

GOOD MORNING!

早上好!

안녕하세요!

DAY 2



DAY 1 RECAP



2 PROJECTS

- Mini Project (Individual Team)
 - For learning techniques

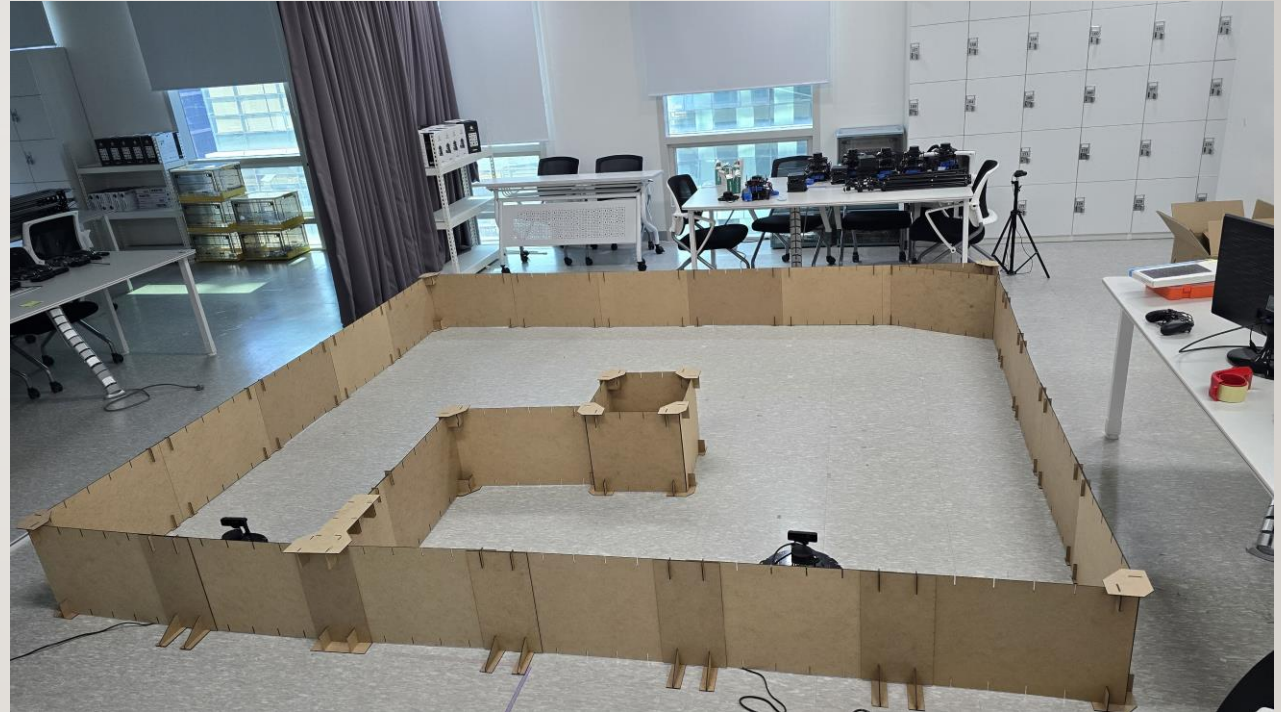
1	로봇 AI 시스템 개발 프로세스 이해	시스템 개발 프로세스의 이해 개발 환경 구축
2	프로젝트에 필요 기술 검증	AI VISION 기술 탐색 및 검증
3	프로젝트에 필요 기술 검증	로봇 AMR 제어 기술 탐색 및 검증
4/5	MINI 프로젝트	통합 시스템 설계 및 개발

2 PROJECTS

- Final Project (2 Teams in One)

6	프로젝트에 필요 기술 검증	웹 시스템 모니터 기술 탐색 및 검증
6-9	파이널 프로젝트	통합 시스템 설계 및 개발
10	최종 프레젠테이션 및 시연	시스템 발표 및 시연

MINI PROJECT DESCRIPTION



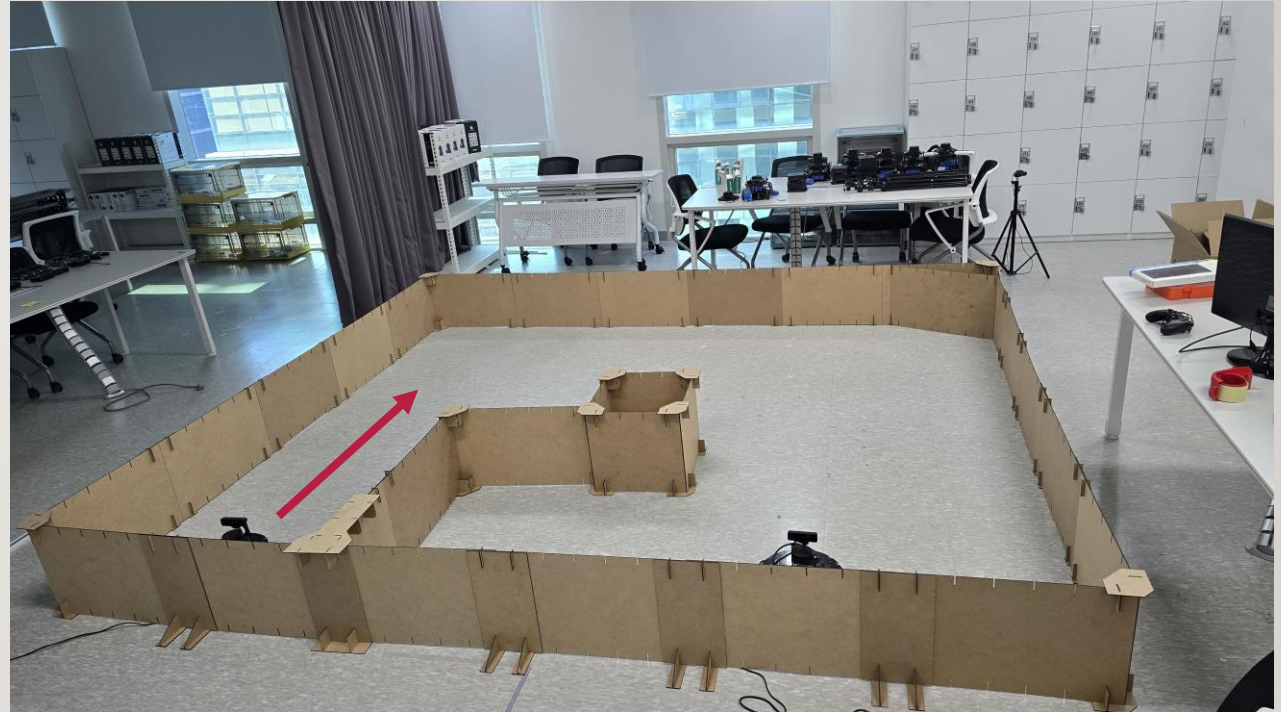
KEY SUBSYSTEM (MODULES) TO DEVELOP

- Detection Alert
 - Camera Capture
 - Object Detection
 - Send messages to other subsystems
- AMR Controller
 - Receive messages and act accordingly
 - Move using (SLAM) with Obstruction avoidance
 - Target Acquisition (Obj. Det.) and Tracking
 - Follow target using camera and motor control

