

GOOD MORNING!

早上好!

안녕하세요!

DAY 3



DAY I (DONE)

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

DAY 2 (DONE?)

- YOLOv8 기반 데이터 수집/학습/deploy (Detection Alert)
 - 감시용 데이터 수집(bus, truck, tank 등)
 - 감시용 데이터 라벨링
 - YOLOv8 기반 학습
 - YOLOv8 Object Detection
- Porting to ROS
 - Create Detection Alert Node
 - Generate Topics to send image and Obj. Det. results
 - Create Subscriber node and display image and print data from the Topic

DAY 3

- AMR (Autonomous Mobile Robot)기반 카메라 인식 autonomous driving 시스템 with obstacle avoidance 구축 (AMR Controller)
 - Digital Mapping of environment
 - Goal Setting and Obstacle Avoidance using Navigation
 - Object Tracking w/ AMR camera
 - Control logic between navigation/obj. tracking/ obj. following (teleop)
- Porting to ROS
 - Create AMR Controller Node
 - Create and send Obj.Tracking Image and data to Sysmon
- And finally, Integration and Test of Detection Alert & AMR Controller

DAY 4

- Flask 를 이용한 웹 서버 구축 (System Monitor)
 - Flask/HTML Intro
 - Deploy YOLOv8 Obj. Det results to web
 - Log in 기능 구현
 - Sysmon 웹기능 구현
 - 알람 기능 구현
- SQLite3를 이용한 데이터베이스 구축 및 연동 (System Monitor)
 - SQLite3 기본 기능 구현
 - DB 기능 구축
 - 알람이 울리는 경우 DB에 저장하는 기능 구현
 - 저장된 내용 검색하는 기능 구현

DAY 4

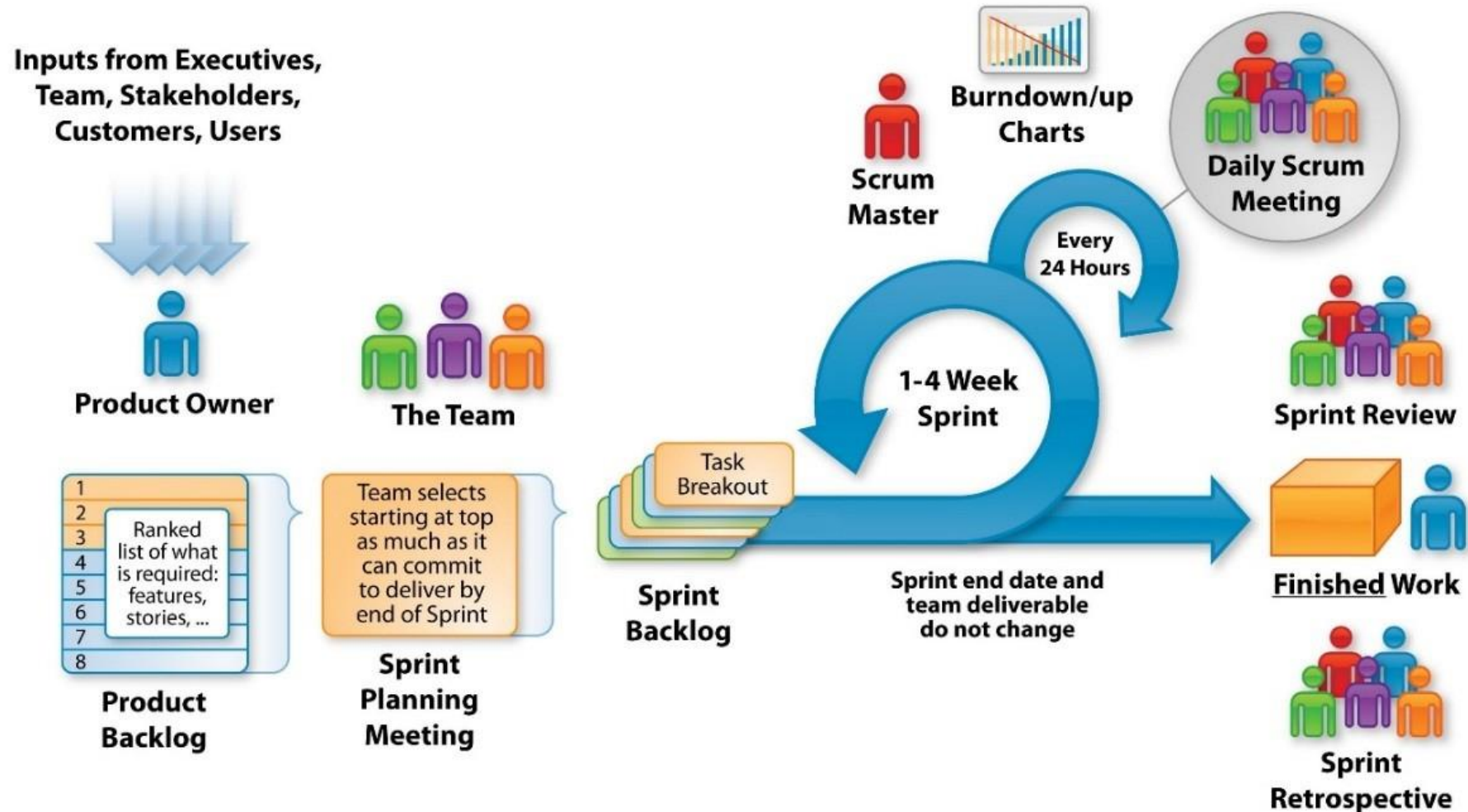
- Porting to ROS
 - Update Sysmon Node code
 - Update the database with received Obj. Det. Data from Detection Alert Node
 - Display the content of DB on System Monitor web page
- And finally, Integration and Test of Detection Alert, AMR Controller & System Monitor

프로젝트 RULE NUMBER ONE!!!

Have Fun Fun Fun!



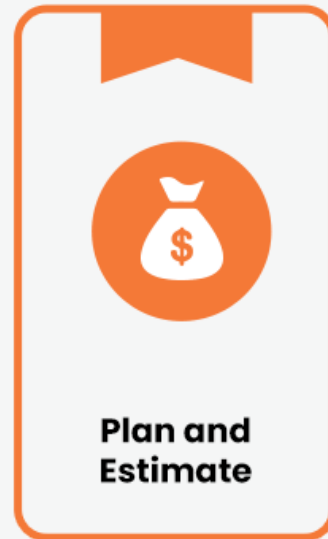
The Agile - Scrum Framework



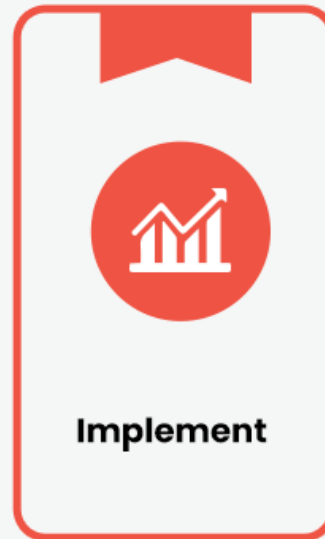
5 Stages of Scrum Sprint



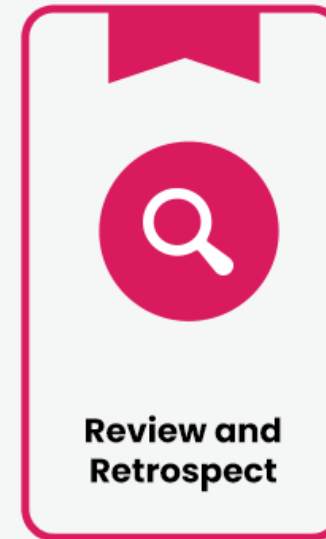
This phase includes the processes related to the commencement of a project, such as a scope and objectives, creating and distributing its charter, and taking other steps to guarantee success.



