Its federated package model allows unrelated entities to create and reuse compatible software

History

- In 2007, Willow Garage provided resources for researchers to create general-purpose robotics software that could be **reused** and **shared**. From the start, it was an effort involving a community of people at different institutions working on the core infrastructure and various packages
- In 2014, stewardship of ROS was transferred to the Open Source Robotics Foundation (now referred to as Open Robotics) a 501(c)(3) non-profit organization.
- In late 2022 the majority of Open Robotics was acquired by Intrinsic, an Alphabet company. The Open Source Robotics Foundation controls the ROS trademarks and websites but the primary development team for ROS now works at Intrinsic.
- In the intervening years, ROS has been adopted as a defacto standard in industry and academia.

Why ROS?

- ROS는 전 세계적으로 큰 사용자·기여자 커뮤니티가 있어서, 문제 해결이나 질문에 대한 도움을 쉽게 받을 수 있습니다.
- 로봇을 **시각화, 제어, 시뮬레이션**할 수 있는 다양한 인프라를 제공합니다.
- ROS에는 많은 패키지가 있으며, 그중 일부는 구현하기 어려운 **최신 알고리즘**을 포함하고 있어 연구와 개발에 큰 도움이 됩니다.
- 많은 로봇들이 ROS 인터페이스를 지원하기 때문에, 한 번 ROS를 배우면 다양한 로봇을 다룰 수 있습니다.
- ROS는 **오픈소스**이므로 블랙박스가 없고, 소스코드를 직접 이해·수정·확장할 수 있습니다.
- 자신이 발견한 기능이나 알고리즘을 커뮤니티에 기여할 수 있고, 이를 통해 연구자나 개발자로서의 **평판**을 높일 수 있습니다.
- 많은 기업들이 ROS를 **핵심 기술 역량**으로 여기며, 비록 ROS를 직접 사용하지 않는 기업이라도 ROS 경험은 **로봇 전반에 대한 이해도**를 보여주는 중요한 지표로 평가됩니다.

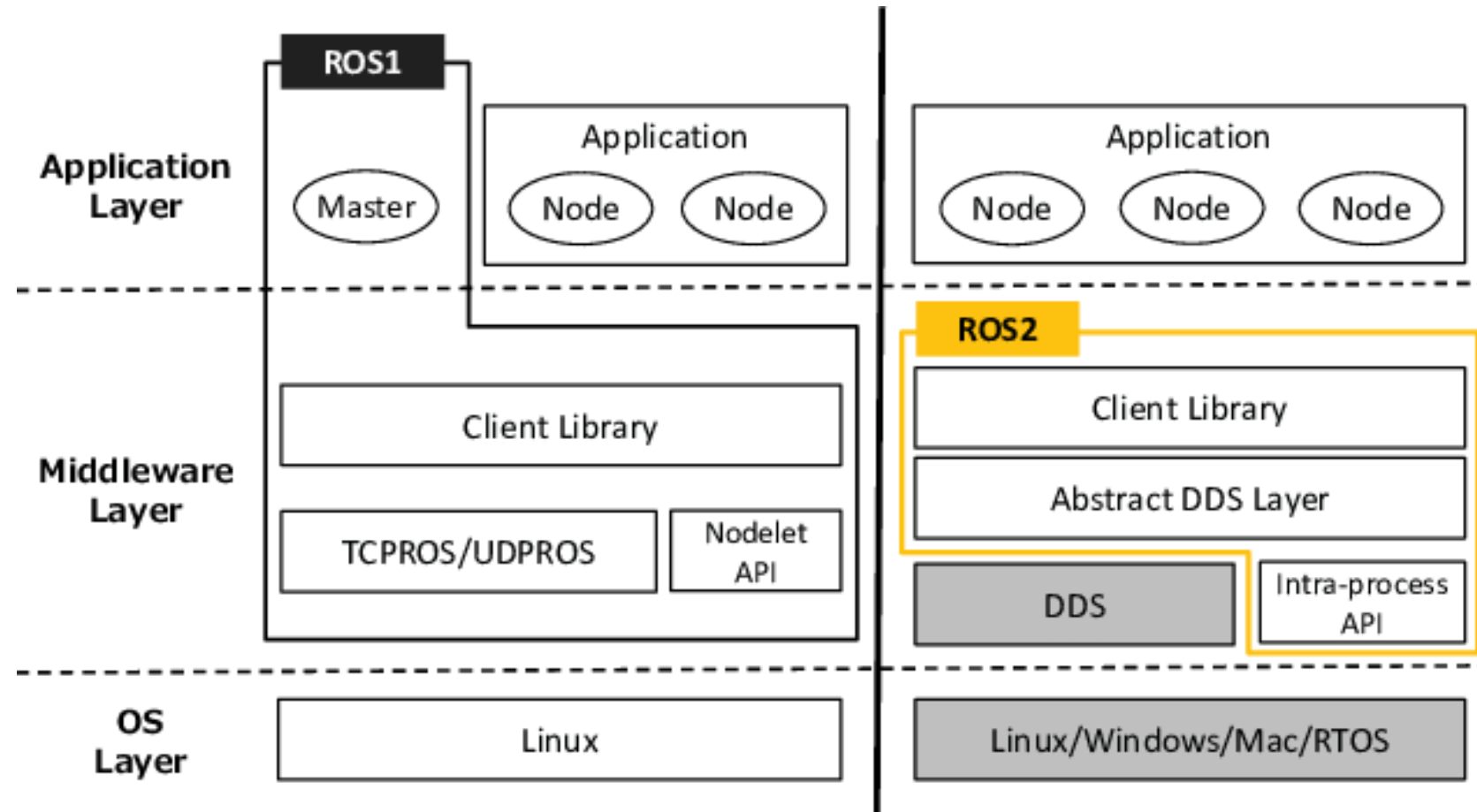
Why ROS2?