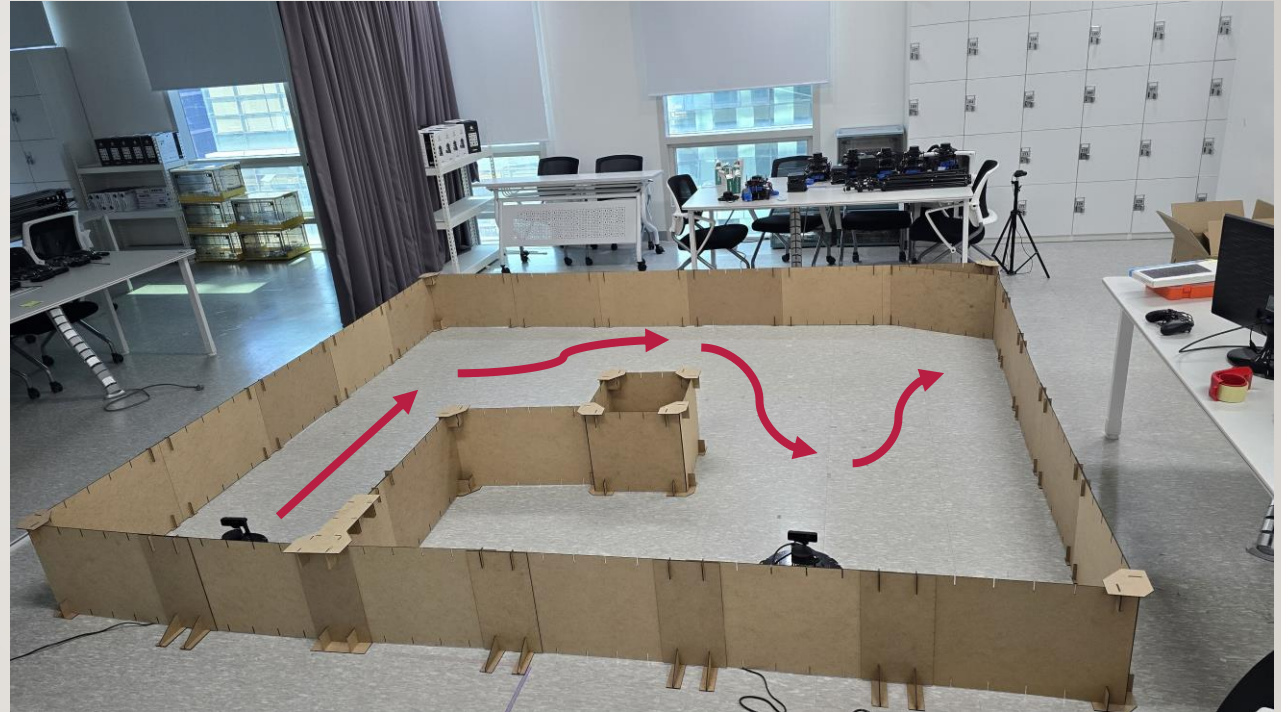
DETECTION ALERT



START



NAVIGATE



TRACK & FOLLOW



DAY I

- Welcome
- Project Introduction
- Introduction to Project Development Process
- Business Requirement Development
- System Requirement Development
- Time Management
- System(High Level) Design (Intro.)

DAY 2 (MINI PROJECT)

- Yolo객체 인식 모델 활용과 성능 평가 방법 이해
 - Custom Dataset과 Fine Tuning으로 자체 객체 인식 모델 구현 및 평가
 - (Optional)경량화 모델 등 개별 요구사항에 적합한 모델 탐색 및 성능 검증

DAY 2 (MINI PROJECT)

WEB-CAM 기반 객체 인식

(IF NEEDED)

- YOLOv8 기반 데이터 수집/학습/deploy (Detection Alert)
 - 감시용 데이터 수집(rc_car, dummy, 등)
 - 감시용 데이터 라벨링
 - YOLOv8 기반 학습
 - YOLOv8 Object Detection

AMR-CAM 기반 객체 인식

- AMR(Autonomous Mobile Robot) Turtlebot4 개발 환경 구축
- 로봇 개발 환경에 완성 모델 서빙 및 테스트 / 로봇 H/W, 제반 환경의 한계점 도출
 - Tracking 데이터 수집((rc_car, dummy, 등)
 - Tracking 데이터 라벨링
 - YOLOv8 기반 학습
 - YOLOv8 Object **Tracking**

DAY 3 (MINI PROJECT)

- Auto. Driving 시스템 학습
 - Digital Mapping of environment
 - Operate AMR (Sim. & Real)
 - Tutorial 실행
 - Detection, Depth and AMR 주행
 - 로봇 개발 환경에 적용 및 테스트 / 로봇 H/W, 제반 환경의 한계점 도출

TURTLEBOT4 시뮬레이션

- 환경 구축
- SLAM과 AutoSLAM으로 맵 생성
- Sim.Tutorial 실행
- Detection, Depth and AMR 주행 example

DAY 3 (MINI PROJECT)

REAL ROBOT

- Manually operating the AMR (Teleops)
- autonomous driving 시스템 with obstacle avoidance
 - Digital Mapping of environment
 - Launching Localization, Nav2, and using Rviz to operate a robot
 - Goal Setting and Obstacle Avoidance using Navigation

TUTORIAL

- Turtlebot4 API를 활용한 Initial Pose Navigate_to Pose 구현
- Turtlebot4 API를 활용한 Navigate_Through_pose, Follow Waypoints 구현

HOW TO WORK TOGETHER

- Participate, Participate, Participate!!!
- No long emails or Kakaotalk, prefer face to face
- Be open to suggestions and idea
- Be proactive (적극적), take initiative (주도적)
- HOW is as important as WHAT

프로젝트 RULE

80/20 → 20/80

TEAMWORK AND PROJECT MANAGEMENT



BRAINSTORMING RULES

- Every input is good input
- Do not critique inputs only seek to understand
- Organize inputs into logical groupings
- Sequence or show relationships as needed
- Use Posted Notes on Flip Chart



프로젝트 RULE NUMBER ONE!!!

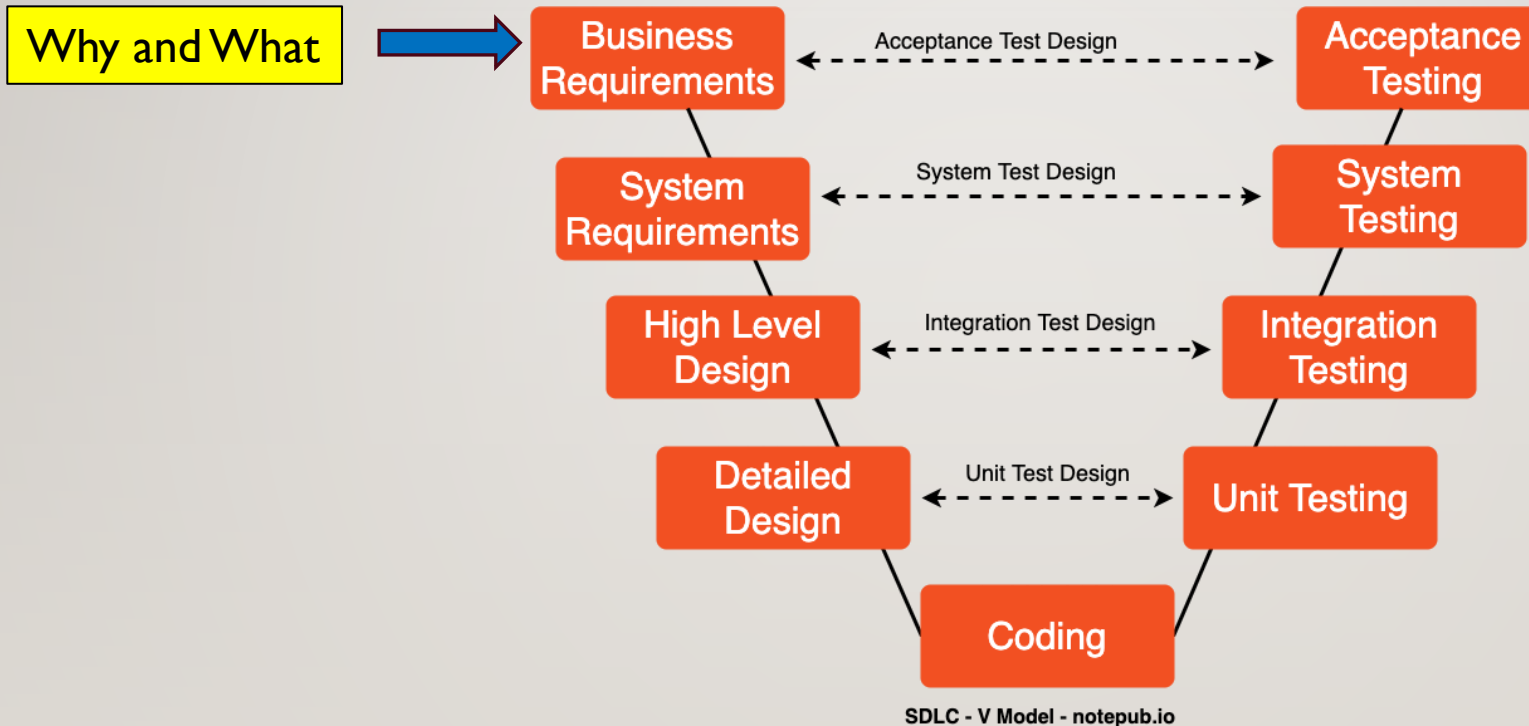
Have Fun Fun Fun!



