개발환경 구축 방법

1. 개요
2. 운영체재
3. Firmware, Middleware, Agent, Simulation
4. Ground Control
5. 영상정보처리 도구
6. `ros-gz-bridge` 설치

연세대학교
Space-Y 항공팀 제어부
2025/11/11

1. 개요

해당 문서는 '항공팀 개발 환경 통일안'에서 정한 개발 환경을 구축하는 방법을 기술한다. 해당 문서는 '개발환경 통일안'에 맞는 버전을 설치하는 방법을 기술한다. 따라서 개발 환경 개정 시에는 해당 문서의 최신화도 병행되어야한다.

운영체재(OS)	Ubuntu 22.04
Firmware	PX4 v1.16.0 – Stable Release
Middleware	ROS2 Humble
Agent	Micro-XRCE-DDS-Agent v2.4.3
Compiler (C++)	g++ v11.4.0
CMake	v3.22.1
Python	Python v3.10.12
OpenCV	v4.5.4
Simulation	Gazebo Simulator "Harmonic"
Ground Control	QGC v5.0.7

<표 1> 개발환경 통일안(2025/09/25)

2. 운영체재

2-1. 개요

기체의 제어 프로그램은 ROS2의 Node 형태로 작성된다. 따라서 현재 채택된 ROS2 Humble의 권장 실행 환경인 Ubuntu 22.04을 개발 환경의 운영체재로 활용한다.

Ubuntu는 초보자도 쉽게 사용할 수 있는 Linux Distro로, 가장 많이 사용되는 Linux Distro 중 하나이다.

2-2. 설치하기

아래의 링크를 참고하여 설치를 진행하면 된다.

https://ubuntu.com/tutorials/install-ubuntu-desktop#1-overview

'2. Download Ubuntu Image'에서는 Ubuntu 22.04를 다운받으면 된다. 아래 링크에서 'Desktop Image'를 다운받으면 된다.

https://releases.ubuntu.com/jammy/

'3. Creat a bootable USB stick'에서 USB의 Flash를 완료 후 Windows에서 해당 USB (일반적으로 E:\와 F:\ 디스크)를 사용할 수 없으니 포맷하라는 내용의 메시지가 뜰 수도 있다.. 이때 USB를 포맷하지 않도록 주의한다.

'4. Boot from USB flash drive'에서 Ubuntu를 설치하고자 하는 컴퓨터에 USB를 연결하고 컴퓨터 부팅을 진행하게 된다. 이때 자동으로 Ubuntu 설치가 진행되지 않으면 재부팅 중 부트 메뉴를 띄운다. 일반적으로 F2 혹은 F12를 활용하면 부트 메뉴에 접속할 수 있다. 이후 부팅 우선순위의 1순위를 연결한 USB로 바꿔준다. 또, secure boot 기능을 찾아 꺼주도록 한다.

'5. Installation setup'에서 필요 드라이브를 설치할거냐는 질문을 한다. 이때 "Install third party software for graphics and Wi-Fi hardware and additional media formats"를 선택하여 필요한 드라이브 등을 설치해주면 나중에 편하다.

'6. Type of Installation'에서는 Ubuntu 설치 유형을 선택한다. "Erase disk and install

Ubuntu"를 선택하면 기존 디스크를 밀어버리기 때문에 원하는 설치 유형에 따라 조심히 선택을 하도록 한다.