- ROS 2 was started in 2007, first announced in 2014, and first released in 2017. It was created to address several issues identified with ROS 1:
 - ROS 1 was designed for use with a single robot and hence had a central component (roscore) that needed to be accounted for.
 - ROS 1 communicates all data unencrypted over the network and is insecure. ROS 2 has a formal security model.
 - ROS 1 has no real-time performance guarantees. ROS 2 is working on enabling real-time control.
 - ROS 1 used a custom service/message middleware whereas ROS 2 uses industry standard versions.
- Overall ROS 2 is less complete than ROS 1 (in that it has fewer packages and more rough edges to its declared functionality), but it is growing every day..

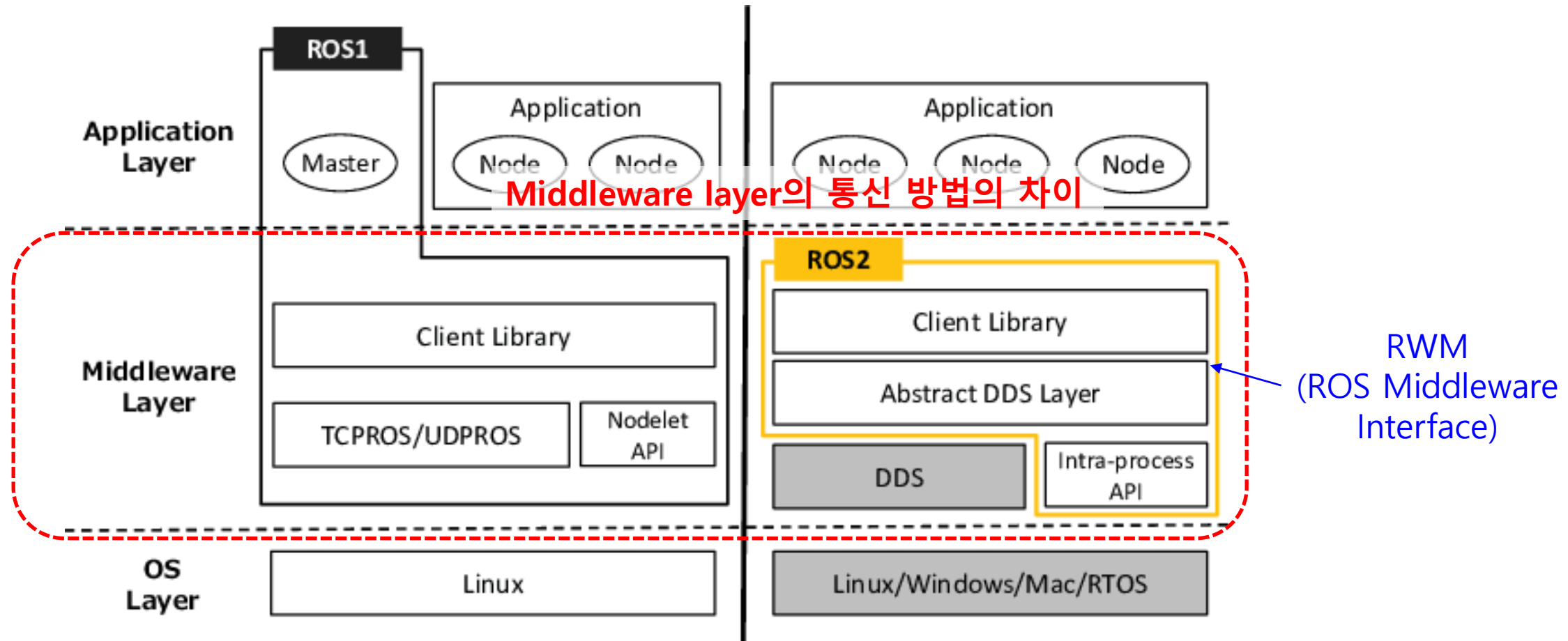
Why ROS2?