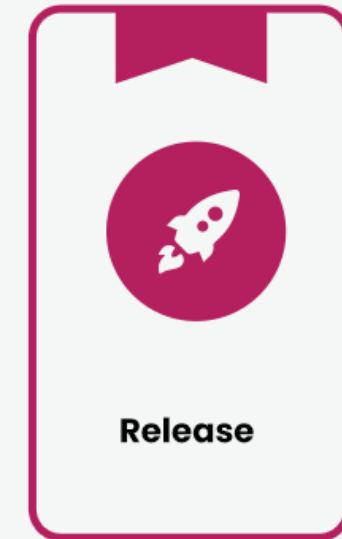
This phase involves planning and estimating processes, including creating user stories, approving, assessing, committing user stories, creating tasks, evaluating tasks, and creating a Sprint backlog.



This phase is about executing the tasks and activities to create a product. These activities include building the various outputs, conducting daily standup meetings, and grooming the product backlog.

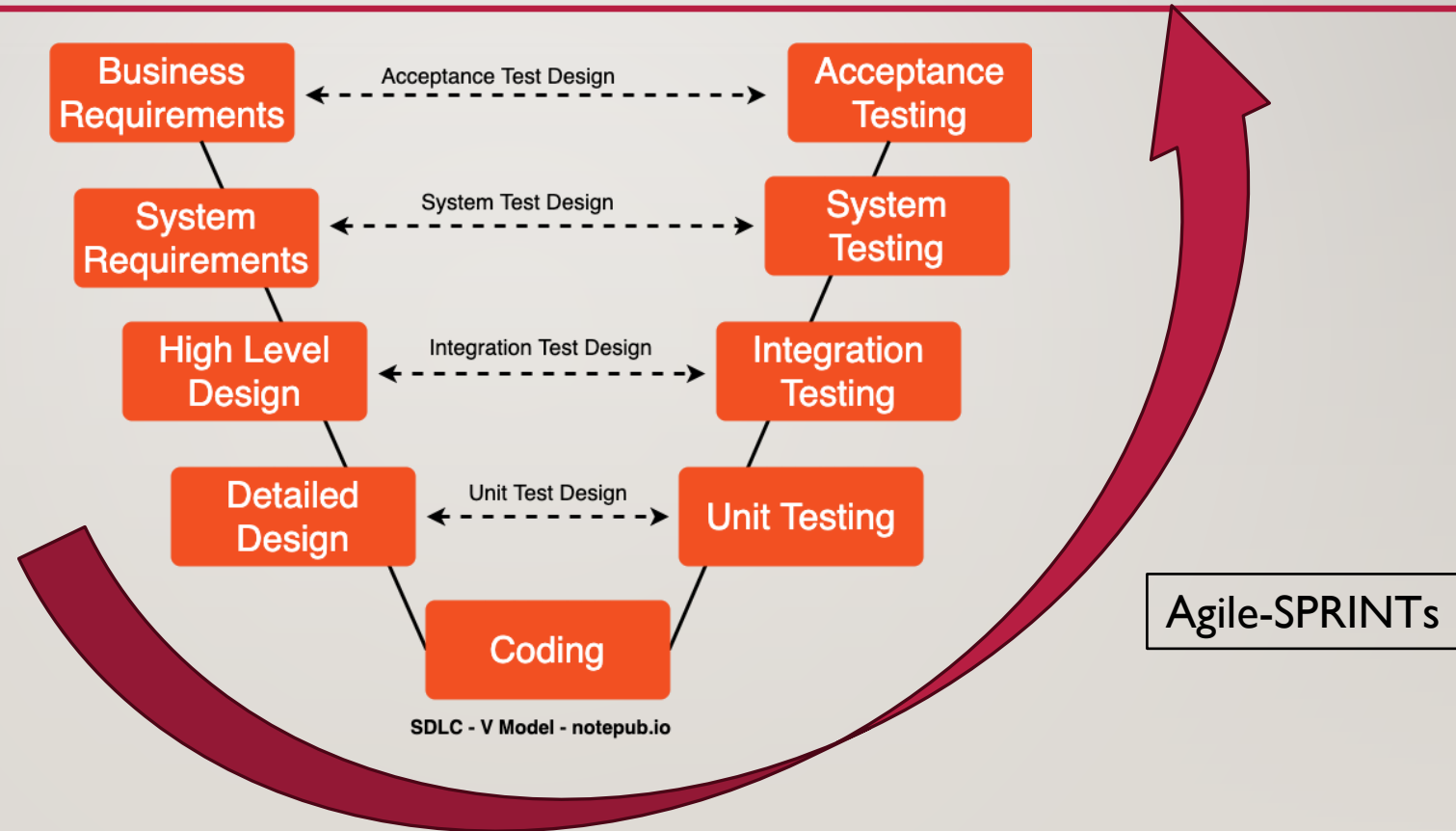


This stage of the project lifecycle is concerned with evaluating what has been accomplished so far, whether the team has worked to plan, and how it can do things better in the future.



This stage highlights delivering the accepted deliverables to the customer and determining, documenting, and absorbing the lessons learned during the project.

SW DEVELOPMENT PROCESS



PROJECT SPRINTS

- 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

- System Monitor

- Receive and Display Detection Camera and info
- Receive and Display AMR Camera and info
- Store, display, and report Information and Alerts

TEAM EXERCISE 5

Perform coding and testing of Detection Alert Module

RESULTS & CODE REVIEW BY EACH TEAM

Show actual results against the expected results and explain the code written

PROJECT SPRINTS

- Detection Alert

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

- AMR Controller

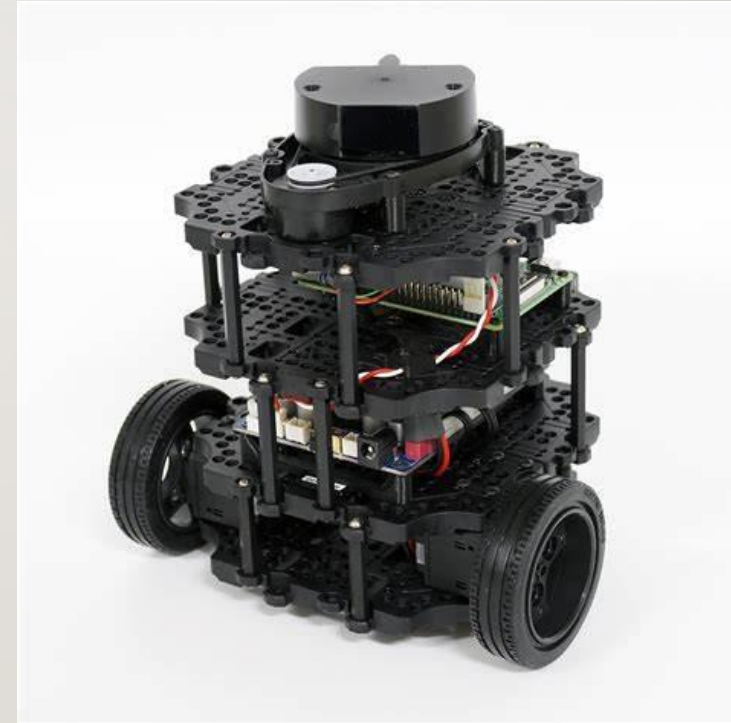
- Receive messages and act accordingly
- Move using (SLAM) with Obstruction avoidance
- Target Acquisition (Obj. Det.) and Tracking
- Follow target using camera and motor control