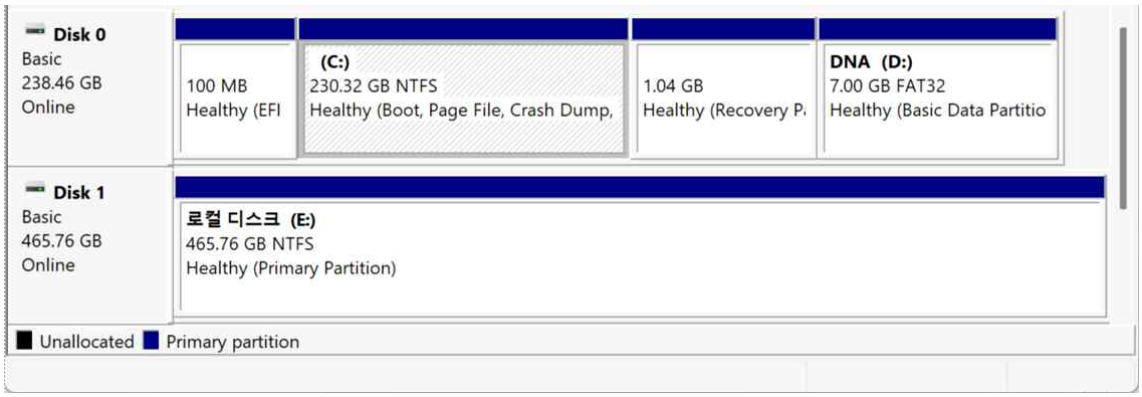
설치 완료 후 Ubuntu의 새로운 버전인 24.05로 업데이트하겠냐는 메시지가 뜬다. 이때 업데이트하지 않도록 조심한다. 이 외 보안 관련 소프트웨어 업데이트 등은 해도 상관 없다.

2-3. (선택사항) USB 포맷하기

Boot drive로 사용한 USB를 다시 USB로 사용하려고 포맷을 시도하면 포맷이 불가능하다는 메시지가 뜬다. 아래는 Windows에서 해당 USB를 포맷하는 방법이다.

1. 명령 프롬프트(`cmd`)를 관리자 권한으로 실행한다.
2. `C:₩Users₩User> diskpart` 명령을 입력하면 디스크 관리를 할 수 있는 새로운 콘솔이 뜬다.
3. `DISKPART> list disk`를 입력하여 뜬 목록에서 USB의 디스크 번호를 확인한다. `Size`에서 USB의 용량을 갖고 있는 디스크를 확인하면 된다.
4. `DISKPART> select disk 1`를 입력하여 USB에 대한 행동을 할 수 있도록 한다. 이때 USB의 디스크 번호가 1번이 아니라 다른 번호면 그 번호를 사용하면 된다.
5. `DISKPART> clean`을 하여 USB를 포맷한다.

여기까지 하면 파일 탐색기에서 USB가 인식되지 않는다. 이제 Windows+X에서 Disk Management를 들어간다.



아래쪽에 이런 직사각형 모양으로 디스크 파티션 구조가 뜬다. Disk1을 찾으면 상태가 Healthy가 아닌 Not Initialized와 비슷한 내용으로 되어있을 것이다. 해당 직사각형을 우클릭 후 New Simple Volume 선택, 이후 따로 설정을 바꿀 필요 없이 다음 단계로 넘어가면 USB 설정이 끝난다. 이후 파일 탐색기에서 USB를 확인할 수 있고, 해당 디스크를 사용하려면 포맷해야한다는 내용이 뜰 수도 있다. 포맷 해주면 일반적인 USB처럼 다시 사용할 수 있다.

3. Firmware, Middleware, Agent, Simulation

3-1. 시스템 개요

기체의 제어는 Flight Controller(FC)에서 담당하게 된다. FC로는 오픈소스 제어 시스템인 Pixhawk를 사용한다. Pixhawk에는 ROS2 Node를 활용해 topic의 형태로 여러 명령을 전달할 수 있다. 이때 PX4에는 uXRCE-DDS-client가, 제어 프로그램이 탑재된 Companion Computer(CC)에는 uXRCE-DDS-agent가 실행중이어야 ROS2와 PX4 사이의 통신이 가능해진다.

개발 중인 제어 프로그램은 실(實)기체에 탑재시 기체 파손 및 사고의 가능성이 있기 때문에 시뮬레이션을 활용한다. 기체 제어의 시뮬레이션을 위해서는 FC를 개발 환경에서 재현해야하므로 개발 환경에 PX4 Firmware 및 uXRCE-DDS-agent를 설치해야한다.