구분	ROS 1	ROS 2
출시 시기	2010년 (공식 첫 배포)	2017년 (첫 배포)
시스템 구조	단일 로봇 중심, roscore라는 중앙 컴포넌트 필요	분산 시스템 지원, 중앙 집중식 구조 제거
보안(Security)	네트워크 통신 암호화 없음 → 보안 취약	정식 보안 모델 (암호화/인증) 지원
실시간성(Real-time)	실시간 성능 보장 불가	실시간 제어 가능 하도록 설계 (DDS 기반)
통신 미들웨어	자체 제작(custom) 미들웨어	산업 표준 DDS (Data Distribution Service) 사용
운영체제 지원	주로 Ubuntu Linux 중심	Windows, macOS, 임베디드 RTOS까지 지원
패키지 생태계	방대한 패키지, 안정적	아직 패키지 수 적음, 기능 미완성 부분 존재
적용 분야	연구용/프로토타입 중심	연구 + 산업·상업용 로봇까지 확장

Why ROS2?



Why ROS2?



Why ROS2?

ROS1

ROS2

구분	TCPROS (ROS1)	UDPROS (ROS1)	DDS (ROS2)
기반	TCP	UDP	산업 표준 DDS
신뢰성	높음 (순서/전송 보장)	낮음 (손실 가능)	QoS(Quality of Service)에 따라 조정 가능
지연 시간	상대적으로 큼	낮음 (빠름)	실시간 제어 가능
보안	없음	없음	암호화·인증 지원
확장성	제한적	제한적	멀티로봇/분산 환경 최적화
주 용도	일반 메시지 전송	대용량 센서 데이터	모든 로봇 통신 (범용)

DDS?

항목	Fast DDS (eProsima)	Cyclone DDS (Eclipse)	Connnext DDS (RTI)	GurumDDS (GurumNetworks)
라이선스	Apache 2.0 (오픈소스)	EPL v2.0 (오픈소스)	Commercial / 연구용 무료	Commercial
기본 포함 여부	O (ROS 2 기본 포함 (Default))	O (ROS 2 기본 포함)	X (별도 설치 필요)	X (별도 설치 필요)
성능	빠른 discovery, 낮은 latency	안정적인 통신, 중간 정도 성능	매우 우수 (산업용 실시간 환경에 적합)	RTOS에 적합, 실시간 대응 강점
지원 플랫폼	Linux, macOS, Windows	Linux 중심	다수 OS (RTOS 포함)	RTOS 포함
메모리 사용량	중간	낮음 (임베디드에 적합)	높음 (기능이 많아서)	낮음
사용 복잡도	낮음 (설정 간단)	낮음	중간 이상 (라이선스 등록, 설정 필요)	중간
QoS 지원	대부분 지원	대부분 지원	완전 지원	대부분 지원

QoS (Quality of Service): ROS2의 통신의 "Reliability (신뢰성)", "Durability (지속성)", "Timing (타이밍)" 등을 설정하는 장치

DDS?