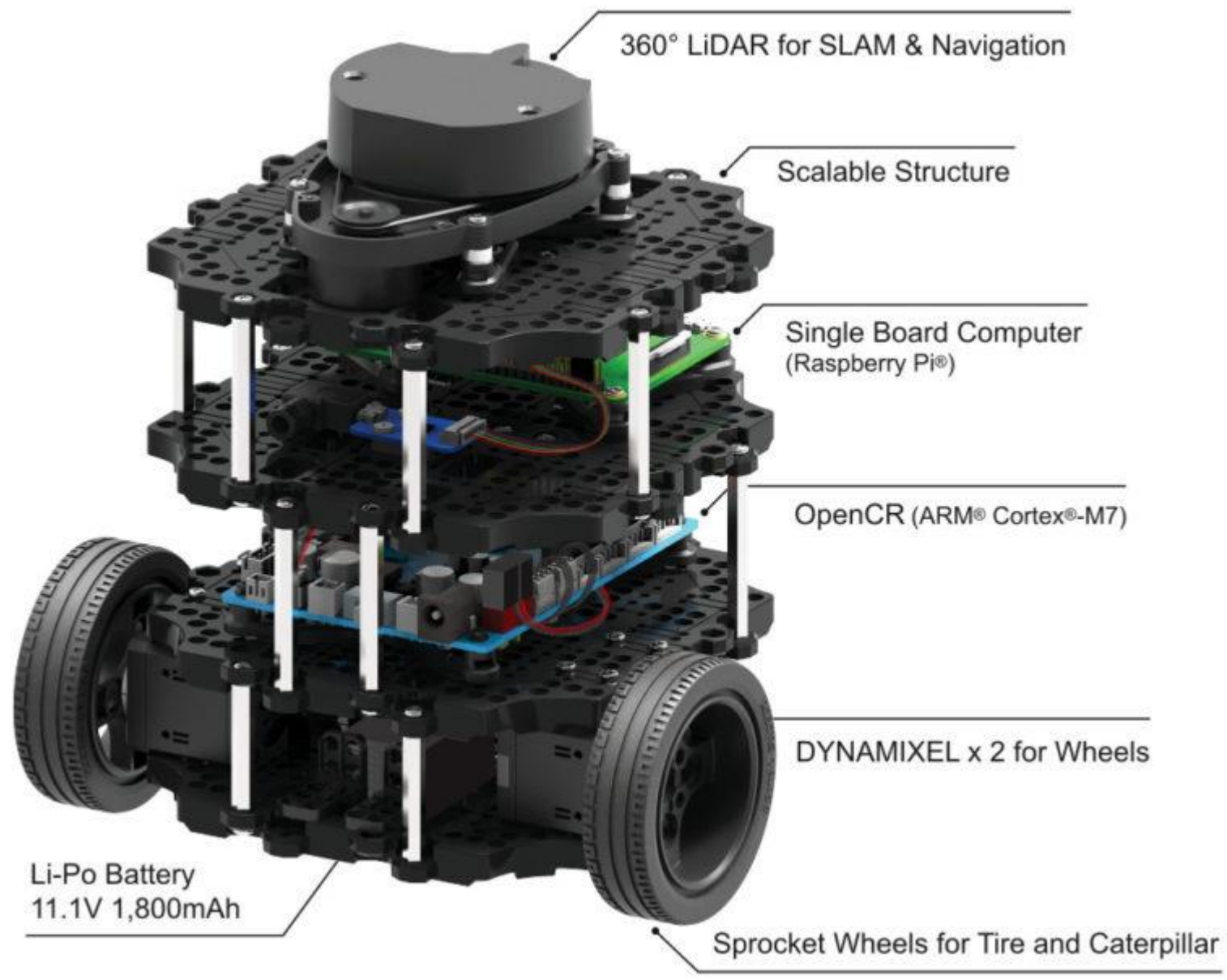
- System Monitor

- Receive and Display Detection Camera and info
- Receive and Display AMR Camera and info
- Store, display, and report Information and Alerts

INTRODUCTION TO AMR

- [TurtleBot3](https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/)
- <https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/>





POWER ON AND OFF AMR (VERY IMPORTANT!!!!)

POWER ON

- Make sure both Jetson and OpenCR is powered off.
- Must turn on OpenCR first!!!!
- Wait for the music from OpenCR
- Finally turn on Jetson

POWER OFF

- Execute proper shutdown

\$ sudo shutdown now

Turn off Jetson, Turn off OpenCR

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

SETUP PC FOR AMR

INSTALL DEPENDENT ROS 2 PACKAGES

\$ Check if installed

\$ apt list | grep nav2-map-server

\$ apt list | grep carto

If not,

\$ sudo apt install ros-humble-nav2-map-server

\$ sudo apt install ros-humble-cartographer

\$ sudo apt install ros-humble-cartographer-ros

TURTLEBOT3

<https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/>

SETUP PC FOR AMR

CHECK IF ALREADY INSTALLED

```
$ apt list | grep dynam
```

```
$ apt list | grep turtlebot3
```

INSTALL TURTLEBOT3 PACKAGES

If not already installed,

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

```
$ sudo apt install ros-humble-dynamixel-sdk
```

```
$ sudo apt install ros-humble-turtlebot3-  
msgs
```

```
$ sudo apt install ros-humble-turtlebot3
```

SETUP PC FOR AMR

- If you want to download the source code

```
$ mkdir -p ~/turtlebot3_ws/src
```

```
$ cd ~/turtlebot3_ws/src/
```

```
$ git clone -b humble-devel  
https://github.com/ROBOTIS-  
GIT/DynamixelSDK.git
```

```
$ git clone -b humble-devel  
https://github.com/ROBOTIS-  
GIT/turtlebot3_msgs.git
```

```
$ git clone -b humble-devel  
https://github.com/ROBOTIS-GIT/turtlebot3.git
```

```
$ cd ~/turtlebot3_ws
```

```
$ colcon build --symlink-install
```

```
$ echo 'source  
~/turtlebot3_ws/install/setup.bash' >> ~/.bashrc
```

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


HOW TO CONNECT TO AMR

PC

\$ sudo ufw status

\$ sudo ufw disable

- disables firewall for ubuntu systems

SETUP PC FOR AMR - ROS ID

PC

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

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

```
$ env | grep ROS
```

If ROS_DOMAIN_ID does not exist or is not correctly set,

```
$ echo 'export ROS_DOMAIN_ID=I  
#TURTLEBOT3' >> ~/.bashrc
```

(ID = 1,2,3,4,5 depends on your team number)

HOW TO CONNECT TO AMR

YOUR AMR IS A HOTSPOT

- Connect to wifi:
 - Turtlebot3_AP_<n>
 - PW: rokey12345

PC TERM I

\$ dpkg -l | grep openssh

If not installed...

