EE405A
Autonomous System Configuration

School of Electrical Engineering
KAIST

Contents

Autonomous System

- Software Architecture

Software Configuration

- Setting up Jetson NX
- Install Libraries
- Install Sensor Interfaces
- Install Control Interface

Hardware Configuration

- Hardware Architecture
- Electronics
- Chassis

Notice

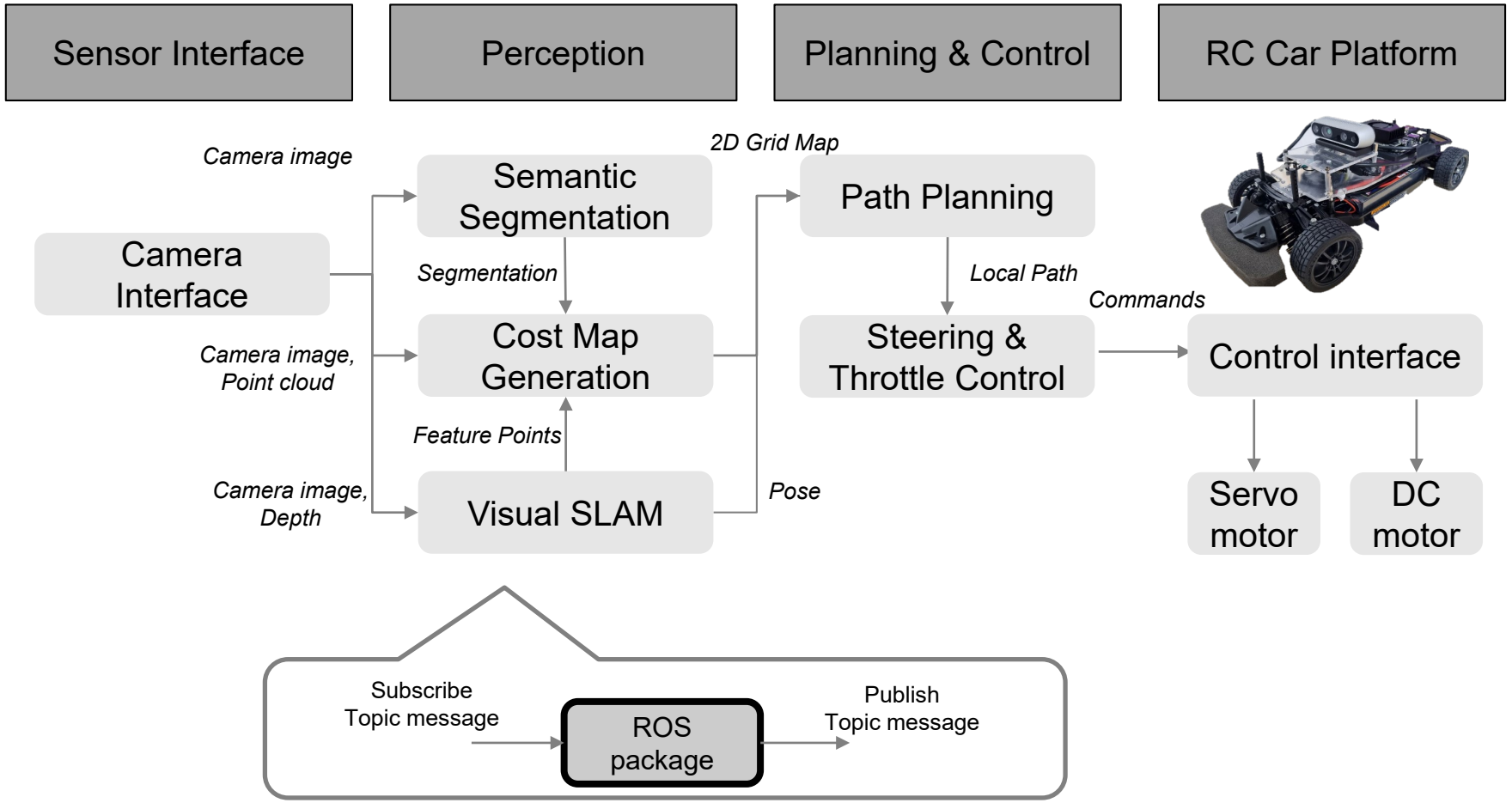
- Re-arrange Desktop PC Assignment
- Weekly Progress Report