PROJECT DEVELOPMENT
IS A PROCESS



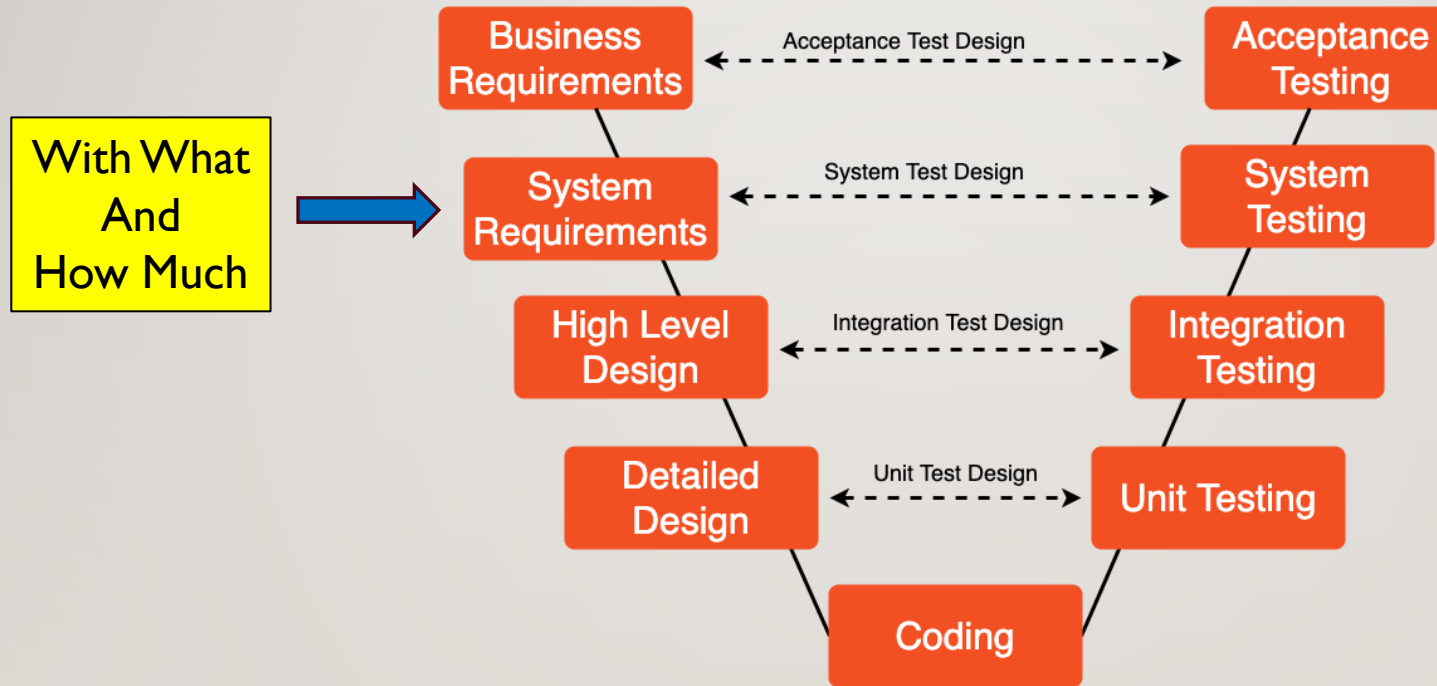
SW DEVELOPMENT PROCESS



TEAM EXERCISE I

Brainstorm Business Requirement for the project and write business requirement statement

SW DEVELOPMENT PROCESS



SDLC - V Model - notepub.io

TEAM EXERCISE 2

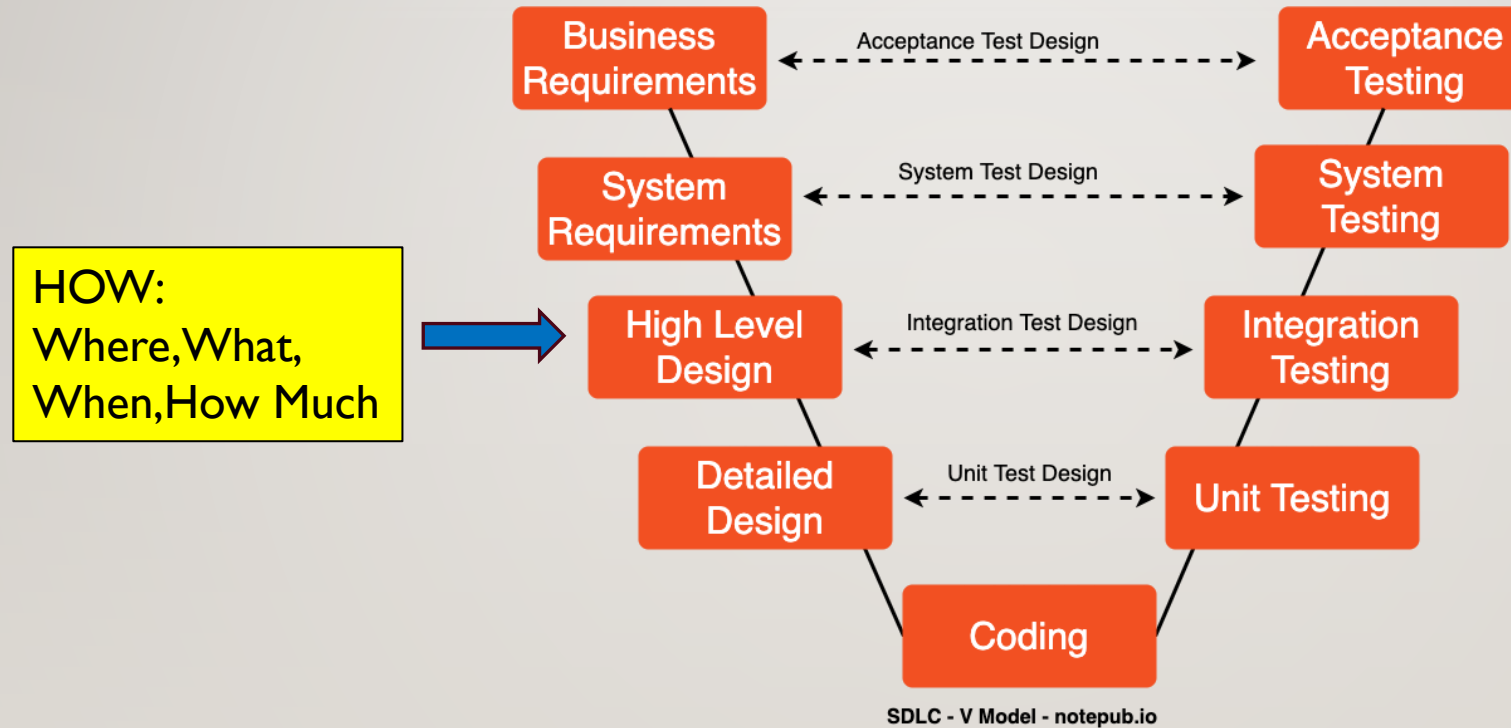
Brainstorm **Updated** System Requirement for the project and document