\$ sudo apt install openssh-server -y

- Connect via ssh

HOW TO CONNECT TO AMR

PC TERM2

```
$ ssh -X  
rokey<n>@<ip_address  
>
```



SSH AMR TERM I

```
rokey<n> @ rokey....:$
```

PW: rokey1234

HOW TO CONNECT TO AMR

PC TERMINAL

```
$ ros2 run demo_nodes_cpp listener
```

SSH AMR TERMINAL

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

If ROS_DOMAIN_ID is not correctly set,

```
$ echo 'export ROS_DOMAIN_ID=1 #TURTLEBOT3' >>  
~/.bashrc
```

(ID = 1,2,3,4,5 depends on your team number)

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

```
$ ros2 run demo_nodes_cpp talker
```

CREATE YOUR WORKSPACE UNDER AMR \$HOME DIRECTORY

SSH AMR TERMINAL

```
$ mkdir  
~/rokey2_<grp_letter><grp_num>_ws  
(i.e. mkdir ~/rokey2_A2_ws)
```

- Put all your file under this directory and remove at the end of the class
- Delete the directory at the end of the class

SETTING UP TO USE TURTLEBOT3

PC TERM 1

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

Check if export command on the right is in the **.bashrc**, if not execute the command on the right

PC TERM 1

```
$ echo 'export  
TURTLEBOT3_MODEL=burger' >>  
~/.bashrc
```

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

SETTING UP TO USE TURTLEBOT3

PC TERM2

```
$ ssh -X  
rokey<n>@<ip_address>
```



SSH AMR TERM 1

```
$
```


SETTING UP TO USE TURTLEBOT3

SSH AMR TERM I

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

Check if export command on the right is in the .bashrc, if not execute the command on the right

SSH AMR TERM I

```
$ echo 'export  
TURTLEBOT3_MODEL=burger' >>  
~/.bashrc
```

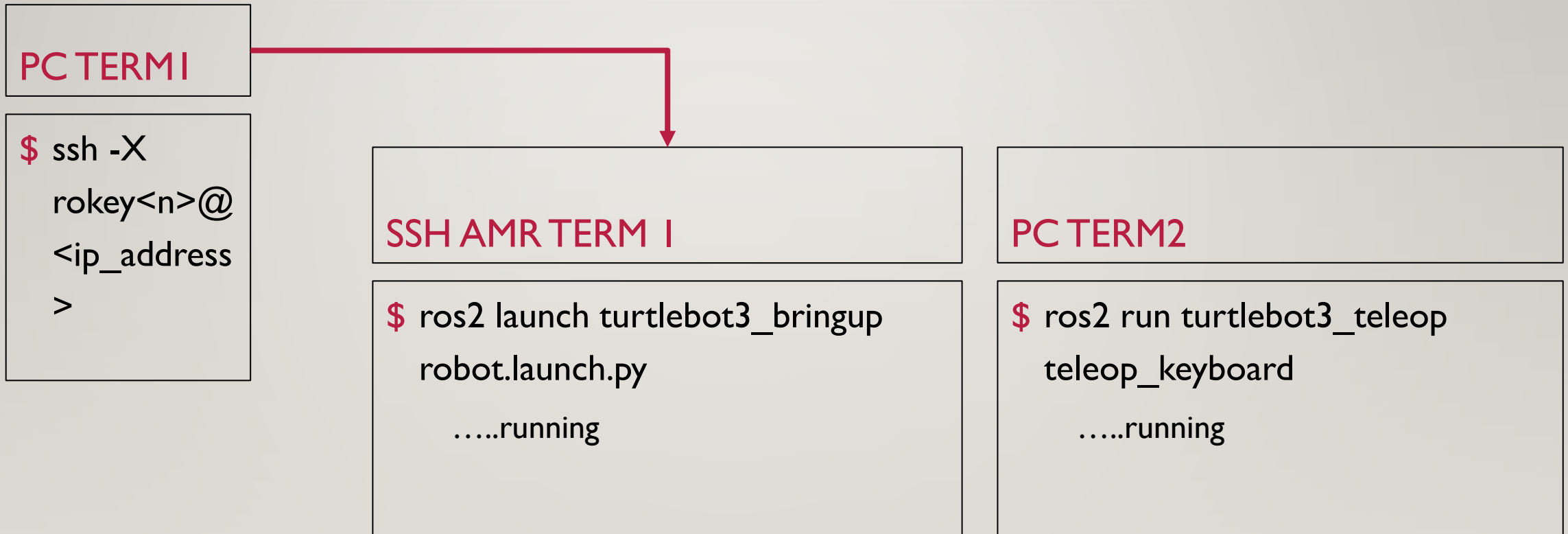
```
$ source ~/.bashrc
```

AMR DEMO

- [TurtleBot3](https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/)
- <https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/>
- Navigation with SLAM
- Teleop with keyboard



USING AMR TELEOP



DIGITAL MAPPING

STEP1: SSH AMR TERM 1

```
$ source ~/bashrc  
  
$ ros2 launch turtlebot3_bringup  
  robot.launch.py
```

STEP2: PC TERM 2

```
$ source ~/bashrc  
  
$ ros2 launch turtlebot3_cartographer  
  cartographer.launch.py
```


DIGITAL MAPPING

STEP1: SSH AMR TERM 1

```
$ ros2 launch turtlebot3_bringup  
  robot.launch.py  
  
..... running
```

STEP2: PC TERM 2

```
$ ros2 launch turtlebot3_cartographer  
  cartographer.launch.py  
  
..... running
```

STEP3: PC TERM 3

```
$ source ~/bashrc  
  
$ ros2 run turtlebot3_teleop  
  teleop_keyboard
```

DIGITAL MAPPING

STEP1: SSH AMR TERM 1

```
$ ros2 launch turtlebot3_bringup  
  robot.launch.py  
  
..... running
```

STEP2: PC TERM 2

```
$ ros2 launch turtlebot3_cartographer  
  cartographer.launch.py  
  
..... running
```

STEP3: PC TERM 3

```
$ ros2 run turtlebot3_teleop  
  teleop_keyboard  
  
..... running
```

STEP4: PC TERM 4(AT THE END)

```
$ source ~/bashrc  
  
$ ros2 run nav2_map_server  
  map_saver_cli -f ~/<my_dir>/map
```

DIGITAL MAPPING

CHECK IF CORRECT

\$ xdg-open <map-path>/map.pgm

Or,

\$ eog <map-path>/map.pgm

NAVIGATION W/ MAP

STEP1: SSH AMR TERM 1

```
$ source ~/bashrc  
  
$ ros2 launch turtlebot3_bringup  
  robot.launch.py
```

STEP2: PC TERM 2

```
$ source ~/.bashrc  
  
$ ros2 launch turtlebot3_navigation2  
  navigation2.launch.py  
  map:=<map_file_path> (i.e:  
    $HOME/my_dir/map/map.yaml)
```


NAVIGATION W/ MAP

STEP1: SSH AMR TERM 1

```
$ ros2 launch turtlebot3_bringup  
  robot.launch.py  
....running
```