3-2. ROS2 Humble 설치하기

ROS2 Humble은 아래의 링크를 참고하여 설치를 진행하면 된다. 터미널은 Ctrl+Alt+T로 띄울 수 있다. Ubuntu 22.04 터미널에서 붙이기 단축키는 Ctrl+Shift+V이다.

https://docs.ros.org/en/humble/Installation/Ubuntu-Install-Debs.html

`sudo apt install ros-humble-desktop`까지 진행했으면 ROS2 Humble의 설치는 완료된 것이다.

3-3. PX4 Firmware 설치하기

PX4 firmware 및 시뮬레이션, agent 등은 아래 링크를 참고하여 설치한다. `$ git clone https://...` 명령어를 직접 입력하면 github 사용자 ID를 입력하라는 문구가 뜰 수도 있다. 복사-붙이기를 활용하면 github 사용자 ID를 물어보지 않는다. 개발환경 통일안에 따라 v1.16.0을 설치하기 위해서는 아랫니 명령어들을 따라한다. Firmware 설치에 대한 추가적인 정보는 아래의 링크를 보면 알 수 있다.

```
$ git clone https://github.com/PX4/PX4-Autopilot.git
$ cd PX4-Autopilot
$ git checkout v1.16.0
$ make submodules clean
$ make px4_sitl
```

https://docs.px4.io/main/en/ros2/user_guide.html#install-px4

git 명령어가 없다는 오류가 뜰 수도 있다. 아래 명령으로 설치 가능하다.

```
$ sudo apt install git
```

만약 이미 v1.16.0 이외의 버전을 설치한 경우 아래의 명령어를 통해 Firmware v1.16.0을 설치할 수 있다.

```
$ cd PX4-Autopilot
$ make clean
$ make distclean
$ make submodules clean
$ make px4_sitl
```

3-4. uXRCE Agent 설치하기

기체 제어노드의 ROS2 메시지를 PX4에서 이해할 수 있는 형태의 메시지로 변환하기 위해 해당 Agent가 필요하다. 아래의 링크를 통해 설치할 수 있다.

https://docs.px4.io/main/en/middleware/uxrce_dds#install-standalone-from-source

문서를 따라가며 `MicroXRCEAgent udp4 -p 8888`까지 실행했으면 다음 단계부터는 따라하지 않아도 된다. 아래와 같은 내용이 뜨면 Agent도 정상적으로 설치된 것이다.

```
$ MicroXRCEAgent udp4 -p 8888
[1760385760.313256] info  | UDPv4AgentLinux.cpp    | init    | running...      | port: 8888
[1760385760.314153] info  | Root.cpp               | set_verbose_level       | logger
setup    | verbose_level: 4
```

3-5. ROS2 참조하기

새 터미널에서 ROS2 관련 명령어를 사용하기 위해서는 항상 아래의 명령을 통해 ROS2 디렉토리를 참조해야한다.

```
$ source /opt/ros/humble/setup.bash
```

이때 위의 명령을 .bashrc 파일에 추가해주면 ROS2 관련 명령을 사용할때마다 입력할 필요가 없어진다. 아래의 명령을 통해 .bashrc에 추가한다.

```
$ echo "source /opt/ros/humble/setup.bash" >> ~/.bashrc
```

이후 현재 터미널을 닫고 새로운 터미널을 다시 열면 .bashrc 파일의 수정사항이 반영된다.

3-6. px4_msgs workspace 구축하기

PX4 제어는 **px4_msgs**를 통해 이루어진다. 따라서 PX4 제어를 위한 ROS2 workspace에는 항상 **px4_msgs**가 있어야한다. 만약 PX4 workspace가 여러개 있으면 같은 내용(**px4_msgs**)을 가진 파일이 불필요하게 많을 뿐만 아니라, PX4 Firmware 업데이트시 모든 workspace의 **px4_msgs**를 최신화해줘야하는 버거움이 있다. 따라서 **px4_msgs**만 있는 workspace를 구축하고 해당 workspace를 참조하여 개발을 진행하면 더욱 쾌적한 개발 경험을 할 수 있다. 아래의 명령을 따라 **px4_msgs**를 위한 workspace를 구축하고 참조한다.

```
$ mkdir -p ~/px4_ros2_ws/src/
$ cd ~/px4_ros2_ws/src/
$ git clone https://github.com/PX4/px4_msgs.git
$ cd ..
$ colcon build
$ cd
$ echo "source ~/px4_ros_ws/install/setup.bash" >> ~/.bashrc
```