Using the posted notes and flipchart as needed

SYSTEM REQUIREMENT PRESENTATION BY EACH TEAM

Using the posted notes and flipchart as needed

SW DEVELOPMENT PROCESS



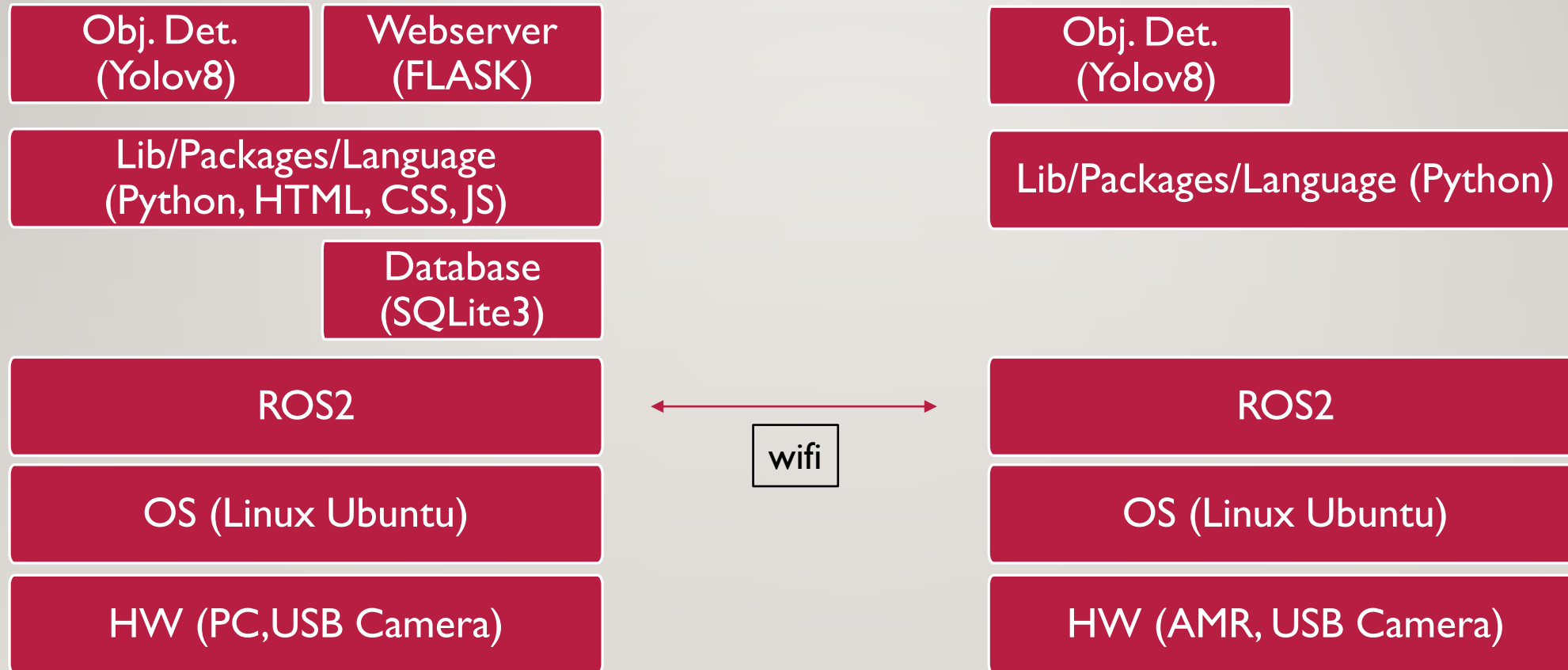
DAY 2



SYSTEM AND DEVELOPMENT ENVIRONMENT SETUP



PROJECT SW STACK



USEFUL COMMANDS

\$ lsb_release -a

- Linux distribution info

\$ echo \$ROS_DISTRO

- ROS: Humble

\$ code --version

- Vscode

\$ python3 --version

- Python

\$ sudo apt update

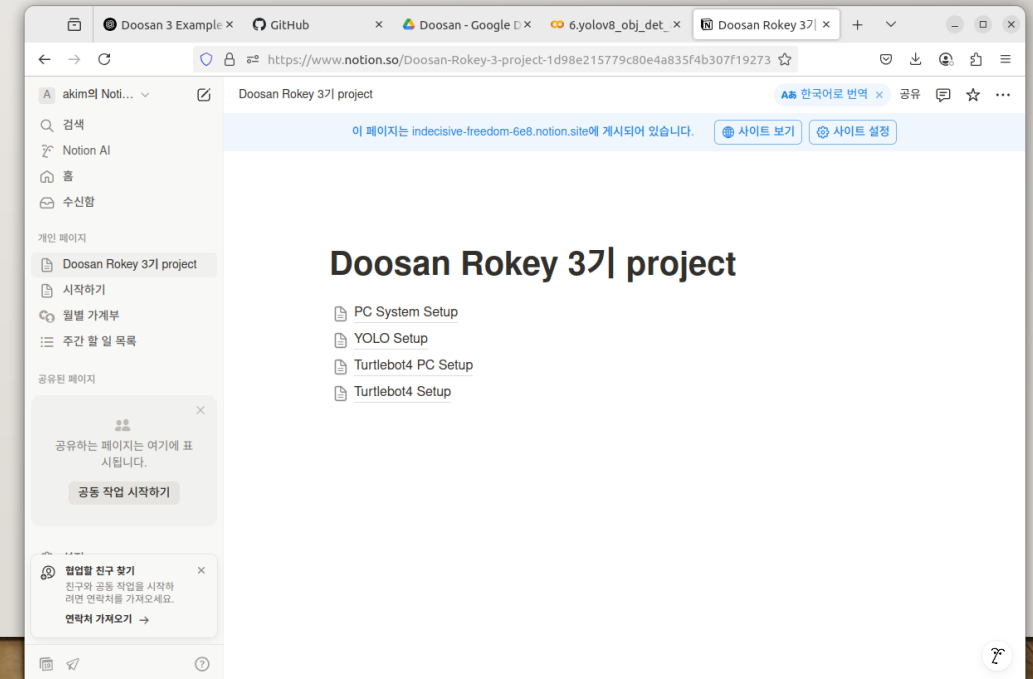
\$ sudo apt upgrade

\$ python -m ensurepip --upgrade

- Assumes Linux (Ubuntu 22.04), ROS Humble, VScode, and Python are already installed globally

SYSTEM ENVIRONMENT SETUP SHELL SCRIPT

- PC System Setup
- <https://indecisive-freedom-6e8.notion.site/PC-System-Setup-1d98e215779c806080bbd1014d63a406>



ROS2 DEVELOPMENT WORKSPACE

```
$ cat ~/.bashrc
```

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

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

```
$ echo "source /usr/share/colcon_argcomplete/hook/colcon_argcomplete.bash" >> ~/.bashrc  
#check path
```

```
$ source ~/.bashrc
```

PREPARING FOR YOLO LABELLING

Data Labelling : use previously installed
Labellmg









Or

[YOLO Setup](#)

<https://indecisive-freedom-6e8.notion.site/YOLO-Setup-1d98e215779c80f389eefbe0d86ee0ec>

\$ labellmg

Doosan Rokey 371 project









-  [PC System Setup](#)
-  [YOLO Setup](#) 
-  [Turtlebot4 Setup](#)
-  [Turtlebot4 PC Setup](#)
-  [Single Robot Setup](#)
-  [Multi Robot Setup](#)
-  [Transform Explained](#)

HOW TO SETUP AMR

Turtlebot4 Setup

<https://indecisive-freedom-6e8.notion.site/Turtlebot4-Setup-1d98e215779c801b8e74eff67eb6305e>

Doosan Rokey 371 project









-  [PC System Setup](#)
-  [YOLO Setup](#)
-  [Turtlebot4 Setup](#) 
-  [Turtlebot4 PC Setup](#)
-  [Single Robot Setup](#)
-  [Multi Robot Setup](#)
-  [Transform Explained](#)

SETUP PC FOR AMR

Turtlebot4 PC Setup

<https://indecisive-freedom-6e8.notion.site/Turtlebot4-PC-Setup-1d98e215779c80e887b6c1dff4f151be>

Doosan Rokey 371 project

-  [PC System Setup](#)
-  [YOLO Setup](#)
-  [Turtlebot4 Setup](#)
-  [Turtlebot4 PC Setup](#) 
-  [Single Robot Setup](#)
-  [Multi Robot Setup](#)
-  [Transform Explained](#)

SETUP BASH

Add in ~/.bashrc:









- `source ~/turtlebot4_ws/install/setup.bash`
- `ROS_DOMAIN_ID = 0`

`$ source ~/.bashrc`

SETUP PC FOR AMR

- [Single Robot Setup](#)
- <https://indecisive-freedom-6e8.notion.site/Single-Robot-Setup-1f98e215779c805fa521dd07277a2880>

Doosan Rokey 371 project

-  [PC System Setup](#)
-  [YOLO Setup](#)
-  [Turtlebot4 Setup](#)
-  [Turtlebot4 PC Setup](#)
-  [Single Robot Setup](#) 
-  [Multi Robot Setup](#)
-  [Transform Explained](#)

AMR (DEMO)

- Power On and Off
- Connecting to Robot
 - Wifi Router
 - SSH
- Docking and Undocking (using pannel/buttons)
- Teleop with keyboard



CONNECTING TO AMR -- SSH

Connect your PC to WiFi router that your AMR is connected

Ex: turtle09

Obtain the ip address shown on the OLED display of the Turtlebot4

EX: 192.168.10.16

Open a terminal window

```
$ dpkg -l | grep openssh
```

If not installed...

```
$ sudo apt install openssh-server -y
```

Connect to AMR via SSH

```
$ ssh ubuntu@192.168.10.16
```

PW: rokey1234

Do **NOT INSTALL** any packages to AMR **WITHOUT** speaking with me first!!!

UNDOCK/DOCK AMR (ON PC OR AMR)

UNDOCK

\$ ros2 topic list

Check the list

\$ ros2 action send_goal
/robot<n>/undock
irobot_create_msgs/action/Undock
“{”

DOCK

\$ ros2 topic list

Check the list

\$ ros2 action send_goal /robot<n>/dock
irobot_create_msgs/action/Dock “{”

HOW TO TEST PC – AMR CONNECTION

PC TERMINAL

\$ ros2 topic list (will need to execute it twice)

#Check the list

\$ ros2 run teleop_twist_keyboard
teleop_twist_keyboard --ros-args -r
/cmd_vel:=/robot<n>/cmd_vel

\$ ro2 topic echo /robot<n>/oakd/rgb/image_raw
--once

AMR TERMINAL

\$ ros2 topic list (will need to execute it twice)

#Check the list

- Undock your robot

\$ ro2 topic echo /robot<n>/oakd/rgb/image_raw
--once

USING ROS_WS

ROS2 DEVELOPMENT WORKSPACE

CREATE WORKSPACE

```
$ mkdir -p  
  ~/rokey3_<grp_letter><grp_num>_ws/src  
  • (i.e. mkdir -p ~/rokey3_A2_ws/src)
```

```
$ cd ~/rokey3_A2_ws
```

```
$ rosdep install --from-paths src --ignore-src  
-r -y
```

If not installed...

```
$ sudo rosdep init
```

```
$ rosdep update
```

*NOT CREATED UNTIL COLCON

```
workspace/      # Root of the workspace  
├── src/         # Source code (ROS packages)  
├── build/       # Build files (generated by colcon)  
├── install/     # Installed packages and setup scripts  
└── log/         # Build logs
```

```
$ colcon build
```

```
$ source install/setup.bash
```

ROS2 DEVELOPMENT WORKSPACE

```
$ cd ~/rokey3_A2_ws/src
```

```
$ ros2 pkg create --build-type  
ament_python <my_package>
```

```
my_package/  
├─ package.xml          # Package metadata and dependencies  
├─ setup.py             # Build instructions for Python packages  
├─ setup.cfg            # Optional, configures metadata for setuptools  
├─ launch/              # Launch files for starting nodes (optional)  
├─ config/              # Configuration files (optional)  
├─ resource/            # Empty file matching package name for ament index  
├─ my_package/          # Python package directory (contains code)  
│   └─ __init__.py      # Makes this directory a Python package  
│   └─ my_node.py       # Example Python node  
└─ msg/                 # Message definitions (optional)
```