항목	Fast DDS (eProsima)	Cyclone DDS (Eclipse)	Connex DDS (RTI)	GurumDDS (GurumNetworks)
라이선스	Apache 2.0 (오픈소스)	EPL v2.0 (오픈소스)	Commercial / 연구용 무료	Commercial
기본 포함 여부	O (ROS 2 기본 포함 (Default))	O (ROS 2 기본 포함)	X (별도 설치 필요)	X (별도 설치 필요)
성능	빠른 discovery, 낮은 latency	안정적인 통신, 중간 정도 성능	매우 우수 (산업용 실시간 환경에 적합)	RTOS에 적합, 실시간 대응 강점
지원 플랫폼	Linux, macOS, Windows	Linux 중심	다수 OS (RTOS 포함)	RTOS 포함
메모리 사용량	중간	낮음 (임베디드에 적합)	높음 (기능이 많아서)	낮음
사용 복잡도	낮음 (설정 간단)	낮음	중간 이상 (라이선스 등록, 설정 필요)	중간
QoS 지원	대부분 지원	대부분 지원	완전 지원	대부분 지원

모두 같은 DDS 표준을 기반으로 설계되었기 때문에, 다른 종류의 RMW 끼리 통신이 가능

DDS?

- **더 빠른 로봇 제어 주가 필요할 때? (Latency 개선 필요)**
 - Fast DDS → RTI Connex
- **Raspberry Pi에 ROS2를 올려야 할 때?**
 - Fast DDS → Cyclone DDS
- **로봇의 센서 데이터 유실이 없어야 할 때? (실시간성 확보 필요)**
 - Fat DDS → RTI Connex 또는 GurumDDS

Usage Statistics

- At the start of Fall 2024:
 - 95% of commits to ROS are to ROS 2 repositories.
 - 72% of ROS downloads are from ROS 2 repositories.
- At the start of Fall 2023:
 - 74% of commits to ROS are to ROS 2 repositories.
 - 55% of ROS downloads are from ROS 2 repositories.
- At the start of Fall 2024:
 - 63% of commits to ROS are to ROS 2 repositories.
 - 40% of ROS downloads are from ROS 2 repositories.

ROS/ROS2 Version

Distro	Release date	Poster
Noetic Ninjemys		
ROS Noetic Ninjemys	May 23rd, 2020	
ROS Melodic Morenia	May 23rd, 2018	
ROS Lunar Loggerhead	May 23rd, 2017	
ROS Kinetic Kame	May 23rd, 2016	
ROS Jade Turtle	May 23rd, 2015	
ROS Indigo Igloo	July 22nd, 2014	
ROS Hydro Medusa	September 4th, 2013	

Distro	Release date	Logo	EOL date
Iron Irwini	May 23rd, 2023		November 2024
Humble Hawksbill	May 23rd, 2022		May 2027
Galactic Geochelone	May 23rd, 2021		December 9th, 2022
Foxy Fitzroy	June 5th, 2020		June 20th, 2023
Eloquent Elusor	November 22nd, 2019		November 2020
Dashing Diademata	May 31st, 2019		May 2021
Crystal Clemmys	December 14th, 2018		December 2019
Bouncy Bolson	July 2nd, 2018		July 2019

ROS 2 Document / Community

ROS 2 Documentation: Humble

ROS 2 Documentation

You're reading the documentation for an older, but still supported, version of ROS 2. For information on the latest version, please have a look at Kilted.

ROS 2 Documentation

The Robot Operating System (ROS) is a set of software libraries and tools for building robot applications. From drivers and state-of-the-art algorithms to powerful developer tools, ROS has the open source tools you need for your next robotics project.

Since ROS was started in 2007, a lot has changed in the robotics and ROS community. The goal of the ROS 2 project is to adapt to these changes, leveraging what is great about ROS 1 and improving what isn't.

Are you looking for documentation for a particular ROS package like MoveIt, image_proc, or octomap? Please see [ROS Index](#) or check out [this index of per-package documentation](#).

This site contains the documentation for ROS 2. If you are looking for ROS 1 documentation, check out the [ROS wiki](#).

If you use ROS 2 in your work, please see [Citations](#) to cite ROS 2.