Autonomous System

Software Architecture

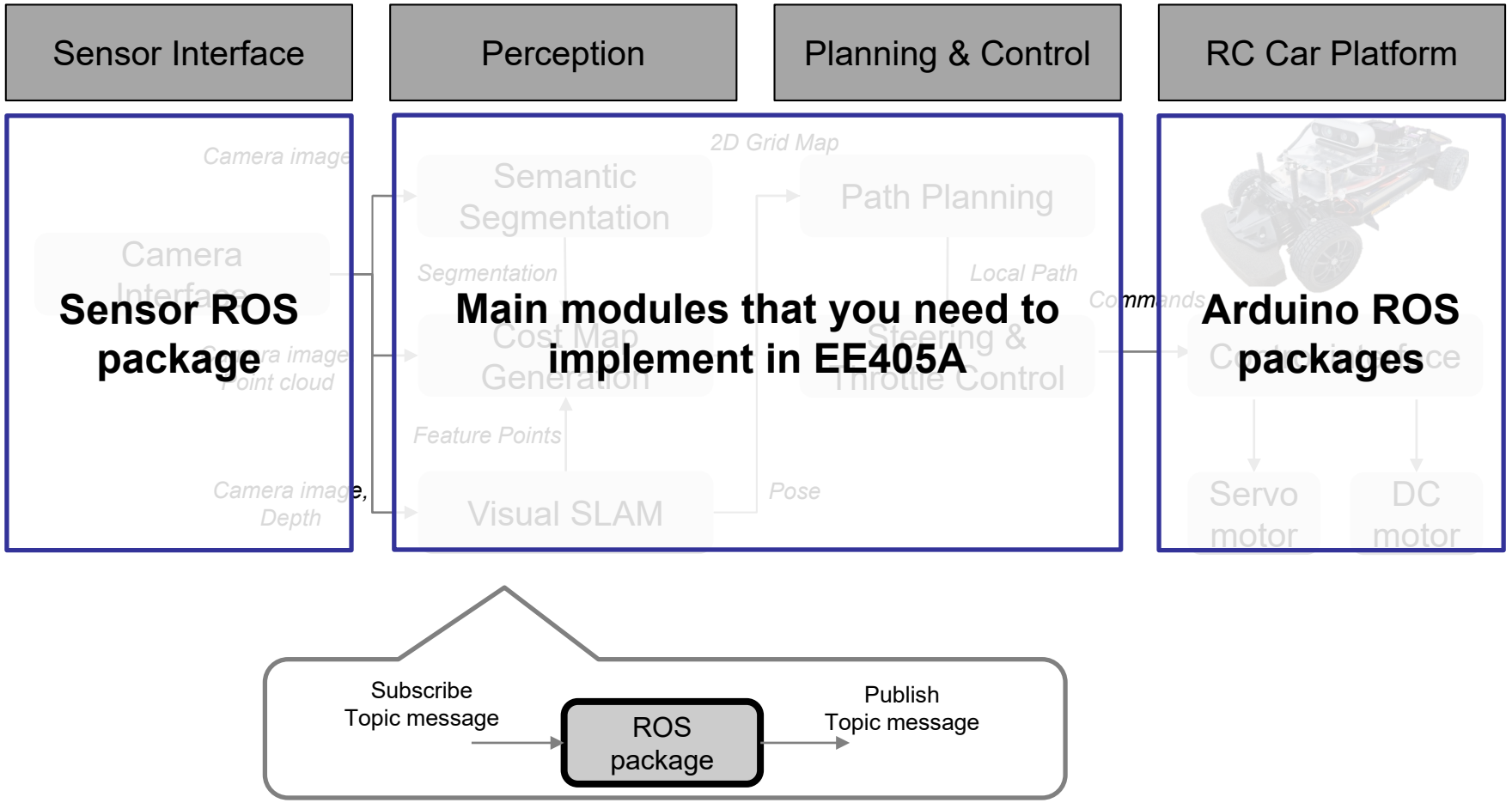
Software Architecture

- **Reference Software Architecture (Camera-only autonomous system)**



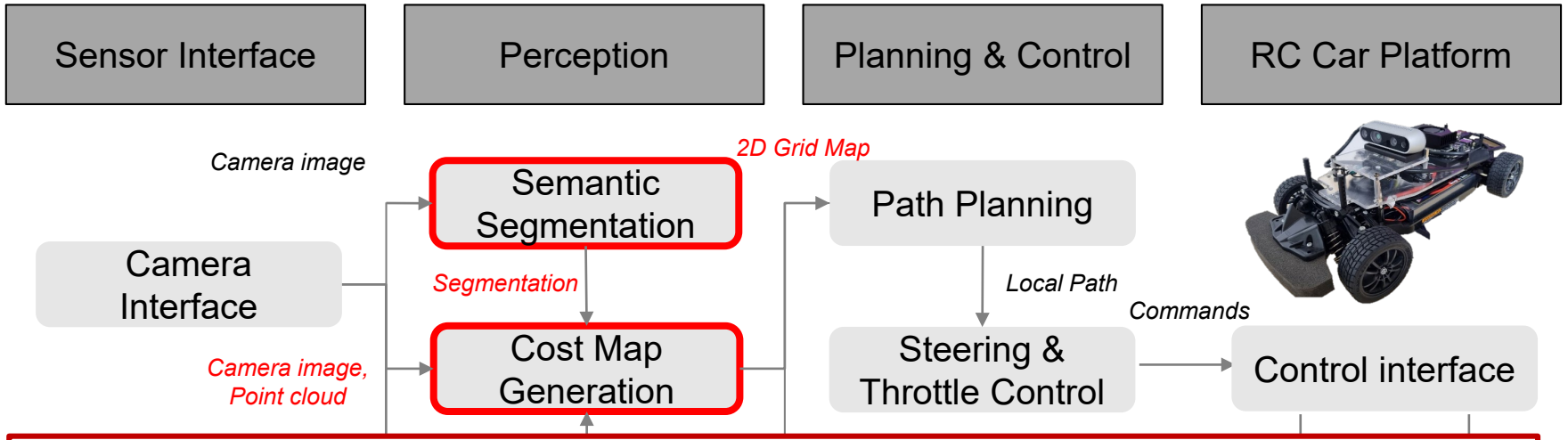
Software Architecture

- **Reference Software Architecture (Camera-only autonomous system)**



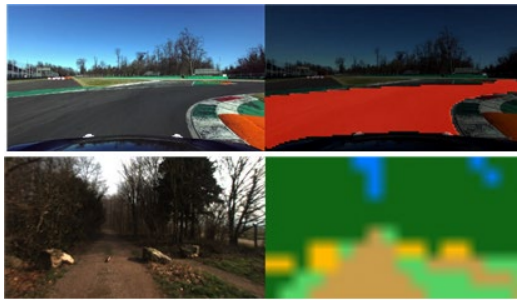
Software Architecture

➢ Perception – Cost Map Generation (w/ Semantic segmentation)

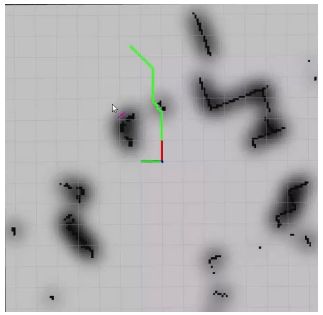


➢ Cost Map Generation (Week 9, 11)

- ❑ A cost map is a 2D grid map that indicates the **difficulty** or **danger level** when the robot plans a collision-free path.
- ❑ The grid map can be generated based on the semantic segmentation or point cloud data from RGB-D camera sensor.
- ❑ **Track boundary** and **other vehicles** would be represented on the 2D grid map for our project.



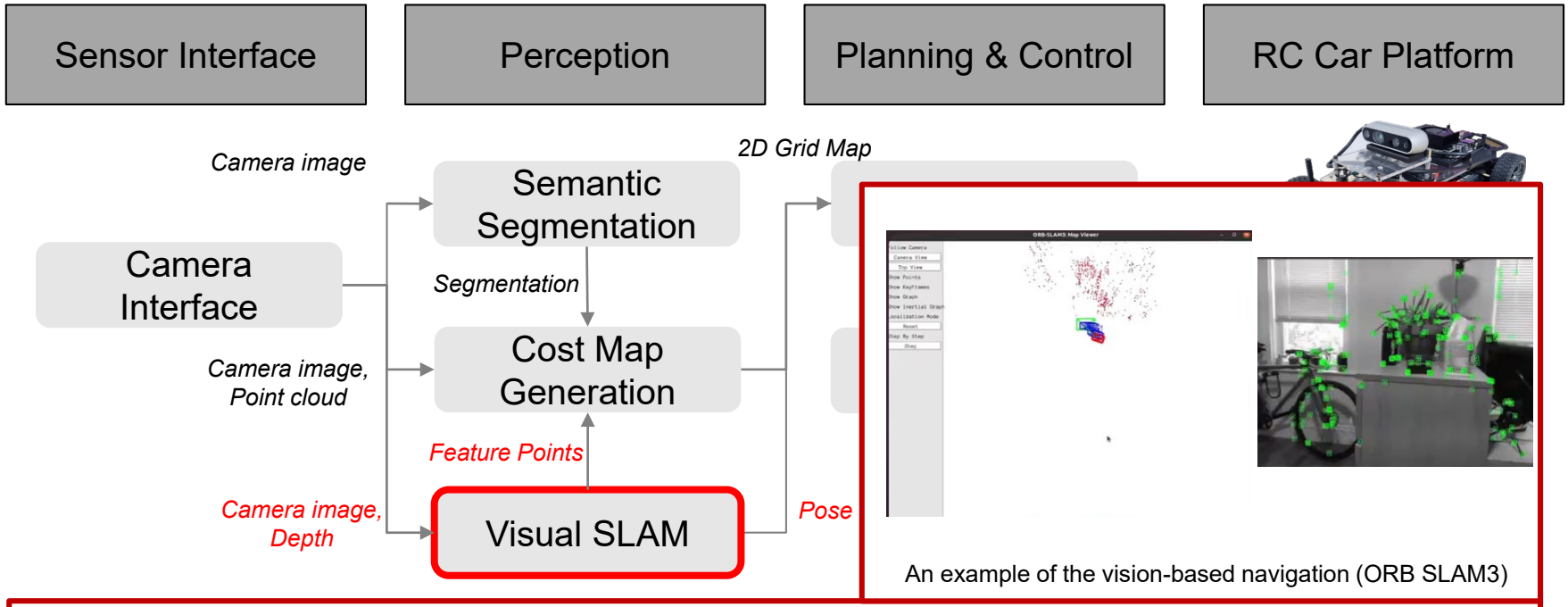
Semantic segmentation to recognize drivable regions