ROS2 DEVELOPMENT WORKSPACE

Write your code below the `my_package/` directory under `my_package/ package` directory

```
my_package/
├─ package.xml          # Package metadata and dependencies
├─ setup.py             # Build instructions for Python packages
├─ setup.cfg            # Optional, configures metadata for setuptools
├─ launch/              # Launch files for starting nodes (optional)
├─ config/              # Configuration files (optional)
├─ resource/            # Empty file matching package name forament inc
├─ my_package/          # Python package directory (contains code)
│   └─ __init__.py      # Makes this directory a Python package
│   └─ my_node.py       # Example Python node
└─ msg/                 # Message definitions (optional)
```

ROS2 DEVELOPMENT WORKSPACE

```
$ cd ~/ rokey3_A2_ws
```

```
$ colcon build
```

```
workspace/      # Root of the workspace
├─ src/         # Source code (ROS packages)
├─ build/       # Build files (generated by colcon)
├─ install/     # Installed packages and setup scripts
└─ log/         # Build logs
```

```
$ echo "source ~/ rokey3_A2_ws/install/setup.bash" >> ~/.bashrc #check path
```

```
$ source ~/.bashrc
```


ROS HINTS

- 🔗 2_0_a_image_publisher.py
- 🔗 2_0_b_image_subscriber.py
- 🔗 2_0_c_data_publisher.py
- 🔗 2_0_d_data_subscriber.py

ROS HINTS

- Edit setup.py under <package_name> directory add entry for each node

```
entry_points={ 'console_scripts':  
[ '<command_name> =  
<package_name>.<code_filename>:main', },
```

<command_name> is used when ros2 run is executed i.e. data_publisher

```
tests_require=['pytest'],  
entry_points={  
    'console_scripts': [  
        'pub_image = day2.2_0_a_image_publisher:main',  
        'show_image = day2.2_0_b_image_subscriber:main',  
        'pub_data = day2.2_0_c_data_publisher:main',  
        'show_data = day2.2_0_d_data_subscriber:main',
```

ROS HINTS

\$ cd ~/rokey3_A2_ws

\$ colcon build

\$ source

~/rokey1_A2_ws/install/setup.bash

\$ ros2 run <package_name>

<command_name>

\$ sudo apt update

\$ sudo apt install terminator



ubuntu@ubuntu: ~ 100x12

rokey-kim:~\$ ros2 run day2 pub_image

ROS HINTS

```
2_0_a_image_publisher.py
2_0_b_image_subscriber.py
2_0_c_data_publisher.py
2_0_d_data_subscriber.py
```

```
$ ros2 run rqt_graph rqt_graph
```

```
$ ros2 node list
```

```
$ ros2 node info <node_name>
```

```
$ ros2 topic list
```

```
$ ros2 topic info <topic_name>
```

```
$ ros2 topic echo /chatter
```

```
$ ros2 interface list
```

```
$ ros2 interface show
<package_name>/msg/<MessageName>
```


KEY SUBSYSTEM (MODULES) TO DEVELOP

- Detection Alert

- Camera Capture
- Object Detection
- Send messages to other subsystems

- AMR Controller


- Receive messages and act accordingly
- Move using (SLAM) with Obstruction avoidance
- Target Acquisition (Obj. Det.) and Tracking
- Follow target using camera and motor control

PERFORM DATA COLLECTION FOR DETECTION ALERT



COLLECTION IMAGES FROM WEBCAM

- Image Capture (WEBCAM)



```
2_0_a_image_publisher.py
2_0_b_image_subscriber.py
2_0_c_data_publisher.py
2_0_d_data_subscriber.py
2_1_a_capture_wc_image.py
2_1_b_cont_capture_wc_image.py
2_1_c_capture_wc_thread.py
```

COLLECTION IMAGES FROM AMR CAMERA



AMR INTRODUCTION

- [User Manual · Turtlebot4 User Manual](#)
- <https://turtlebot.github.io/turtlebot4-user-manual/>





UNDOCK/DOCK AMR

UNDOCK

\$ ros2 topic list

Check the list

\$ ros2 action send_goal
/robot<n>/undock
irobot_create_msgs/action/Undock
“{”

DOCK

\$ ros2 topic list

Check the list

\$ ros2 action send_goal /robot<n>/dock
irobot_create_msgs/action/Dock “{”

WHICH IMAGE TOPIC TO USE?

- /oakd/rgb/preview/image_raw
 - /oakd/rgb/image_raw
 - /oakd/rgb/image_raw/compressed
 - /oakd/stereo/image_raw
 - ...
-
- *** not all of the topics are visible
- EXERCISE
 - Create a script to display and compare

HOW TO MOVE FILE FROM PC MOVE MAP TO AMR

PC TERMINAL

Connect to turtlebot via ssh first

```
$ scp <dir_path>/<file_name>  
ubuntu@<rokey IP>:/home/ubuntu  
$ EX: scp oakd_pro.yaml  
ubuntu@172.30.1.1:/home/ubuntu/
```

A terminal window with a dark background. It shows a file transfer progress bar for 'my_best.pt' which is nearly full. Below it, a file named 'oakd_pro_new.yaml' is listed with a small icon to its left.

my_best.pt

oakd_pro_new.yaml

AMR TERMINAL

```
$ cd ~ #go to home  
  
$ ls oakd_pro_new.yaml #check if the file  
transferred correctly  
  
$ sudo mv oakd_pro_new.yaml  
/opt/ros/humble/share/turtlebot4_bringup/config/  
  
$ ls /opt/ros/humble/share/turtlebot4_bringup/config/
```

UPDATING THE OAKD CONFIG (ROBOT)

ON TURTLEBOT4:

```
$ cd
/opt/ros/humble/share/turtlebot4_bringup/co
nfig

$ sudo cp oakd_pro.yaml oakd_pro_orig.yaml

$ sudo cp oakd_pro_new.yaml oakd_pro.yaml

$ sudo reboot
```

```
1 /oakd:
2   ros_parameters:
3     use_sim_time: false
4
5   camera:
6     i_enable_imu: false
7     i_enable_ir: false
8     i_floodlight_brightness: 0
9     i_laser_dot_brightness: 100
10    i_nn_type: none
11    i_pipeline_type: RGBD # RGB + Depth
12    i_usb_speed: SUPER_PLUS
13
14    rgb:
15      i_board_socket_id: 0
16      i_width: 640
17      i_height: 480
18      i_fps: 30.0
19      i_enable_preview: true
20      i_interleaved: false
21      i_low_bandwidth: true
22      i_publish_topic: true
23      i_resolution: 480P # sets 640x480 internally
24
25
26    stereo: # [x] Required to enable depth
27      i_board_socket_id: 1
28      i_fps: 30.0
```

UPDATING THE OAKD CONFIG (ROBOT)

ON TURTLEBOT4:

```
$ cd  
  /opt/ros/humble/share/turtlebot4_bringup/co  
  nfig  
  
$ sudo cp oakd_pro.yaml oakd_pro_orig.yaml  
  
$ sudo cp oakd_pro_new.yaml oakd_pro.yaml  
  
$ sudo reboot
```

IF NEEDED

```
$ sudo systemctl status turtlebot4.service  
  
$ sudo systemctl restart turtlebot4.service  
  
$ ros2 topic list
```


DIMENSIONS AND RESOLUTION

Supported `i_resolution` values (RGB):

Resolution Keyword	Width × Height	Notes
1080P	1920 × 1080	Default, high-res
720P	1280 × 720	Medium-res
800P	1280 × 800	Slightly taller
480P	640 × 480	✓ Ideal for alignment with stereo
400P	640 × 400	Wide, cropped top/bottom
320P	640 × 360	Lower-res
240P	320 × 240	Very low-res, fast

CODING HINTS

- Image Capture (AMR)



```
2_0_a_image_publisher.py
2_0_b_image_subscriber.py
2_0_c_data_publisher.py
2_0_d_data_subscriber.py
2_1_a_capture_wc_image.py
2_1_b_cont_capture_wc_image.py
2_1_c_capture_wc_thread.py
2_1_d_capture_image.py
2_1_e_cont_capture_image.py
```

CODING HINTS

- Image Capture

```
2_1_a_capture_wc_image.py
2_1_b_cont_capture_wc_image.py
2_1_c_capture_wc_thread.py
2_1_d_capture_image.py
2_1_e_cont_capture_image.py
```

- Data Labelling
 - labellmg

PREPARING FOR YOLO LABELLING

Data Labelling : use previously installed
Labellmg









Or

[YOLO Setup](#)

<https://indecisive-freedom-6e8.notion.site/YOLO-Setup-1d98e215779c80f389eefbe0d86ee0ec>