위의 명령을 실행하면 .bashrc 가장 마지막 줄에 "source ~/px4_ros2_ws/install/setup.bash"가 추가된다. 이후 터미널을 재실행하면 수정사항이 반영된다.

workspace 구축 중 colcon 명령이 존재하지 않는다는 메시지가 뜰 수도 있다. 설치해주면 된다.

```
$ sudo apt update
$ sudo apt install python3-colcon-common-extensions
```

4. Ground Control

4-1. 개요

기체를 지상에서 제어하기 위해 Ground Control(GC)가 필요하다. 이는 QGroundControl(QGC)이라는 앱을 활용해 이루어진다. 시뮬레이션 실행에서 QGC의 버전은 크게 중요하지 않다. Gazebo Classic을 활용한 시뮬레이션에서는 QGC가 아예없어도 실행이 가능하지만, Gazebo Simulation "Harmonic" 이상의 버전에서는 QGC가 있어야 시뮬레이션에서 구현된 가상 기체의 비행이 가능하다.

(참고)Gazebo의 버전 이름은 알파벳 순서로 명명한다. v1.x는 "Acropolis", v2.x는 "Blueprint", v3.x는 "Citadel" 등과 같이 알파벳 순서로 명명한다. 따라서 "Harmonic"은 v8.x임을 알 수 있다. 아래 링크에서 Gazebo 버전에 따른 이름을 확인할 수 있다.

https://gazebosim.org/docs/latest/releases/

4-2. 설치하기

설치는 아래의 링크를 참고하여 진행하면된다.

https://docs.qgroundcontrol.com/master/en/qgc-user-guide/getting_started/download_and_install.html#ubuntu

설치 과정에서 "`$ chmod +x ....`"를 실행하는 단계가 있다. 이 명령은 다운 받은 `QGroundControl-x86_64.AppImage`와 같은 디렉토리에서 진행하거나, 해당 파일의 디렉토리를 지정 후 실행하면 된다. 실행할 때의 편의를 위해 다운 받은 파일을 `~/`로 옮기는 것을 추천한다. 대부분의 경우 다운된 파일은 `~/Downloads/`에 있다.

5. 영상정보처리 도구

5-1. 개요

영상정보처리를 위한 많은 라이브러리가 있다. OpenCV는 Python을 활용한 영상정보처리에서 많이 사용되는 라이브러리 중 하나이다.

5-2. 설치 여부 확인하기

3. Firmware, Middleware, Agent, Simulation을 잘 따랐으면 OpenCV까지 정상적으로 설치된다. 설치 여부를 확인하기 위해 아래 단계를 따르면 된다. 먼저 터미널에 `python3`을 입력하여 shell을 열어주면 아래와 같은 글을 볼 수 있다.

```
Python 3.10.12 (main, Aug 15 2025, 14:32:43) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information
>>>
```

아래 명령을 실행하면 OpenCV의 버전을 확인할 수 있다.

```
>>> import cv2
>>> print(cv2.__version__)
```

"`4.5.4`"라는 문구가 뜨면 성공적으로 설치가 된 것이다. 이후 "`exit()`"을 입력하여 shell을 종료할 수 있다.