An example of the cost map generation

Software Architecture

➢ Perception – Visual SLAM

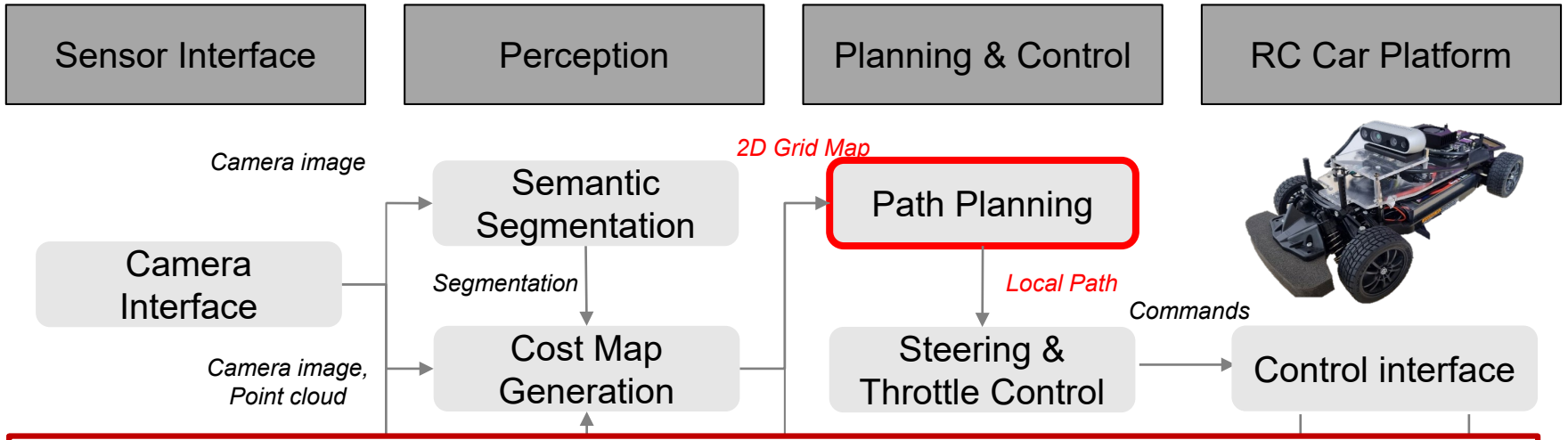


➢ Localization (Week 10)

- ❑ Localization is a module to know where the ego robot is located in the environment.
- ❑ Visual localization often incorporates Visual SLAM, the robot simultaneously builds a map of the environment while estimating the ego's current position within that map.
- ❑ The vehicle's position w.r.t. track environment can be used for the path planner or controller.

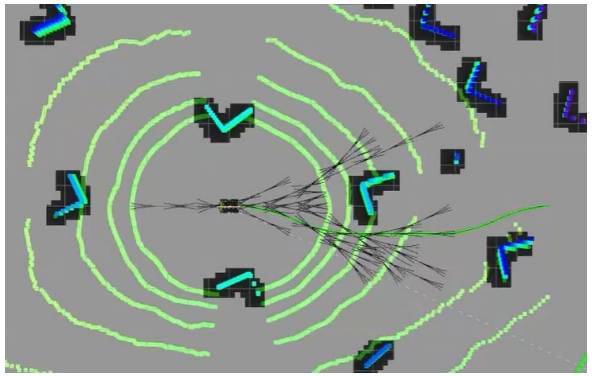
Software Architecture

➢ Planning & Control – Path Planning



➢ Path Planning (Week 9)

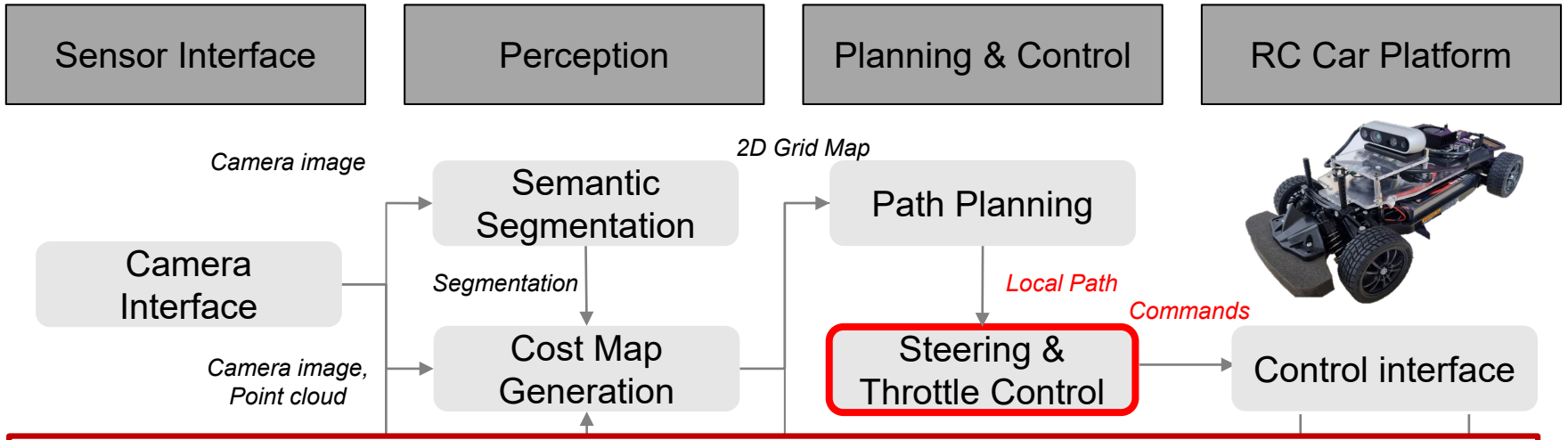
- ❑ Path planning is the process of **planning a safe path for a robot** to travel from a starting point to a goal position.
- ❑ **A 2D grid map** is used as information to generate collision-free paths.
- ❑ Colllision-free path generation enables the ego vehicle **to safely navigate the race track** and **avoid collisions from other vehicles.**



Collision-free path generation via Hybrid A star

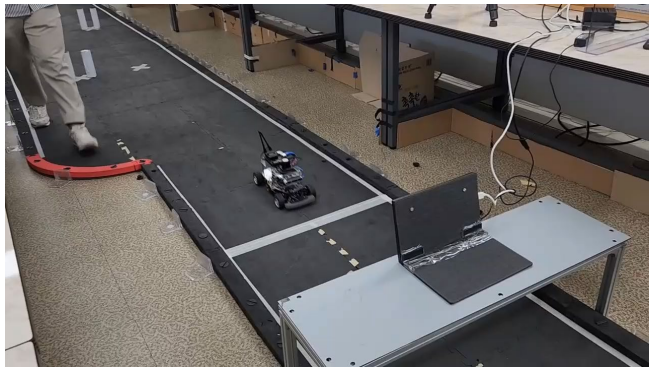
Software Architecture

➢ Planning & Control – Steering & Throttle Control



➢ Steering & Throttle Control (Covered in Week 5!)

- ❑ The control module computes steering angle and throttle commands for the ego vehicle to follow a given path.
- ❑ Proper lateral and longitudinal control needs to be designed to navigate the track environment with both straight lines and curves.



Autonomous control on the track environment

Software Configuration

Setting up Jetson NX
(for the Nvidia Jetson NX with microSD card)

Software Configuration

- **NVIDIA Jetson Xavier NX as the main PC**

Sensor Interface

RGBD Camera
(Realsense D435i)

RGB image
(/camera/color/image_raw)

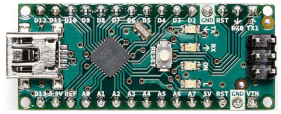
Depth image
(/camera/depth/image_rect_raw)

Point cloud
(/camera/depth_registered/points)

Main PC

Nvidia Jetson NX

Control Interface



Arduino Nano

Steering angle
(/rc_cmd/steer)
(/auto_cmd/steer)

Throttle
(/rc_cmd/throttle)
(/auto_cmd/throttle)

Driving mode
(/auto_mode)