STEP2: PC TERM 2

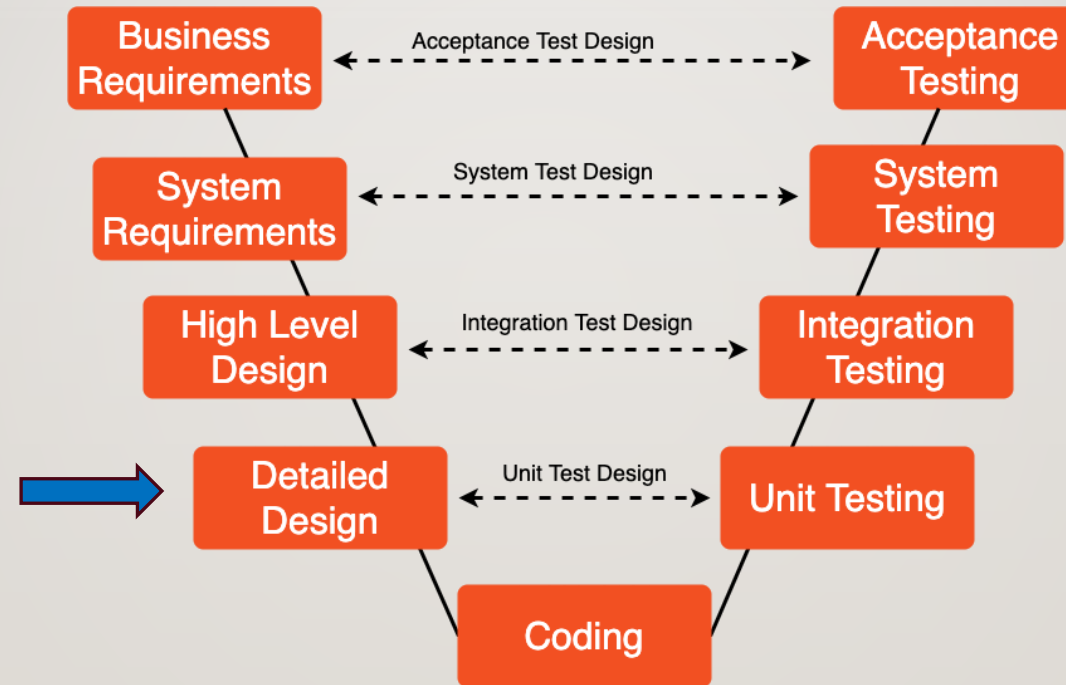
```
$ ros2 launch turtlebot3_navigation2  
  navigation2.launch.py  
  map:=<map_file_path> (i.e:  
    $HOME/my_dir/map/map.yaml)  
....running
```

*Perform 2D Pose Estimate and Send Goal

AMR CONTROLLER SPRINT



SPRINT 2 – AMR CONTROLLER



SDLC - V Model - notepub.io

TEAM EXERCISE 6

Perform Detail Design of AMR Controller Module using Process Flow Diagram

DESIGN HINT

- Initial Pose
 - nav2
 - rviz2 – 2D estimate pose
 - ros2 topic echo /initialpose
- How do you find AMR current position and orientation?
- Sending Goals
 - nav2
 - Rviz2 – send goals
 - ActionClient
 - NavigateToPose.send_goal_async()
- Stopping Navigation
 - NavigateToPose.cancel_all_goals_async()
- Sending multiple goals
 - ActionClient
 - /follow_waypoints
- Manual control of AMR Odometry
 - How to move forward, backward, left and right???

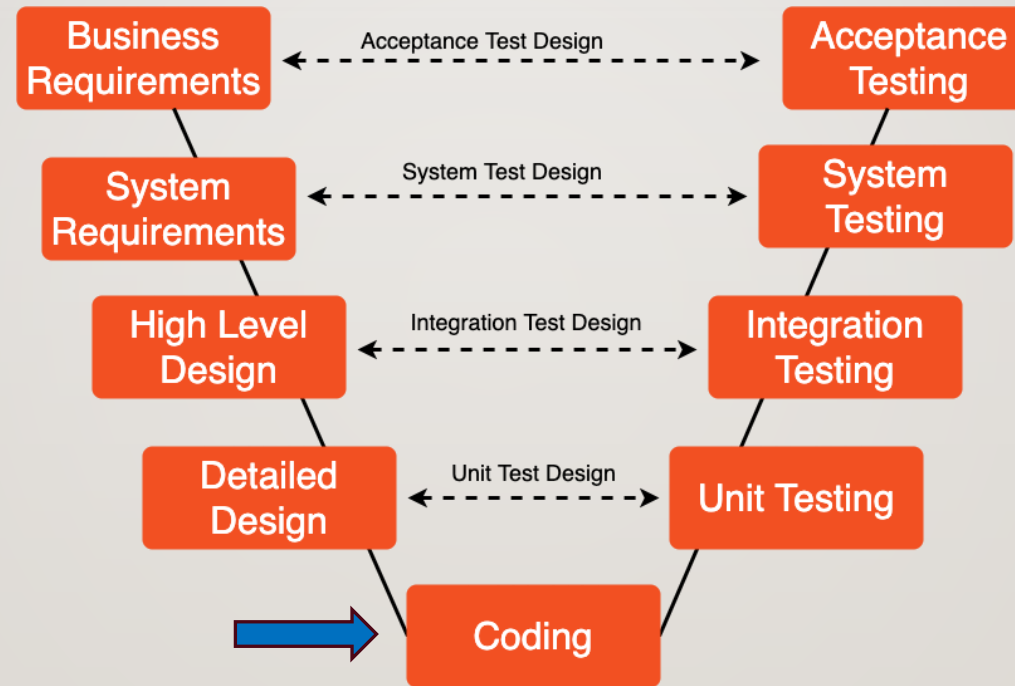
WHAT IS THE FOLLOW ALGORITHM?

- Left/Right?
- Forward/Backward?
- Velocity?
- Camera position?

DETAIL DESIGN REVIEW BY EACH TEAM

Using the process flow diagram present team's design

SPRINT 2 – AMR CONTROLLER



SDLC - V Model - notepub.io

SETTING UP VS CODE FOR REMOTE EDITING

- Install VS Code Remote - SSH Extension:
 - Open VS Code on your local machine.
 - Go to the Extensions view (Ctrl + Shift + X).
 - Search for "Remote - SSH" and install it.
- VS Code Remote - SSH 확장 프로그램 설치:
 - 로컬 컴퓨터에서 VS Code를 엽니다.
 - 확장 프로그램 보기로 이동합니다 (Ctrl + Shift + X).
 - "Remote - SSH"를 검색하여 설치합니다.

SETTING UP VSCODE FOR REMOTE EDITING

- Connect to the Remote Server:
 - Press F1 or Ctrl + Shift + P to open the Command Palette.
 - Type Remote-SSH: Connect to Host and select it.
 - Enter the SSH connection string (e.g., user@hostname) and connect.
- 원격 서버에 연결:
 - F1 또는 Ctrl + Shift + P를 눌러 명령 팔레트를 엽니다.
 - Remote-SSH: Connect to Host를 입력하고 선택합니다.
 - SSH 연결 문자열(예: user@hostname)을 입력하고 연결합니다.

SETTING UP VSCODE FOR REMOTE EDITING

- Open a Remote Folder:
 - Once connected, VSCode will display a new window with a remote indicator in the bottom-left corner.
 - You can open any folder or file from the remote server and edit it in your local VSCode instance.
- 원격 폴더 열기:
 - 연결되면 VSCode는 왼쪽 하단 모서리에 원격 표시기가 있는 새 창을 표시합니다.
 - 원격 서버에서 모든 폴더나 파일을 열고 로컬 VSCode 인스턴스에서 편집할 수 있습니다.