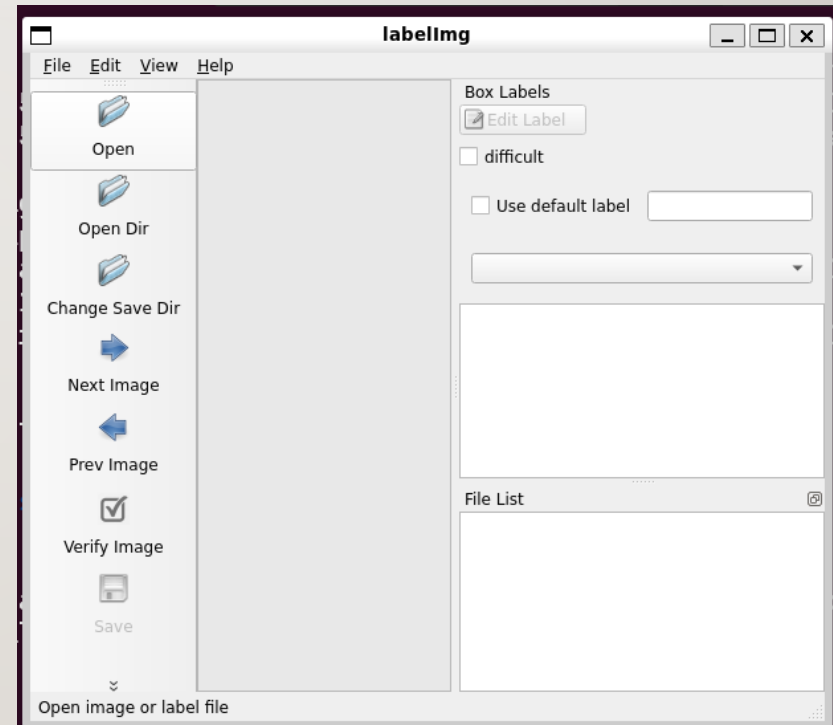
\$ labellmg

Doosan Rokey 371 project

-  [PC System Setup](#)
-  [YOLO Setup](#) 
-  [Turtlebot4 Setup](#)
-  [Turtlebot4 PC Setup](#)
-  [Single Robot Setup](#)
-  [Multi Robot Setup](#)
-  [Transform Explained](#)

CODING HINTS

- Image Capture
- Data Labelling



CODING HINTS

- Data Labelling : Labellmg

라벨링 순서


1. 이미지파일 불러오기 (Open Dir)
2. 저장형식 변경 (PascalVOC, YOLO)
3. 이미지 선택
4. 바운딩 박스 그리기(create rectbox)
5. Class 지정
6. 저장경로 생성 및 변경(Change Save Dir)
7. 저장(Save)

단축키

Ctrl + u	Load all of the images from a directory
Ctrl + r	Change the default annotation target dir
Ctrl + s	Save
Ctrl + d	Copy the current label and rect box
Ctrl + Shift + d	Delete the current image
Space	Flag the current image as verified
w	Create a rect box
d	Next image
a	Previous image
del	Delete the selected rect box
Ctrl++	Zoom in
Ctrl--	Zoom out
↑→↓←	Keyboard arrows to move selected rect box

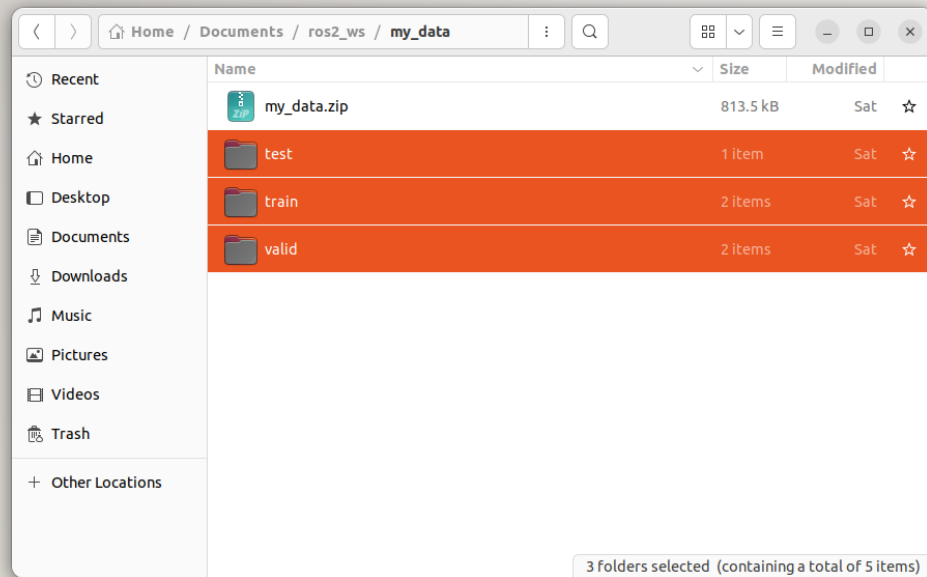
CODING HINTS

- Image Capture
- Data Labelling
- Data Preprocessing



```
2_1_a_capture_wc_image.py
2_1_b_cont_capture_wc_image.py
2_1_c_capture_wc_thread.py
2_1_d_capture_image.py
2_1_e_cont_capture_image.py
2_3_a_create_data_dirs.py
2_3_b_move_image.py
2_3_c_move_labels.py
```

ZIP TRAIN DATA SET



PERFORM YOLO TRAINING & INFERENCE




CODING HINTS

- Image Capture
- Data Labelling
- Preprocessing
- Yolo8 Object Det (Training)