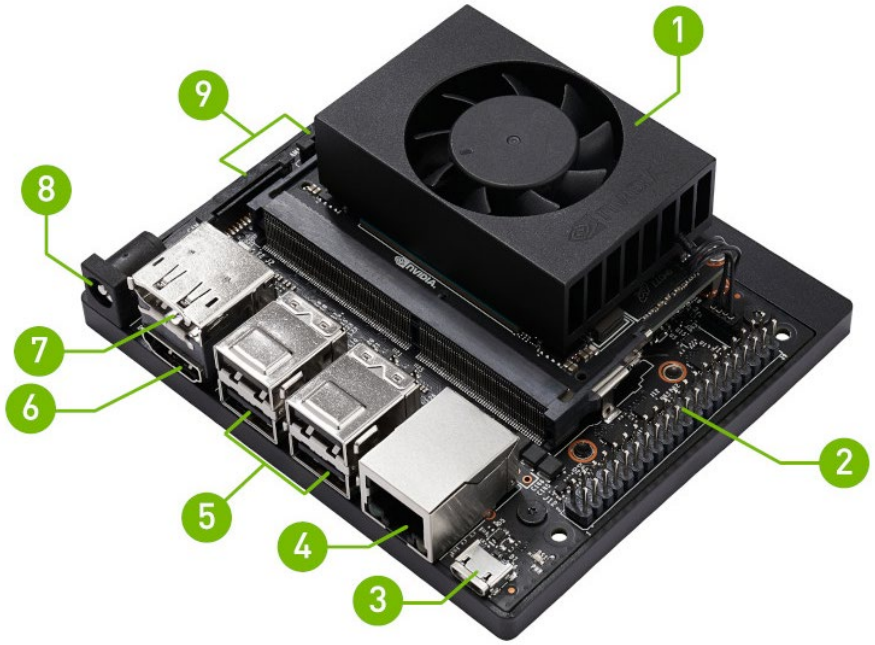
Setting up Jetson NX

- **NVIDIA Jetson Xavier NX**
  - The NVIDIA® Jetson Xavier NX™ Developer Kit includes a power-efficient, compact Jetson Xavier NX module for AI edge devices.
- **Specs**
  - **GPU** : NVIDIA Volta architecture with 384 NVIDIA CUDA® cores and 48 Tensor cores
  - **CPU** : 6-core NVIDIA Carmel ARM®v8.2 64-bit CPU 6 MB L2 + 4 MB L3
  - **Memory** : 8 GB 128-bit LPDDR4x @ 51.2GB/s
  - **Connectivity** : Gigabit Ethernet, M.2 Key E (WiFi/BT included), M.2 Key M (NVMe)
  - **Mechanical** : 103 mm x 90.5 mm x 34.66 mm

Setting up Jetson NX

- **Boot the Jetson Xavier NX with SD card image**
  - Follow the installation:

    https://developer.nvidia.com/embedded/learn/get-started-jetson-xavier-nx-devkit
  - Above Jetson Xavier NX SD card image already has Ubuntu OS with Nvidia driver.



① microSD card slot for main storage
② 40-pin expansion header
③ Micro-USB port
④ Gigabit Ethernet port
⑤ USB 3.1 Type A ports (x4)
⑥ HDMI output port
⑦ DisplayPort connector
⑧ DC Barrel jack for 19V power input
⑨ MIPI CSI camera connectors

Software Configuration

Setting up Jetson NX

(for the Nvidia Jetson NX with external SSD)

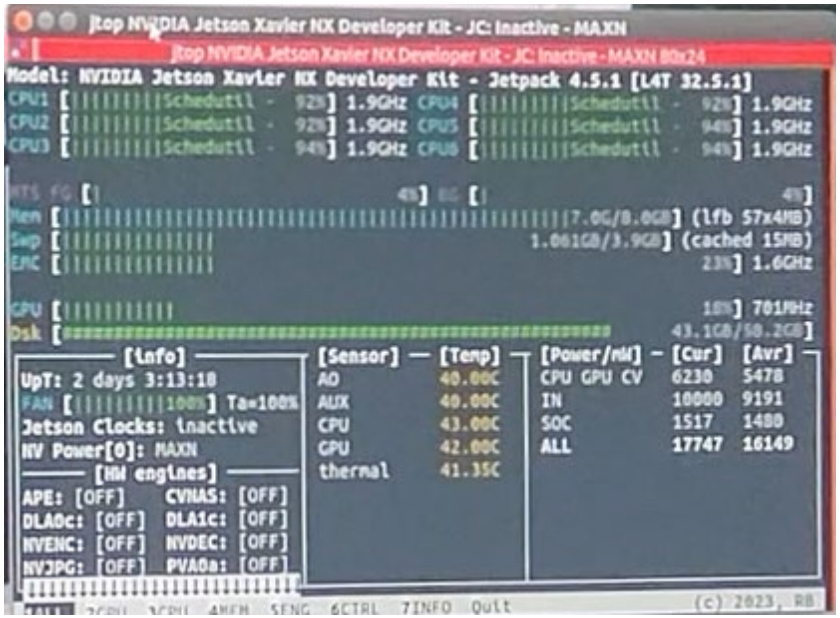
Please refer an extra manual of the new version of Jetson
'[EE405A] Jetson_NX_new_install_tutorial.pdf'

Software Configuration

Install Libraries

Install Libraries

- **Overclock settings for the Jetson Xavier NX**
  - Process of increasing the clock speed beyond its factory-set limits.
  - Overclock allows for enhanced performance and faster processing, mainly beneficial for high-intensity tasks or real-time applications like onboard computer on the autonomous driving robot.
  - It could damage the computer, so you should be careful when applying overclock.



CPU usage during overclock

Install Libraries

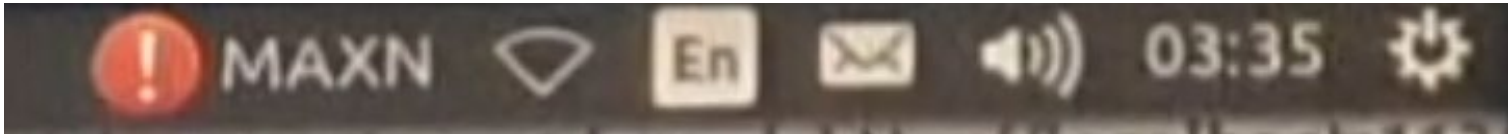
- **Overclock settings for the Jetson Xavier NX**
  - Download the Overclock repository:

    git clone https://github.com/yongeePark/XavierNXOverclock.git

  - Follow the process to implement overclock

```
sudo systemctl stop nvpmodel
cd [path-of-repository]/JetPack-4.5.1 [32.5.1]
sudo cp nvpmodel_t194_p3668.conf /etc/nvpmodel/nvpmodel_t194_p3668.conf
sudo systemctl start nvpmodel
sudo nvpmodel –m 0
sudo reboot now
```

  - After reboot, you may see ‘MAXN’ on the top right of the scereen



Install Libraries

- **pip**
  - sudo apt-get install python-pip
  - sudo apt-get install python3-pip
- **Jetson status monitoring program (jetson-stats)**
  - jetson-stats can be installed with pip, but need superuser:

    sudo pip3 install -U jetson-stats
  - See the following link: https://github.com/rbonghi/jetson_stats
- **Python 3.8** ※ If you don't want Python 3.8, you are free to install other versions.
  - sudo apt-get install python3.8-dev
- **ROS Melodic**
  - Same with the process you have done before on your PC.
  - Follow http://wiki.ros.org/ROS/Installation

Install Libraries

- **CUDA 10.2**
  - CUDA 10.2 has already installed in the Ubuntu image.
  - Add below in the ~/.bashrc:

```
export CUDA_HOME=/usr/local/cuda-10.2
export LD_LIBRARY_PATH=${CUDA_HOME}/lib64
PATH=${CUDA_HOME}/bin:${PATH}
export PATH
```