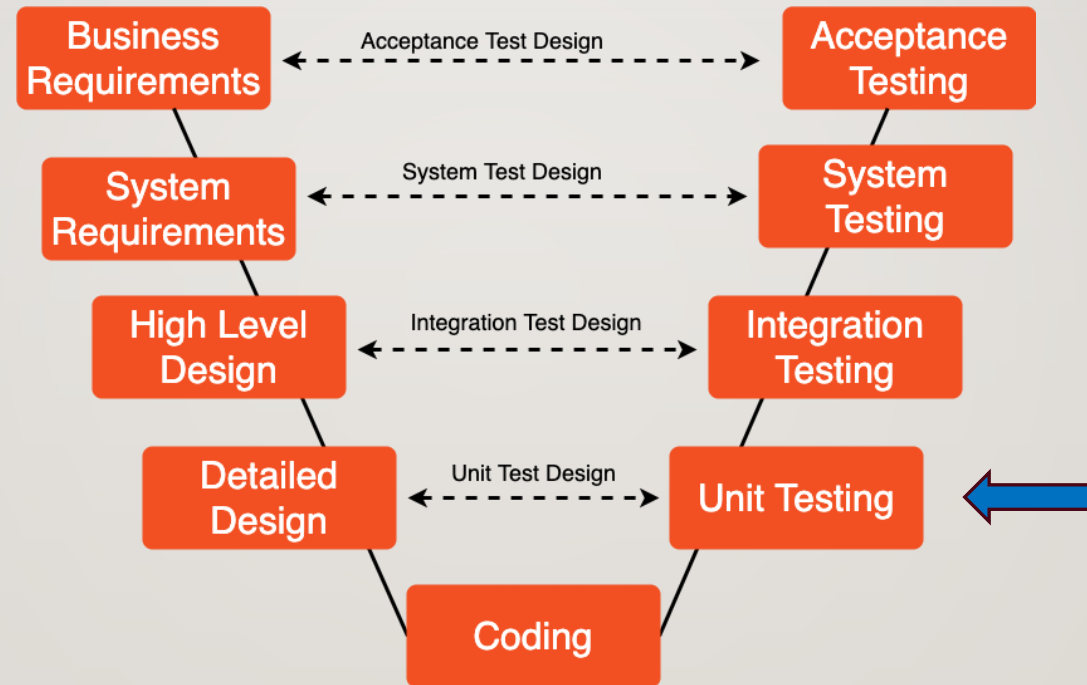
CODING HINT

- Initial Pose
 - nav2
 - rviz2 – 2D estimate pose
 - ros2 topic echo /initialpose
- Sending Goals
 - nav2
 - Rviz2 – send goals
 - ActionClient
 - NavigateToPose.send_goal_async()
 - ros2 topic echo /amcl_pose
- Stopping Navigation
 - ActionClient
 - NavigateToPose.cancel_all_goals_async()
- Sending multiple goals
 - ActionClient
 - /follow_waypoints
- Manual control of AMR Odometry
 - Twist
 - /cmd-vel

EXPECTED OUTCOME

AMR navigates to avoid obstacles, ignores dummies, track, and follow target

SPRINT 2 – AMR CONTROLLER



SDLC - V Model - notepub.io

TEAM EXERCISE 7

Perform coding and testing of AMR Controller Module

CODING HINT

▼ to_students

- 🔗 init_pose.py
- 🔗 send_goal_stop.py
- 🔗 send_waypoint.py
- 🔗 yolo_tracking.py

USING TURTLESIM TO TEST FOLLOWING LOGIC

STEP1: PC TERM 1

```
$ source ~/bashrc
```

```
$ ros2 run turtlesim turtlesim_node --ros-args -r /turtle1/cmd_vel:=/cmd_vel
```

STEP2: PC TERM 2






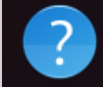






```
$ source ~/bashrc
```

```
$ ros2 run turtlebot3_teleop teleop_keyboard
```


Activities

turtlesim_node

1월 27 14:35



2010 clear
2011 ros2 run data_transfer data_subscriber
2012 apt list | sim
2013 apt list | grep sim
2014 ros2 run turtlesim turtlesim_node
2015 ros2 run turtlesim turtlesim_node --ros-args -r /turtle1/cmd_vel:=/cmd_vel
2016 ros2 run turtlesim turtlesim_node
2017 ssh -X rokey1@192.168.1.1
2018 history
2019 ros2 run turtlesim turtlesim_node
2020 history
2021 ros2 run turtlesim turtlesim_node --ros-args -r /turtle1/cmd_vel:=/cmd_vel
2022 ros2 node lit
2023 ros2 node list
2024 source .bashrc
2025 ros2 node list
2026 history
2027 ros2 run turtlesim turtlesim_node
2028 history
rokey@rokey-550XCJ-550XCR:~\$!2021
ros2 run turtlesim turtlesim_node --ros-args -r /turtle1/cmd_vel:=/cmd_vel
Warning: Ignoring XDG_SESSION_TYPE=wayland on Gnome. Use QT_QPA_PLATFORM=wayland, not qml.
[INFO] [1737956096.658672785] [turtlesim]: Starting turtlesim with node name /turtlesim
[INFO] [1737956096.661274208] [turtlesim]: Spawning turtle [turtle1] at x=[5.0] y=[0.0] theta=[0.0]
x: 2.0
y: 0.0
z: 0.0
angular:
x: 0.0
y: 0.0
z: 0.0

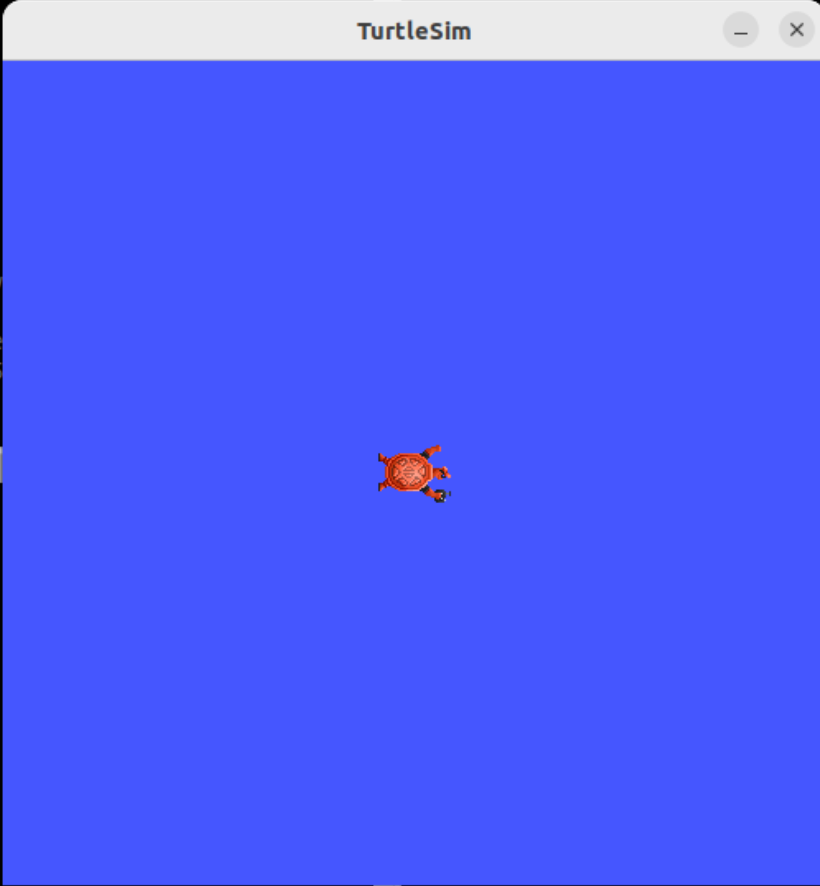
linear:
x: 0.0
y: 0.0
z: 0.0
angular:
x: 0.0
y: 0.0
z: 2.0