Getting started

- Installation
 - Instructions to set up ROS 2 for the first time
- Tutorials
 - The best place to start for new users!
 - Hands-on sample projects that help you build a progression of necessary skills
- How-to Guides
 - Quick answers to your "How do I...?" questions without working through the [Tutorials](#)
- Concepts
 - High-level explanations of core ROS 2 concepts covered in the [Tutorials](#)
- Contact
 - Answers to your questions or a forum to start a discussion

The ROS 2 project

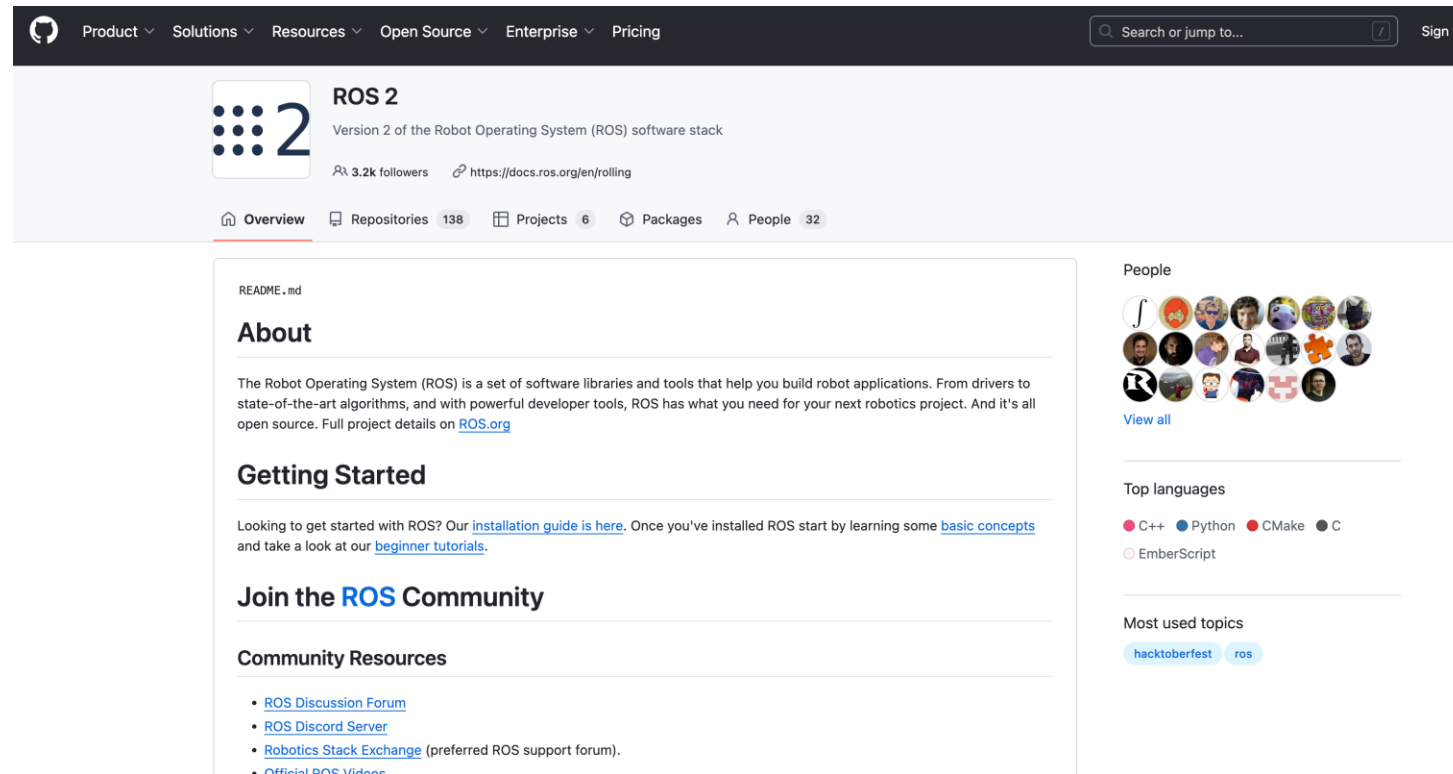
If you're interested in the advancement of the ROS 2 project:

<https://docs.ros.org/en/humble/>

ROS Resources: [ROS Homepage](#) | [Media and Trademarks](#) | [Documentation](#) | [ROS Index](#) | [How to Get Help](#) | [Q&A Help Site](#) | [Discussion Forum](#) | [Service Status](#)

<https://discourse.openrobotics.org/>

ROS2 Source code



The screenshot shows the GitHub repository page for ROS 2. The header includes navigation links for Product, Solutions, Resources, Open Source, Enterprise, and Pricing, along with a search bar and a 'Sign in' button. The repository name 'ROS 2' is prominently displayed, followed by the description 'Version 2 of the Robot Operating System (ROS) software stack'. It shows 3.2k followers and a link to the documentation. The 'About' section describes ROS as a set of software libraries and tools for building robot applications. The 'Getting Started' section provides links to the installation guide, basic concepts, and beginner tutorials. The 'Join the ROS Community' section lists community resources like the ROS Discussion Forum, ROS Discord Server, Robotics Stack Exchange, and Official ROS Videos. The right sidebar features a 'People' section with user avatars, a 'Top languages' section showing C++, Python, CMake, and C, and a 'Most used topics' section with 'hacktoberfest' and 'ros'.

Product ▾ Solutions ▾ Resources ▾ Open Source ▾ Enterprise ▾ Pricing

Search or jump to... / Sign in

ROS 2

Version 2 of the Robot Operating System (ROS) software stack

3.2k followers <https://docs.ros.org/en/rolling>

Overview Repositories 138 Projects 6 Packages People 32

README.md

About

The Robot Operating System (ROS) is a set of software libraries and tools that help you build robot applications. From drivers to state-of-the-art algorithms, and with powerful developer tools, ROS has what you need for your next robotics project. And it's all open source. Full project details on [ROS.org](https://ros.org)

Getting Started

Looking to get started with ROS? Our [installation guide is here](#). Once you've installed ROS start by learning some [basic concepts](#) and take a look at our [beginner tutorials](#).

Join the ROS Community

Community Resources

- [ROS Discussion Forum](#)
- [ROS Discord Server](#)
- [Robotics Stack Exchange](#) (preferred ROS support forum).
- [Official ROS Videos](#)

People

[View all](#)

Top languages

C++ Python CMake C
EmberScript

Most used topics

hacktoberfest ros

<https://github.com/ros2>

ROS2 (Humble) Installation



- Ubuntu 22.04 ← Recommended
- Window 10
- macOS (Mojave 10.14)
- <https://docs.ros.org/en/humble/Installation/Ubuntu-Install-Debs.html>

ROS2 (Humble) Installation

- Set locale: System에서 UTF-8 문자 인코딩을 사용할 수 있도록 설정 (ROS package, message 등에 다양한 유니코드 문자를 활용함)
- Setup sources: ROS2가 제공하는 apt 저장소를 system에 등록 및 적절한 인증 키 추가
- Install ROS 2 Packages: ROS2 설치
 - ros-humble-desktop: GUI, 튜토리얼, RViz 포함된 전체 환경
 - ros-humble-ros-base: GUI 제외, 통신 및 메시지 기능 중심
 - ros-dev-tools: 개발 도구 (컴파일러 등)
- Sourcing the setup script: ROS2 환경을 terminal 에서 활용하기 위해서 환경 변수 및 경로를 로드

ROS2 (Humble): Configuring environment



- Prepare your ROS2 environment
- <https://docs.ros.org/en/humble/Tutorials/Beginner-CLI-Tools/Configuring-ROS2-Environment.html#add-sourcing-to-your-shell-startup-script>

ROS2 (Humble): Configuring environment

- **Source the setup files or add sourcing to our shell startup script:** `'source /opt/ros/humble/setup.bash'`
- **Check environment variables**
- **The 'ROS_TODMAIN_ID' variables**
 - ROS2는 DDS (Data Distribution Service) middleware 기반 통신하며, Domain ID는 논리적인 네트워크 구분을 만드는 핵심 매커니즘
 - 예: 여러 로봇이 동일한 Wi-fi에 연결될 때, 서로 다른 Domain ID를 부여하여 구분짓게 함
 - Network port 충돌을 피하기 위해, Linux에서 일반적으로 0~101 범위를 권장
- **The 'ROS_LOCALHOST_ONLY' variable**