 2_3_a_create_data_dirs.py

 2_3_b_move_image.py

 2_3_c_move_labels.py

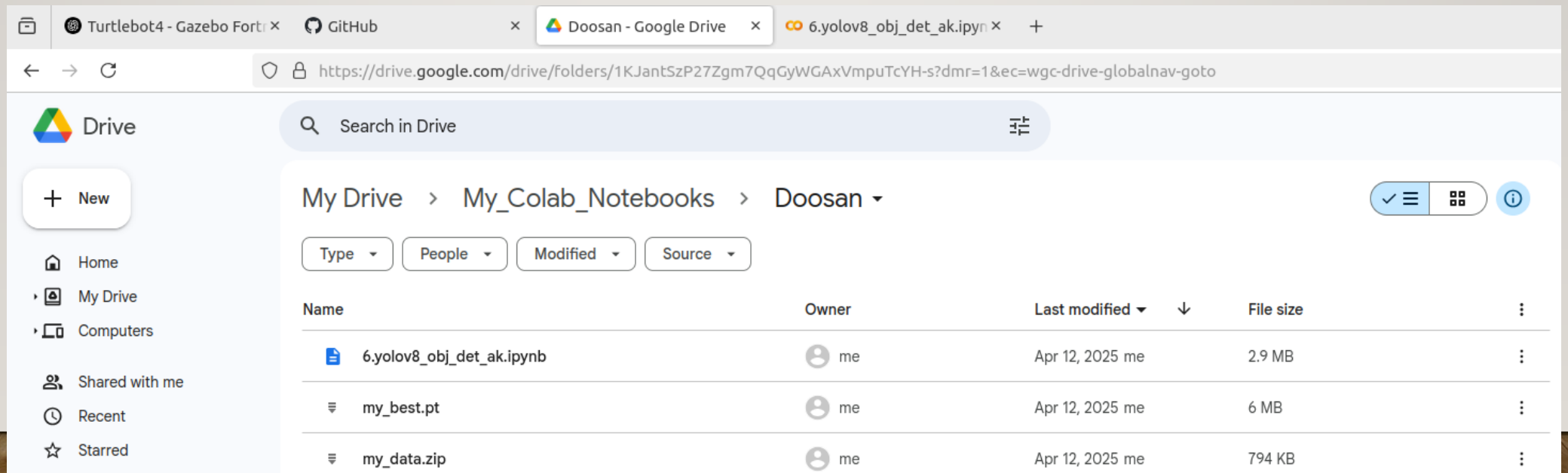
 2_4_a_yolov8_obj_det_ak.ipynb

 2_4_b_gpu_test.py

 2_4_c_compare_yolo.py

USING GOOGLE COLLAB.TO CREATE CUSTOM MODEL

- Move the files to google drive
 - my_data.zip
 - yolov8.obj.det.ak.ipynb

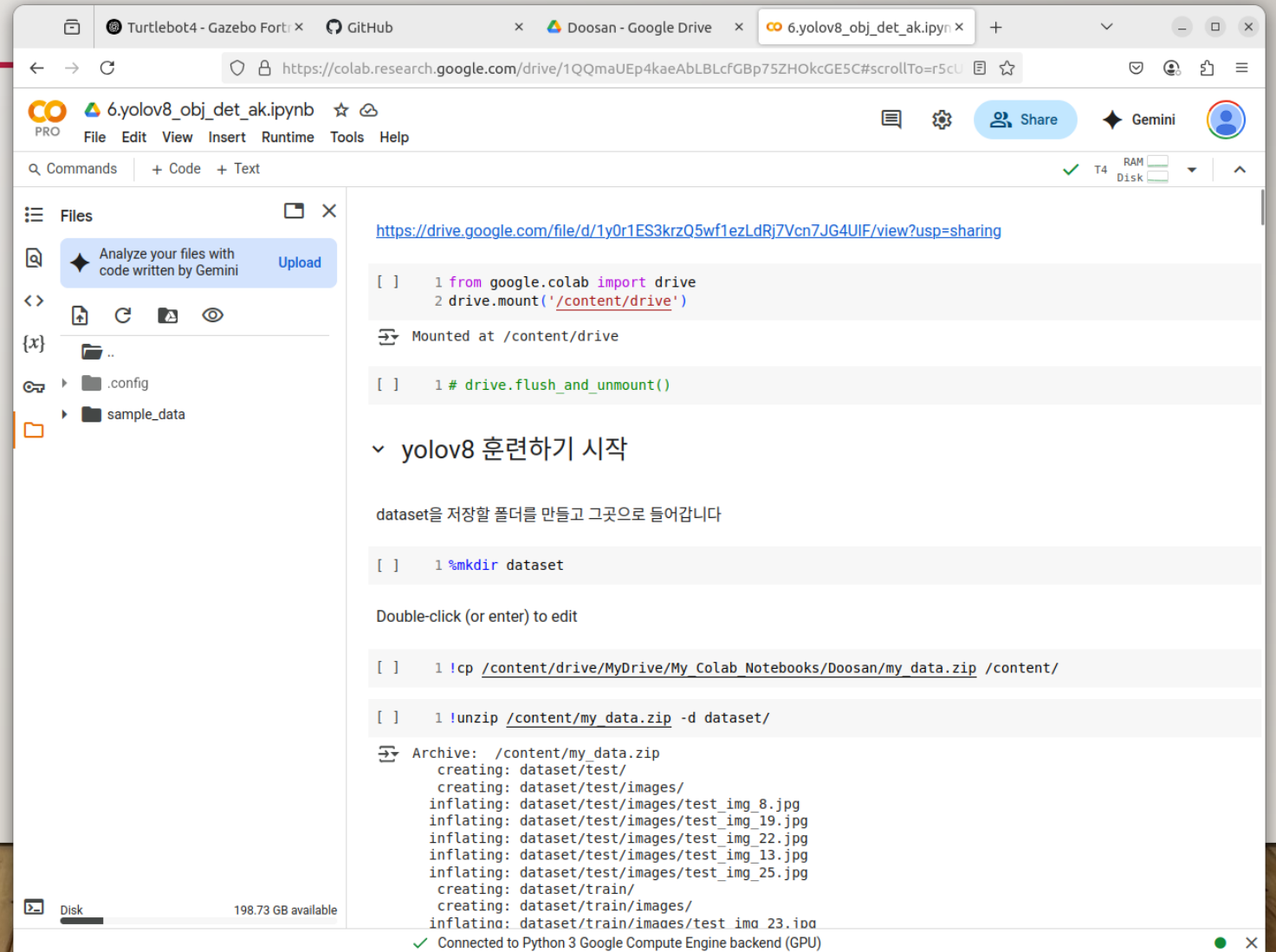


The screenshot shows a web browser with multiple tabs. The active tab is 'Doosan - Google Drive'. The address bar shows the URL: <https://drive.google.com/drive/folders/1KJantSzP27Zgm7QqGyWGAXVmpuTcYH-s?dmr=1&ec=wgc-drive-globalnav-goto>. The Google Drive interface is displayed, showing a sidebar with navigation options: Home, My Drive, Computers, Shared with me, Recent, and Starred. The main area shows the 'My Drive' view, specifically the 'My_Colab_Notebooks' folder, and then the 'Doosan' sub-folder. A search bar is at the top. Below the folder path, there are filters for Type, People, Modified, and Source. A table lists the files in the folder:

Name	Owner	Last modified	File size
6.yolov8_obj_det_ak.ipynb	me	Apr 12, 2025 me	2.9 MB
my_best.pt	me	Apr 12, 2025 me	6 MB
my_data.zip	me	Apr 12, 2025 me	794 KB

USING GOOGLE COLLAB.TO CREATE CUSTOM MODEL

- Move the training script to google collab. and execute line by line



```
from google.colab import drive
drive.mount('/content/drive')

Mounted at /content/drive

# drive.flush_and_unmount()

yolov8 훈련하기 시작

dataset을 저장할 폴더를 만들고 그곳으로 들어갑니다

!mkdir dataset

Double-click (or enter) to edit

!cp /content/drive/MyDrive/My_Colab_Notebooks/Doosan/my_data.zip /content/

!unzip /content/my_data.zip -d dataset/

Archive: /content/my_data.zip
creating: dataset/test/
creating: dataset/test/images/
inflating: dataset/test/images/test_img_8.jpg
inflating: dataset/test/images/test_img_19.jpg
inflating: dataset/test/images/test_img_22.jpg
inflating: dataset/test/images/test_img_13.jpg
inflating: dataset/test/images/test_img_25.jpg
creating: dataset/train/
creating: dataset/train/images/
inflating: dataset/train/images/test_img_23.jpg
```

Connected to Python 3 Google Compute Engine backend (GPU)

CODING HINTS

- Image Capture
- Data Labelling
- Preprocessing
- Yolo8 Object Det (Model)



```
2_3_a_create_data_dirs.py
2_3_b_move_image.py
2_3_c_move_labels.py
2_4_a_yolov8_obj_det_ak.ipynb
2_4_b_gpu_test.py
2_4_c_compare_yolo.py
```


CODING HINTS


- Image Capture
- Data Labelling
- Preprocessing
- Yolo8 Object Det (WEBCAM)



```
2_4_a_yolov8_obj_det_ak.ipynb
2_4_b_gpu_test.py
2_4_c_compare_yolo.py
2_4_d_yolov8_obj_det_wc.py
2_4_e_yolo_publisher_wc.py
2_4_f_yolo_subscriber_wc.py
2_4_g_yolov8_obj_det.py
2_4_h_yolov8_obj_det_thread.py
2_4_i_yolov8_obj_det_track.py
```

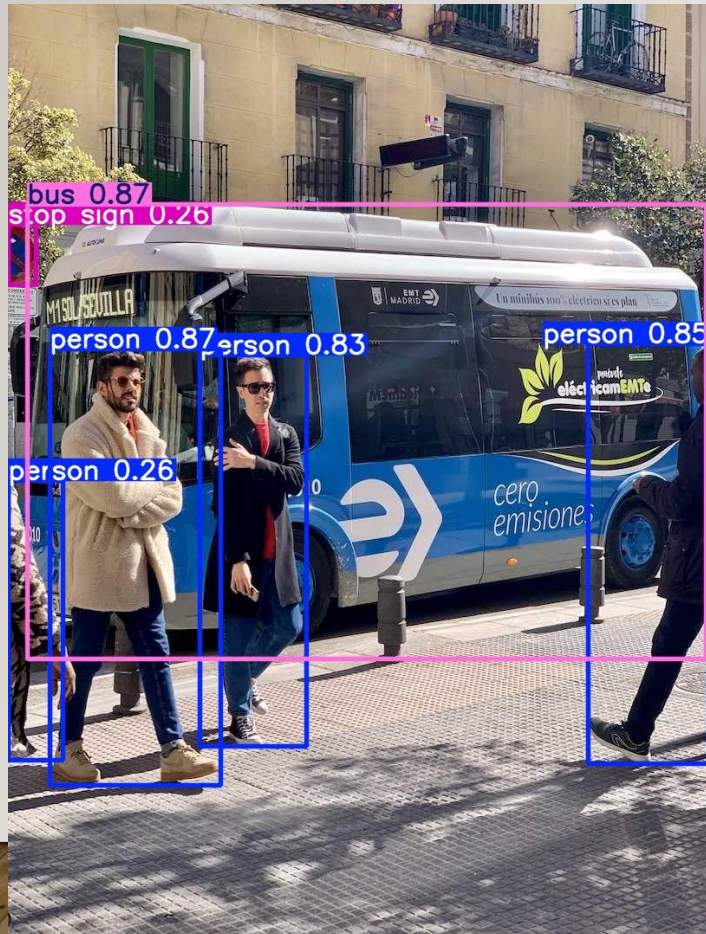
CODING HINTS

- Image Capture
- Data Labelling
- Preprocessing
- Yolo8 Object Det (AMR)