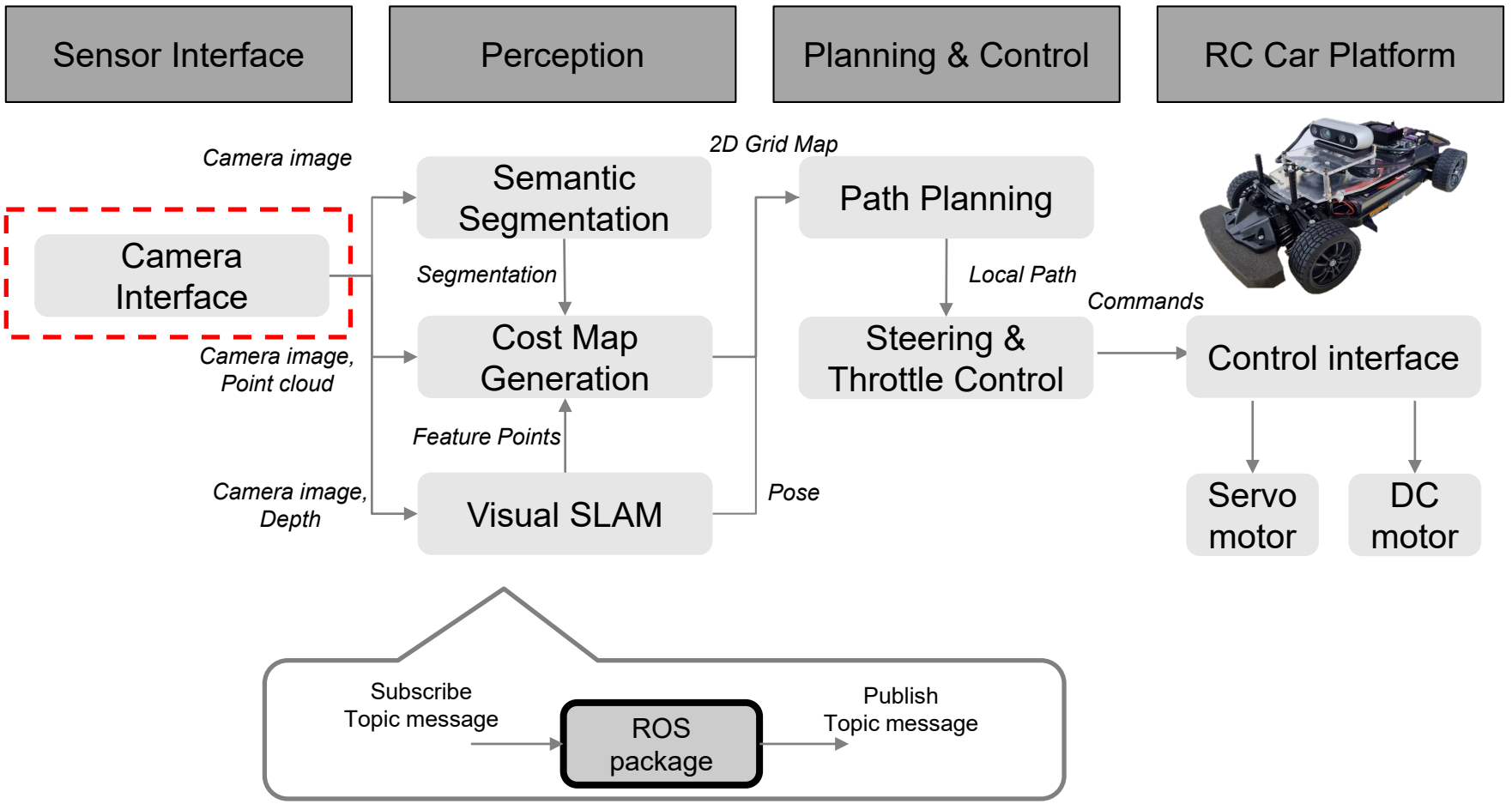
  - Check the CUDA version:

```
nvcc --version
```

Software Configuration

Install Sensor Interfaces

Install Sensor Interfaces



Install Sensor Interfaces

➢ Camera (Intel Realsense D435)

❑ Realsense SDK Installation process

- Add Public Key

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-key
F6E65AC044F831AC80A06380C8B3A55A6F3EFCDE || sudo apt-key adv --keyserver
hkp://keyserver.ubuntu.com:80 --recv-key F6E65AC044F831AC80A06380C8B3A55A6F3EFCDE
```

- Register Server Repository

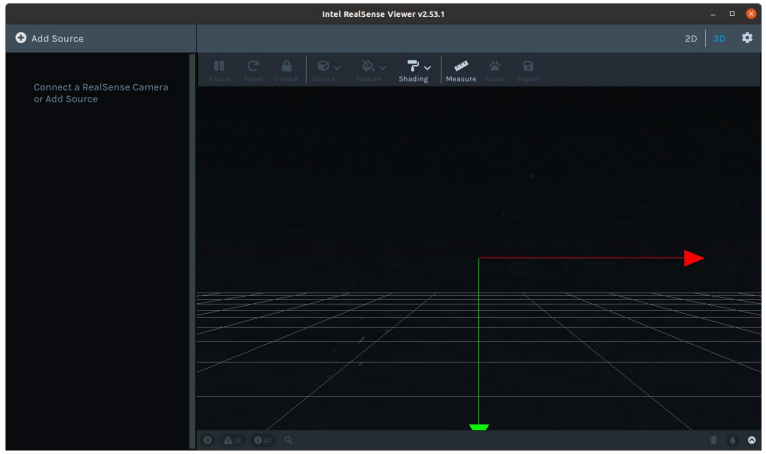
```
sudo add-apt-repository "deb https://librealsense.intel.com/Debian/apt-repo $(lsb_release -cs)
main" -u
```

- Install SDK Library

```
sudo apt-get install librealsense2-dkms
sudo apt-get install librealsense2-utils
sudo apt-get install librealsense2-dev
```

- Run SDK

```
realsense-viewer
```



Realsense viewer

Install Sensor Interfaces

➢ Camera (Intel Realsense D435)

❑ Realsense ROS : https://github.com/IntelRealSense/realsense-ros

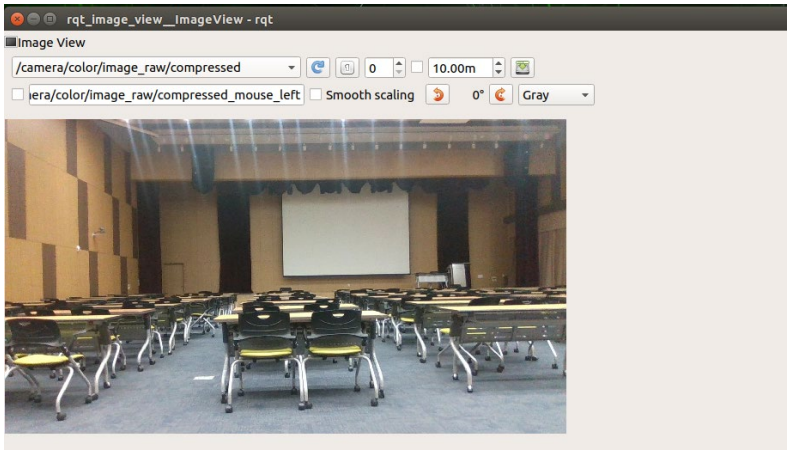
- Install Package

```
sudo apt-get install ros-noetic-realsense2-camera
```

- You can get the image topic by running the launch file in the package

```
roslauch realsense2_camera rs_camera.launch
rqt_image_view
```

RGBD Camera
(Realsense D435i)



Result Image (rqt_image_view)

Install Sensor Interfaces

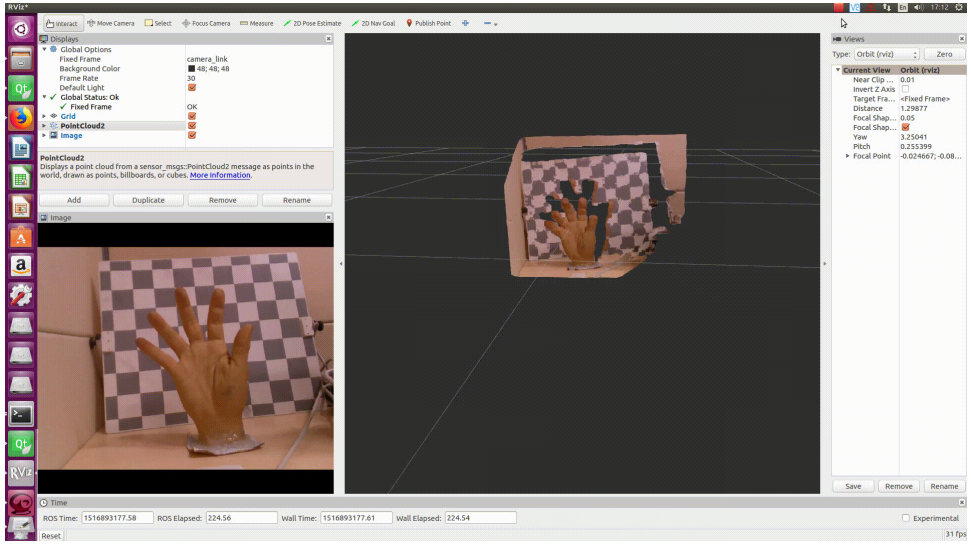
➢ Camera (Intel Realsense D435)

- ❑ Realsense ROS camera nodes example: https://dev.intelrealsense.com/docs/ros1-wrapper
  - Refer to the above link to test some examples of the realsense camera node.
  - By adding an option, you can start the camera node and publish point cloud data.