rokey@rokey-550XCJ-550XCR: ~ 100x27

rokey@rokey-550XCJ-550XCR: ~

rokey@rokey-550XCJ-550XCR:~\$

TurtleSim



ty 0.
ty 0.
ty 0.
ar vel
ar ve
space key, s : force stop

Activities

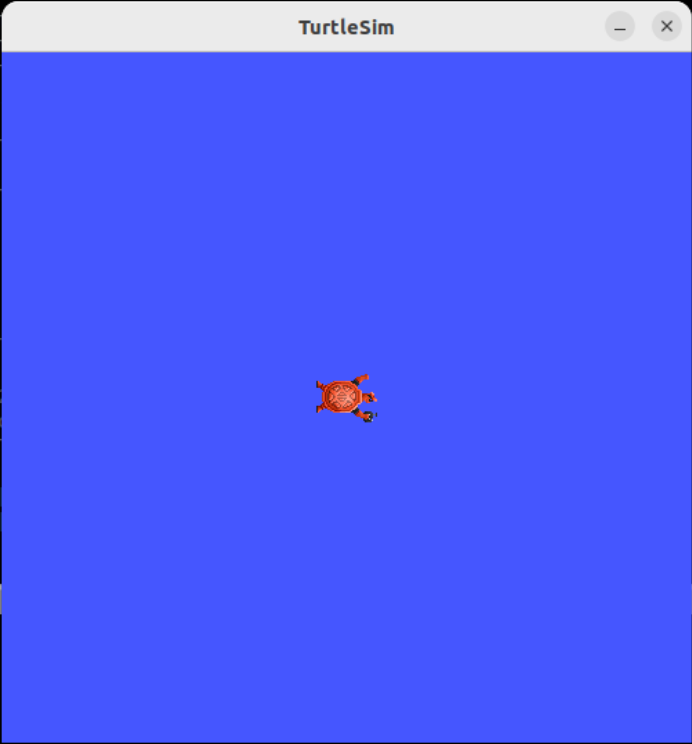
1월 27 / 14:36

rokey@rokey-550XCJ-550XCR: ~

rokey@rokey-550XCJ-550XCR: ~ 100x27

```
2010 clear
2011 ros2 run data_transfer data_subscriber
2012 apt list | sim
2013 apt list | grep sim
2014 ros2 run turtlesim turt
2015 ros2 run turtlesim turt
2016 ros2 run turtlesim turt
2017 ssh -X rokey1@192.168.1
2018 history
2019 ros2 run turtlesim turt
2020 history
2021 ros2 run turtlesim turt
2022 ros2 node lit
2023 ros2 node list
2024 source .bashrc
2025 ros2 node list
2026 history
2027 ros2 run turtlesim turt
2028 history
rokey@rokey-550XCJ-550XCR:~$ !
ros2 run turtlesim turtlesim_n
Warning: Ignoring XDG_SESSION_
nyway.
[INFO] [1737956096.658672785]
[INFO] [1737956096.661274208]
theta=[0.000000]
x: 2.0
y: 0.0
z: 0.0
angular:
x: 0.0
y: 0.0
z: 0.0
```

TurtleSim



rokey@rokey-550XCJ-550XCR: ~ 101x27

```
rokey@rokey-550XCJ-550XCR:~$ ros2 run turtlebot3_teleop teleop_keyboard

Control Your TurtleBot3!
-----
Moving around:
      w
    a  s  d
      x

w/x : increase/decrease linear velocity (Burger : ~ 0.22, Waffle and Waffle Pi : ~ 0.26)
a/d : increase/decrease angular velocity (Burger : ~ 2.84, Waffle and Waffle Pi : ~ 1.82)

space key, s : force stop

CTRL-C to quit


```

rokey@rokey-550XCJ-550XCR: ~ 101x27

```
currently:    linear velocity 0.0    angular velocity 2.84
currently:    linear velocity 0.0    angular velocity 2.84
currently:    linear velocity 0.0    angular velocity 2.84

Control Your TurtleBot3!
-----
Moving around:
      w
```



Activities

1월 27 / 14:37

rokey@rokey-550XCJ-550XCR: ~

rokey@rokey-550XCJ-550XCR: ~ 100x27

2010 clear

2011 ros2 run data_transfer data_subscriber

2012 apt list | sim

2013 apt list | grep sim

2014 ros2 run turtlesim turt

2015 ros2 run turtlesim turt

2016 ros2 run turtlesim turt

2017 ssh -X rokey1@192.168.1

2018 history

2019 ros2 run turtlesim turt

2020 history

2021 ros2 run turtlesim turt

2022 ros2 node lit

2023 ros2 node list

2024 source .bashrc

2025 ros2 node list

2026 history

2027 ros2 run turtlesim turt

2028 history

rokey@rokey-550XCJ-550XCR:~\$!

ros2 run turtlesim turtlesim_n

Warning: Ignoring XDG_SESSION_

nyway.

[INFO] [1737956096.658672785]

[INFO] [1737956096.661274208]

theta=[0.000000]

x: 2.0

y: 0.0

z: 0.0

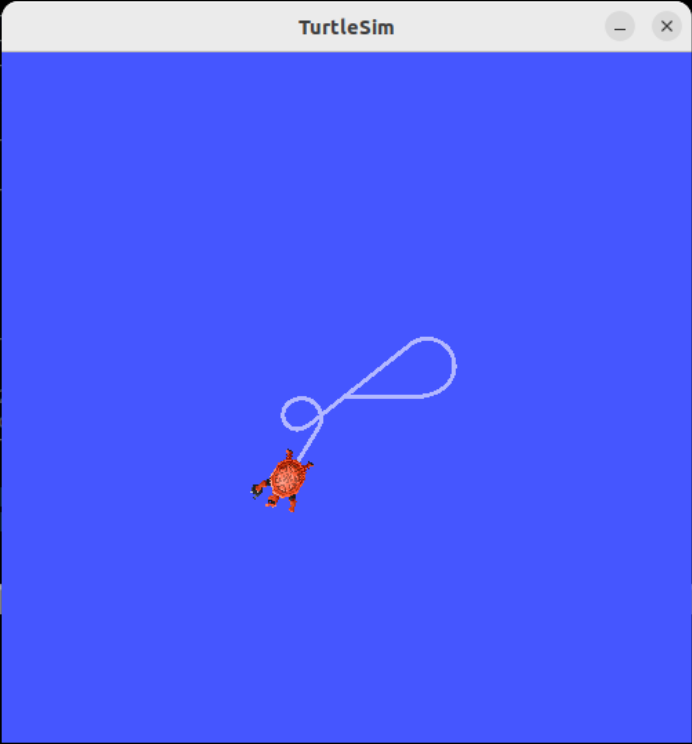
angular:

x: 0.0

y: 0.0

z: 0.0

TurtleSim



rokey@rokey-550XCJ-550XCR: ~ 101x27

a/d : increase/decrease angular velocity (Burger : ~ 2.84, Waffle and Waffle Pi : ~ 1.82)