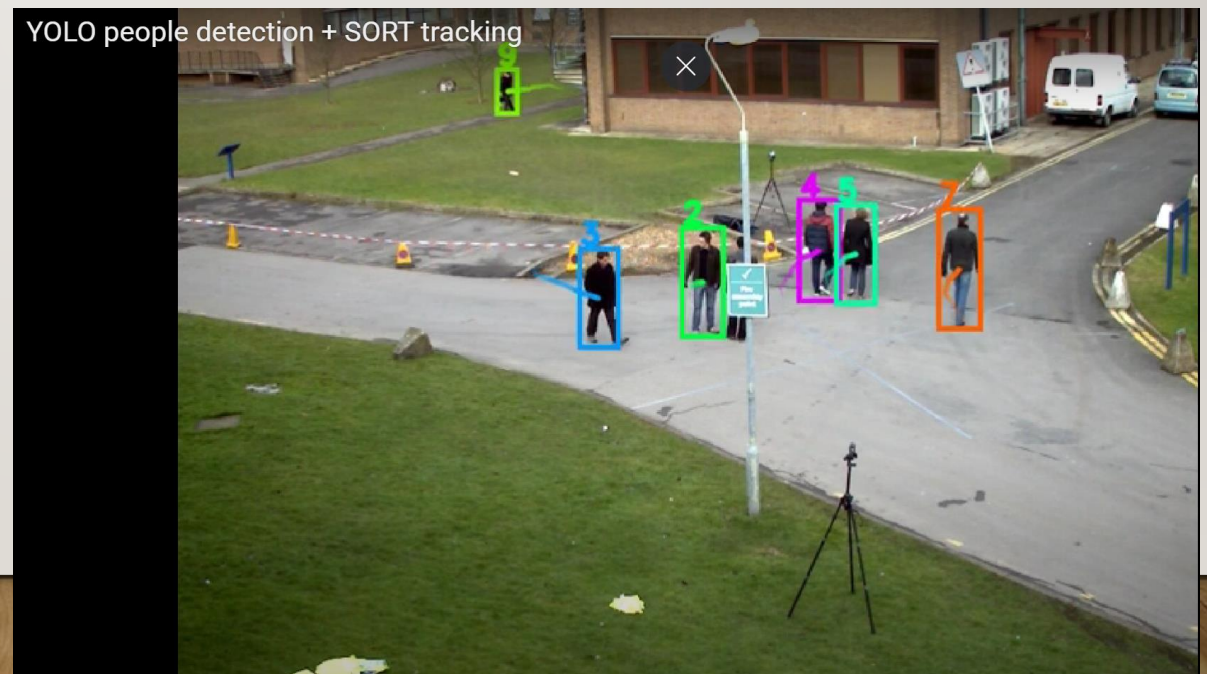


```
2_4_a_yolov8_obj_det_ak.ipynb
2_4_b_gpu_test.py
2_4_c_compare_yolo.py
2_4_d_yolov8_obj_det_wc.py
2_4_e_yolo_publisher_wc.py
2_4_f_yolo_subscriber_wc.py
2_4_g_yolov8_obj_det.py
2_4_h_yolov8_obj_det_thread.py
2_4_i_yolov8_obj_det_track.py
```

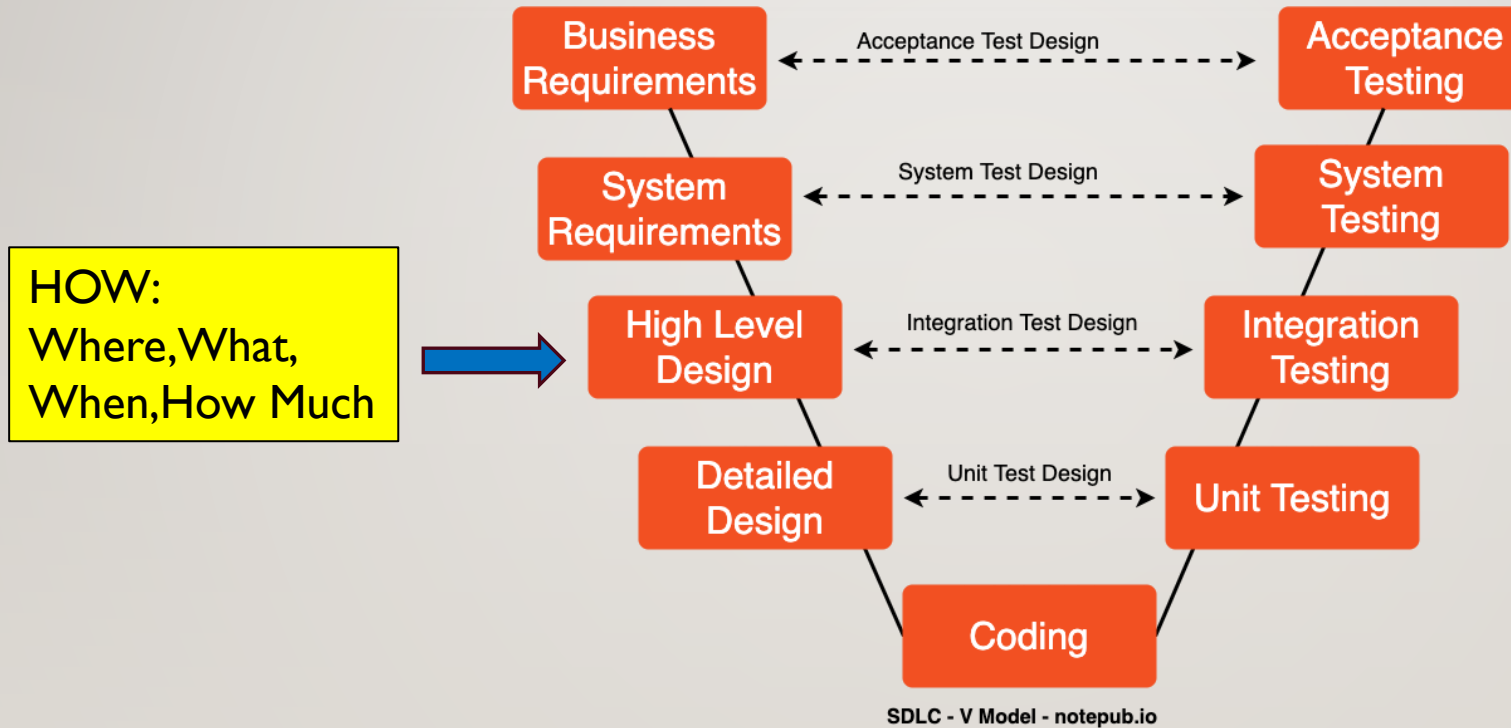
YOLO OBJ. DET. VS. YOLO TRACKING



- [Track - Ultralytics YOLO Docs](#)
 - [\(469\) YOLO people detection + SORT tracking – YouTube](#)
 - [Bing Videos](#)



SW DEVELOPMENT PROCESS



TEAM EXERCISE 3

Update System Design using Process Flow Diagram.

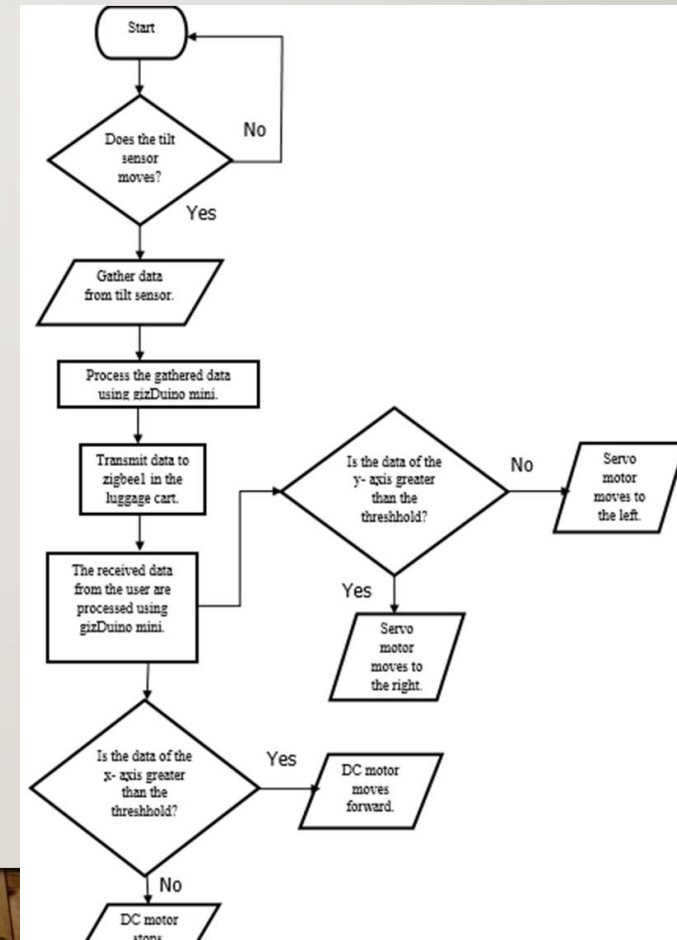
Use the posted notes and flipchart as needed

VISUALIZATION – SYSTEM FUNCTIONAL PROCESS FLOW DIAGRAMS

- To-Be Functional Process Flow Diagram

Detection Alert
AMR Controller

- **F**unctions
 - **I**nterfaces
 - **T**esting
- Dataflow
- Error and Exception Handling



프로젝트 RULE NUMBER ONE!!!

Are we still having
FUN!