```
roslaunch realsense2_camera rs_camera.launch filters:=pointcloud
```

or

```
roslaunch realsense2_camera rs_rgbd.launch
```

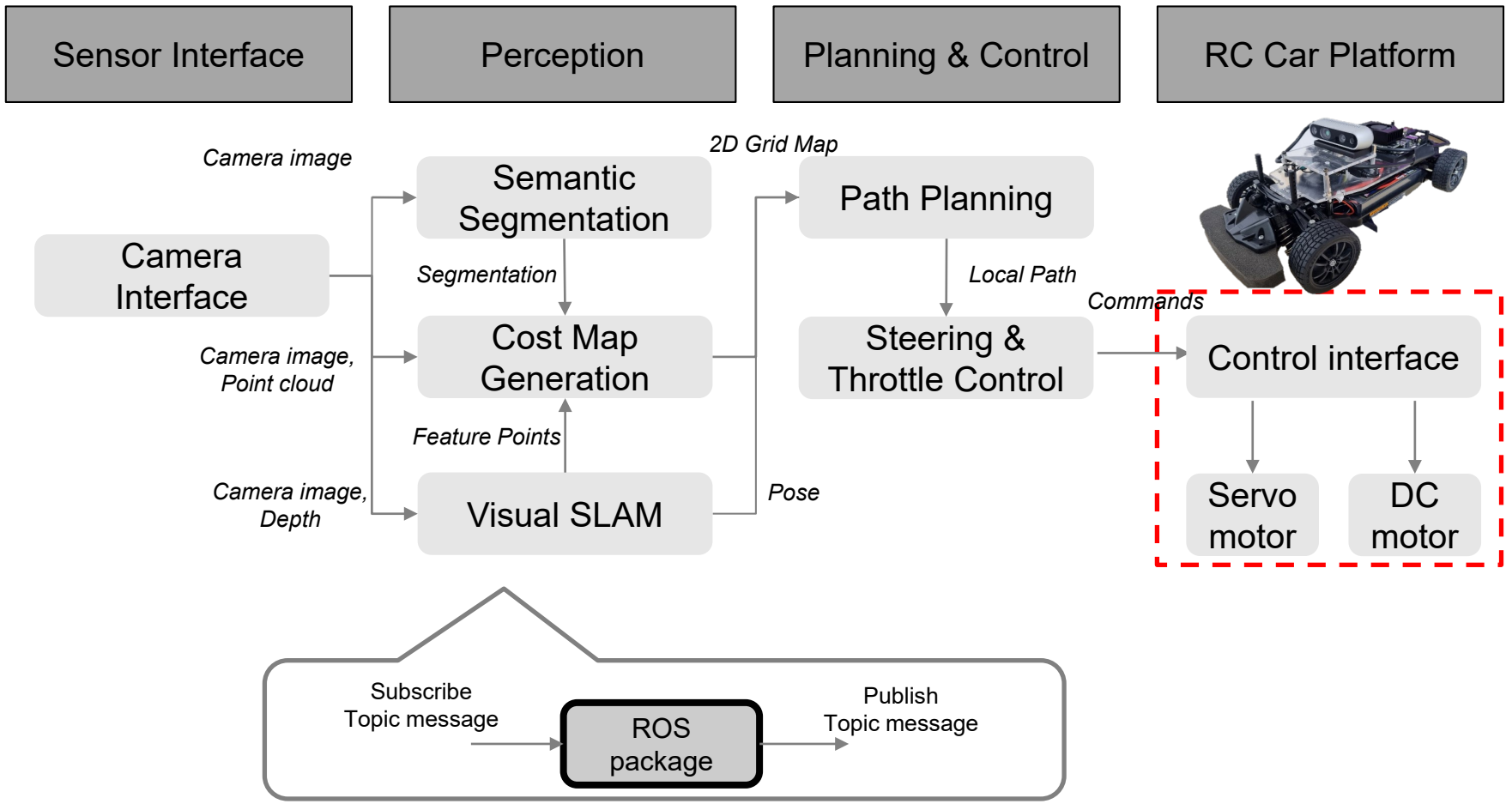


Result point cloud data (rviz)

Software Configuration

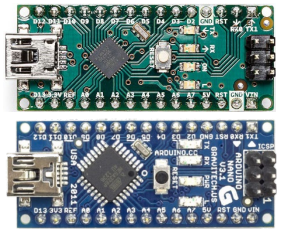
Install Control Interfaces

Install Control Interfaces



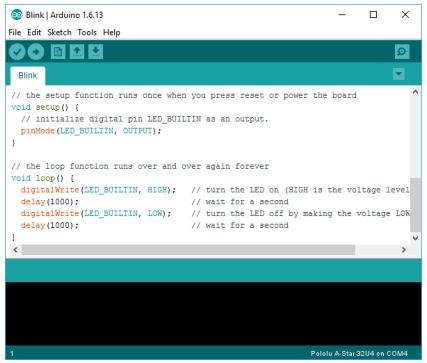
Install Control Interfaces

- **Embedded board (Arduino)**
  - Embedded board is required to control servo motor for steering and DC motor for throttling RC Platform.
  - For Arduino programming, Arduino IDE is required.
  - For communication between Arduino and Jetson NX, rosserial package is required.



Ariduino Nano

- **Install Arduino IDE**
  - in Jetson : https://github.com/JetsonHacksNano/installArduinoIDE
  - in Desktop (Ubuntu) : https://www.arduino.cc/en/guide/linux



Ariduino IDE

- **Install rosserial (ROS library for Arduino)**
  - Arduino IDE setup for rosserial : http://wiki.ros.org/rosserial_arduino/Tutorials/Arduino%20IDE%20Setup
- **Reference code**
  - rosserial_arduino tutorial : http://wiki.ros.org/rosserial_arduino/Tutorials

Install Control Interfaces

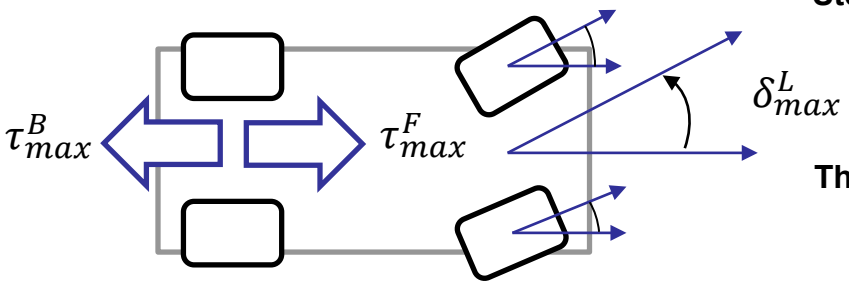
➢ Upload control interface .ino code

- ❑ Steering angle (Servo motor) and throttle (DC motor) are controlled by PWM commands.
- ❑ We use 'writeMicroseconds()' in the Servo library of Arduino.

  reference: https://www.arduino.cc/reference/en/libraries/servo/writemicroseconds/
- ❑ PWM for both motors ranges from 1100 to 1900 (1500 for neutral).
- ❑ Download a arduino control interface code for your RC car Platform :

  Download link: https://www.dropbox.com/s/3bdzybq87huvpch/ros_control_interface.zip?dl=0
- ❑ Upload the code using the Arduino IDE.

  Tutorials for the Arduino IDE: https://www.arduino.cc/en/Tutorial/HomePage/



Steer command (Servo motor) (PWM)

Leftmost ~ Neutral ~ Rightmost

δ^L_{max} ~ 0 ~ δ^R_{max}

1100 ~ 1500 ~ 1900

Throttle command (DC motor) (PWM)

Backward ~ Neutral ~ Forward

τ^B_{max} ~ 0 ~ τ^F_{max}

1100 ~ 1500 ~ 1900

※ The direction of the PWM value would be different.

※ Therfore, you need to check the PWM 1100 is for whether leftmost or rightmost steering angle, backward or forward throttle, respectively.

Install Control Interfaces

➢ Command for Jetson-Arduino ROS communication

- ❑ Modify the permission of the serial port:

  sudo chmod 777 /dev/tty*

  ※ This command is not persist when the terminal window is off.