space key, s : force stop

CTRL-C to quit

currently: linear velocity 0.03 angular velocity 0.0

currently: linear velocity 0.04 angular velocity 0.0

currently: linear velocity 0.05 angular velocity 0.0

currently: linear velocity 0.060000000000000005 angular velocity 0.0

currently: linear velocity 0.07 angular velocity 0.0

currently: linear velocity 0.08 angular velocity 0.0

currently: linear velocity 0.09 angular velocity 0.0

currently: linear velocity 0.09999999999999999 angular velocity 0.0

currently: linear velocity 0.10999999999999999 angular velocity 0.0

currently: linear velocity 0.11999999999999998 angular velocity 0.0

currently: linear velocity 0.12999999999999998 angular velocity 0.0

currently: linear velocity 0.13999999999999999 angular velocity 0.0

currently: linear velocity 0.15 angular velocity 0.0

currently: linear velocity 0.16 angular velocity 0.0

currently: linear velocity 0.17 angular velocity 0.0

currently: linear velocity 0.18000000000000002 angular velocity 0.0

currently: linear velocity 0.19000000000000003 angular velocity 0.0

currently: linear velocity 0.20000000000000004 angular velocity 0.0

currently: linear velocity 0.21000000000000005 angular velocity 0.0

currently: linear velocity 0.0 angular velocity 0.0

rokey@rokey-550XCJ-550XCR: ~ 101x27

currently: linear velocity 0.0 angular velocity 2.84

currently: linear velocity 0.0 angular velocity 2.84

currently: linear velocity 0.0 angular velocity 2.84

Control Your TurtleBot3!

Moving around:

w



USING TURTLESIM TO TEST FOLLOWING LOGIC

STEP1: PC TERM 1

```
$ source ~/bashrc  
$ ros2 run turtlesim turtlesim_node --ros-  
args -r /turtle1/cmd_vel:=/cmd_vel
```

STEP2: PC TERM 2

```
$ source ~/bashrc  
$ ros2 run turtlebot3_teleop  
teleop_keyboard
```

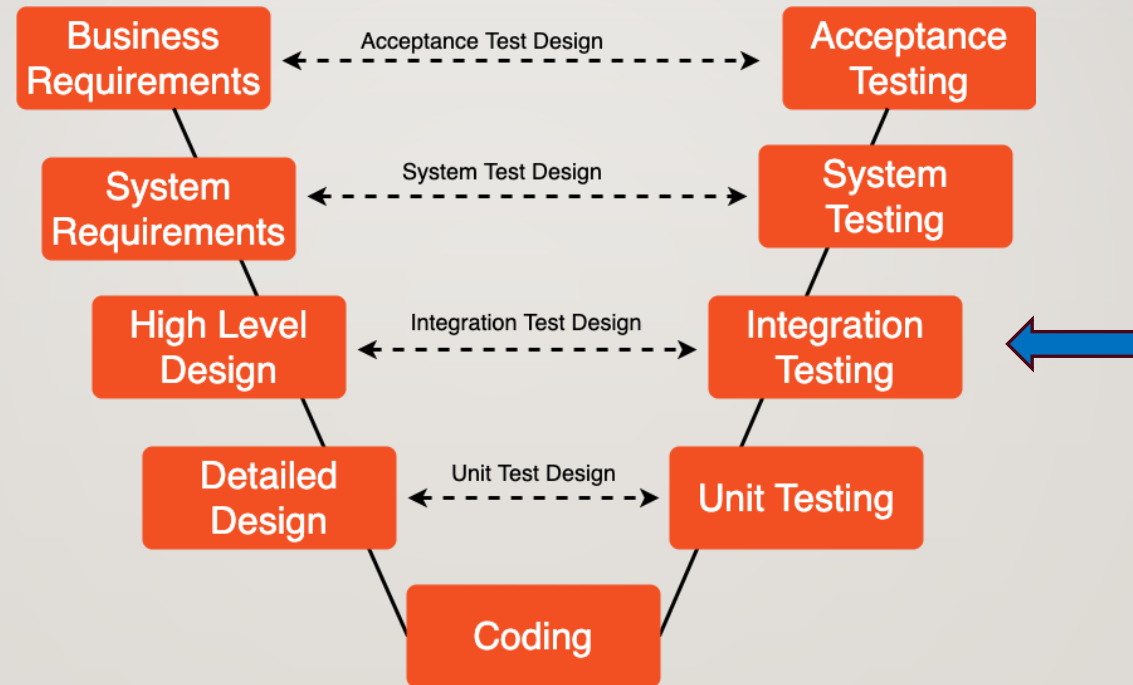


Replace with your code

RESULTS & CODE REVIEW BY EACH TEAM

Show actual results against the expected results and explain the code generated

SPRINT 1&2 – DETECTION ALERT/AMR CONTROLLER INTEGRATION & TEST



SDLC - V Model - notepub.io

EXPECTED OUTCOME

- Detection Alert and AMR Controller able to pass topics for necessary actions between

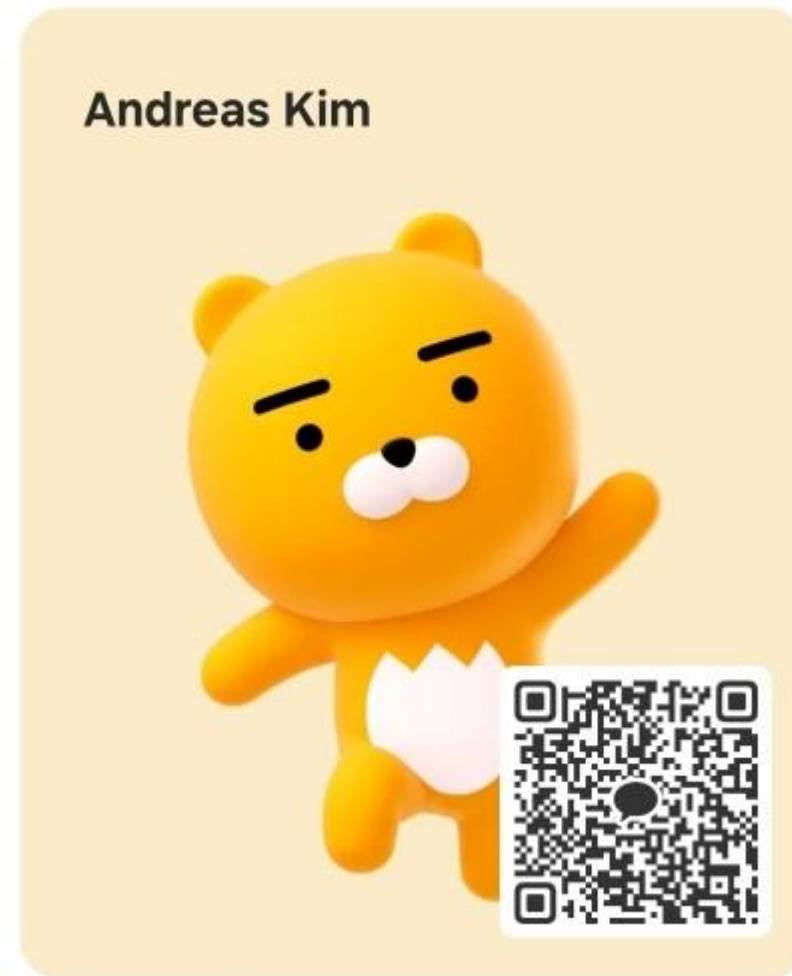
TEAM EXERCISE 8

Perform integrate and test of Detection Alert and AMR Controller Modules

RESULTS & CODE REVIEW BY EACH TEAM

Show actual results against the expected results and explain the code written

Send System Design Doc.
Here:



프로젝트 RULE NUMBER ONE!!!

Are we still having
FUN!