만약 설치가 돼있지 않다면 아래 명령을 통해 설치하면 된다.

```
$ sudo apt install python3-opencv=4.5.4+dfsg-9ubuntu4
libopencv-dev=4.5.4+dfsg-9ubuntu4
```

“import cv2” 실행 시 아래와 같은 문구가 뜰 확률이 크다.

```
A module that was compiled using NumPy 1.x cannot be run in
NumPy 2.2.6 as it may crash. To support both 1.x and 2.x
ersions of NumPy, modules must be compiled with NumPy 2.0.
Some module may need to rebuild instead e.g. with 'pybind11>=2.12'.

If you are a user of the module, the easiest solution will be to
downgrade to 'numpy<2' or try to upgrade the affected module.
We expect that some modules will need time to support NumPy 2.

Traceback (most recent call last): File "<stdin>", line 1 in <module>
  File "/home/user/.local/lib/python3.10/sit-packages/cv2/__init__.py", line 8
, in <module>
    from .cv2 import *
AttributeError: _ARRAY_API not found
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/home/user/.local/lib/python3.10/sit-packages/cv2/__init__.py", line 8
, in <module>
    from .cv2 import *
ImportError: numpy.core.multiarray failed to import
```

이는 OpenCV v4.5.4가 예전 버전이기 때문에 최신 NumPy 버전인 v2.0 이상과 호환이 되지 않아 발생하는 문제이다. 아래 명령을 통해 NumPy 버전을 v2.0 미만으로 내리면 문제가 해결된다.

```
$ pip install "numpy<2" --upgrade -force-reinstall
```

1.xx 버전 중 가장 최근 버전이 설치된다. 이는 아래와 같이 numpy 버전을 확인할 수 있다. 이후 다시 OpenCV의 버전을 확인하여 정상적으로 설치 됐는지 확인해본다.

```
$ python3
Python 3.10.12 (main, Aug 15 2025, 14:32:43) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information
>>> import numpy
>>> print(numpy.__version__)
```

영상 스트리밍을 위해 필요한 GStreamer도 같이 설치돼있어야 한다.

```
$ python3
Python 3.10.12 (main, Aug 15 2025, 14:32:43) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information
>>> import cv2
>>> print(cv2.getBuildInformation())
```

많은 내용이 뜨는데 그 중 GStreamer에 YES (1.19.90)와 같은 말이 있으면 된다. NO가 써져 있으면 다시 설치해본다. PX4 환경 구축시에도 설치 되고 QGC 설치 과정에도 gstreamer 설치 단계가 있기에 이미 설치돼있을 것이다.

6. ros_gz_bridge 설치하기

6-1. 개요

시뮬레이션 실행 시 기체의 센서값은 Gazebo 토픽으로 발행된다. 이를 제어 노드에서 접근하기 위해서는 ROS2 토픽으로 변환해줘야하고, 이는 **ros_gz_bridge**를 통해 할 수 있다.

6-2. 설치

아래의 명령을 통해 설치하면 된다. 사용 ROS2 및 Gazebo 버전에 따라 설치 명령이 달라지니 잘 확인한다. 아래의 링크를 통해 버전에 따른 설치방법을 확인할 수 있다.

```
$ sudo apt install ros-humble-ros-gzharmonic
```

https://github.com/gazebosim/ros_gz/tree/ros2/ros_gz_bridge#readme

6-3. Bridge 구축

설치가 완료되면 새로운 콘솔에서 아래의 명령어를 실행하여 Gazebo 토픽을 ROS 토픽으로 바꿔준다. bridge의 실행 신택스는 **ros2 run ros_gz_bridge parameter_bridge** <기존 토픽 이름>**@**<ROS 토픽 형식><방향><Gazebo 토픽 형식>으로, 방향에는 **@**, **[**, **]** 중 하나를 입력한다. **@**는 Gazebo ↔ ROS, **[**는 Gazebo → ROS, **]**는 ROS → Gazebo 방향의 **bridge**를 구축한다. 이때 ROS 토픽의 형식과 Gazebo 토픽의 형식 사이 변환이 지원되지 않으면 에러 혹은 정보 누락이 발생할 수 있다. 변환 가능한 토픽은 아래의 링크를 통해 확인 가능하다.

https://index.ros.org/p/ros_gz_bridge/