  ※ You may also need this command before launching the IMU sensor.

- ❑ Run the ROS pakcage for Arduino communication:

  rosrun rosserial_python serial_node.py _port:=/dev/ttyUSB0 _baud:=57600

  ※ You can check the port number using below command: ls /dev/tty*

  ※ Usually, the port number is ttyACM* or ttyUSB*.

➢ ROS messages for Control Interface

- ❑ **/rc_cmd/steer**: Int, steer value (1100~1900) from transmitter (Manual controller)
- ❑ **/rc_cmd/throttle**: Int, throttle value (1100~1900) from transmitter (Manual controller)
- ❑ **/auto_cmd/steer**: Int, steer value (1100~1900) for autonomous control (from Jetson, ROS)
- ❑ **/auto_cmd/throttle**: Int, throttle value (1100~1900) for autonomous control (from Jetson, ROS)
- ❑ **/auto_mode**: Bool, true (for autonomous control) / false (for manual control)

Install Control Interfaces

- **Self-check**
  - Default is manual control mode (/auto_mode is false).
  - After completing hardware configuration, your vehicle should be controlled by your transmitter (as the default value of '/auto_mode' is false, i.e., manual control mode).
  - Try to publish '/auto_mode' message as true and publish '/auto_cmd/steer' or '/auto_cmd/steer' with values in (1100~1900). If the ROS commands control your vehicle, your control interface has now configured well.
- **Issues**
  - Sometimes the port number would be changed after unplug and re-plug the USB cable.
  - Make sure the rosserial package version is 'melodic

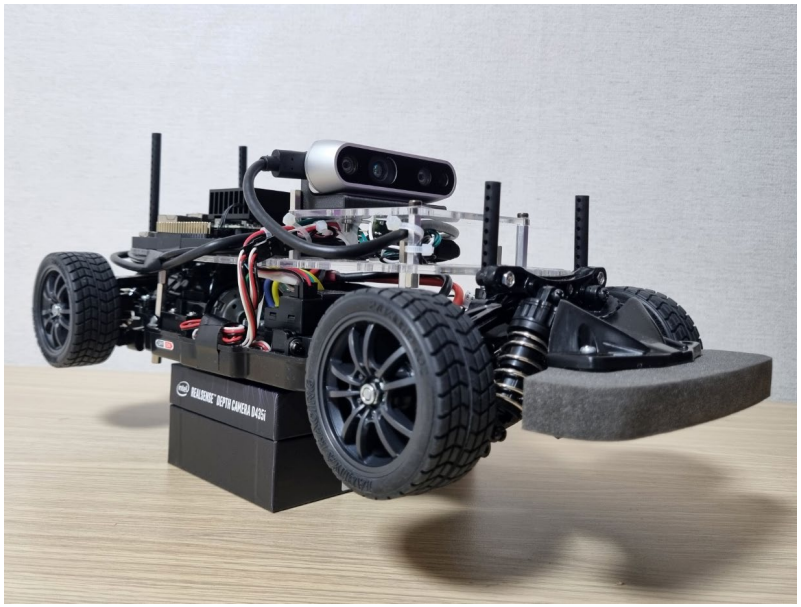
    sudo apt-get install ros-melodic-rosserial-arduino

    sudo apt-get install ros-melodic-rosserial
  - Define appropriate baud rate. (Default baud rate is 57600)

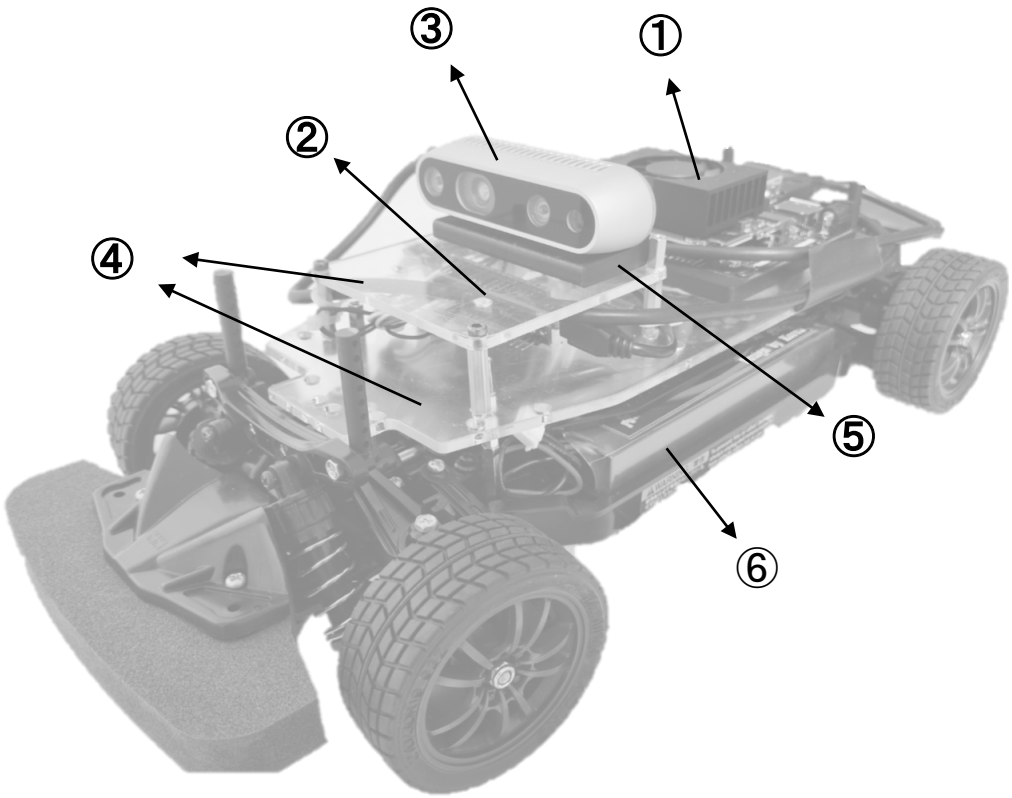
Hardware Configuration

Hardware Architecture

Hardware Architecture



Hardware Architecture



① Board
(Jetson NX2)

② Board
(Arduino Nano)

③ Sensor – Camera
(Realsense D435i)

④ Chassis
(Acrylic Plate)

⑤ Mount
(3D printed camera mount)

⑥ Battery
(LiPo, 2 cell)

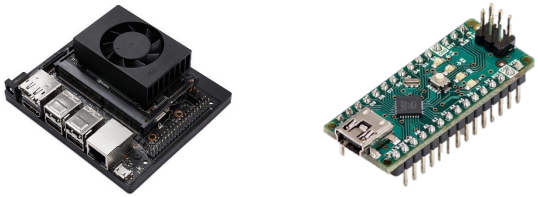
Hardware Configuration

Electronics

Electronics

- **Select electronics to be mounted on RC platform**
  - Sensors : RGB-D Camera-only
  - Boards : Jetson NX, Arduino Nano
  - Power source : 4 cell, 14.4V battery for power source of Jetson NX (5.5~19.6V), Arduino (6~20V) and the sensor
- **Select USB ports to facilitate communication between the sensor and boards**
- **Wiring for power supply**
  - Power supply for the Jetson from the battery.
  - In this lecture, we only need connectors for the battery and Jetson; there is no need for the connector for LiDAR.
- **Wiring between Arduino and Receiver**
  - Receive manual commands from the transmitter and send auto/manual commands to the motors.

RGB-D Camera (D435)

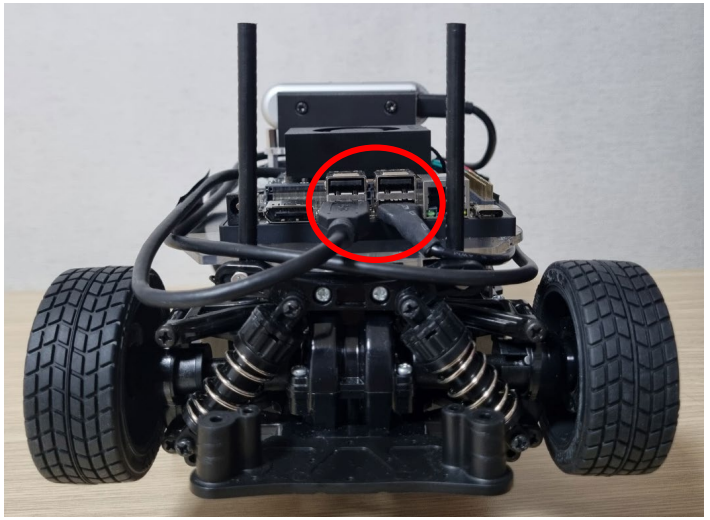


Boards (Jetson NX2, Arduino Nano)

Power source (LiPO Battery 3300mAh)

Electronics

- **Select electronics to be mounted on RC platform**
  - Sensors : RGB-D Camera-only
  - Boards : Jetson NX, Arduino Nano
  - Power source : 4 cell, 14.4V battery for power source of Jetson NX (5.5~19.6V), Arduino (6~20V) and the sensor
- **Select USB ports to facilitate communication between the sensor and boards**
- **Wiring for power supply**
  - Power supply for the Jetson from the battery.
  - In this lecture, we only need connectors for the battery and Jetson; there is no need for the connector for LiDAR.
- **Wiring between Arduino and Receiver**
  - Receive manual commands from the transmitter and send auto/manual commands to the motors.

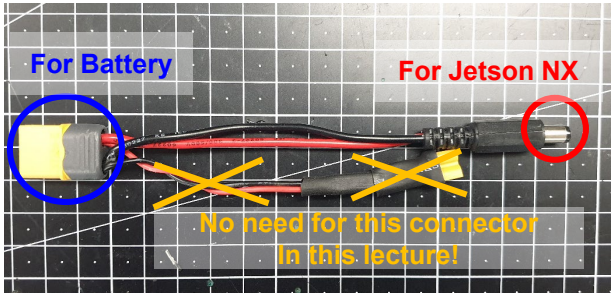


USB connect between the camera sensor (D435), Arduino (Nano), and the Nvidia Jetson.

Electronics

- **Select electronics to be mounted on RC platform**
  - Sensors : RGB-D Camera-only
  - Boards : Jetson NX, Arduino Nano
  - Power source : 4 cell, 14.4V battery for power source of Jetson NX (5.5~19.6V), Arduino (6~20V) and the sensor
- **Select USB ports to facilitate communication between the sensor and boards**
- **Wiring for power supply**
  - Power supply for the Jetson from the battery.
  - In this lecture, we only need connectors for the battery and Jetson; there is no need for the connector for LiDAR.
- **Wiring between Arduino and Receiver**
  - Receive manual commands from the transmitter and send auto/manual commands to the motors.

Power source (LiPO Battery 3300mAh, 4S cell 14.8V)

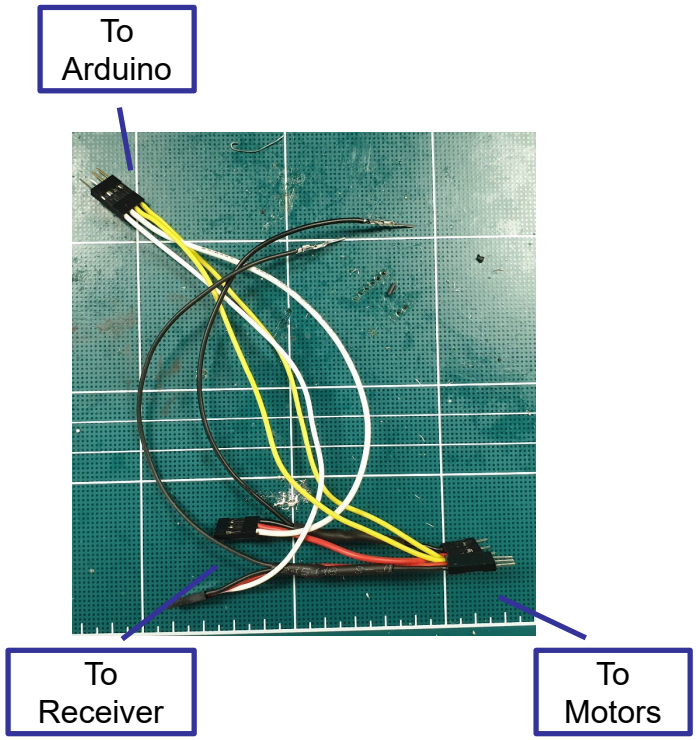


Wire for the power supply

You need to solder this! (1st)

Electronics

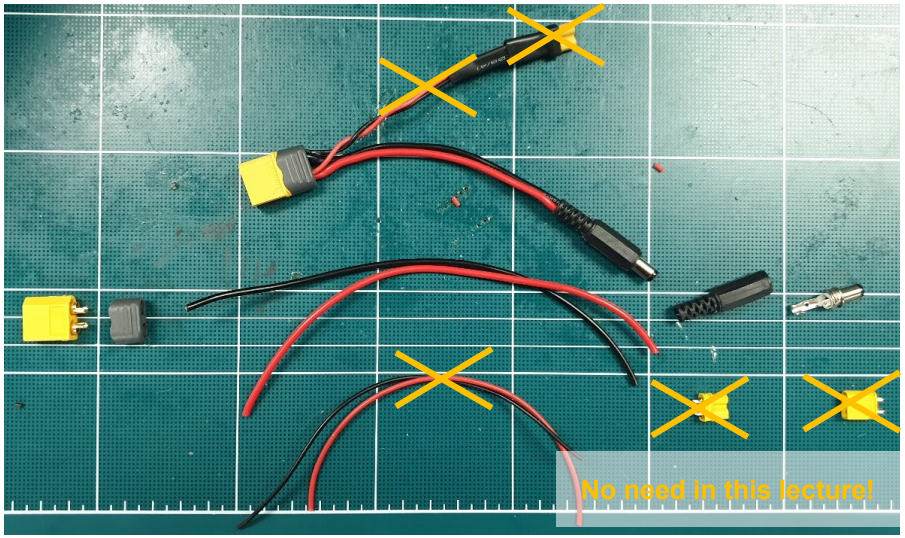
- **Select electronics to be mounted on RC platform**
  - Sensors : RGB-D Camera-only
  - Boards : Jetson NX, Arduino Nano
  - Power source : 4 cell, 14.4V battery for power source of Jetson NX (5.5~19.6V), Arduino (6~20V) and the sensor
- **Select USB ports to facilitate communication between the sensor and boards**
- **Wiring for power supply**
  - Power supply for the Jetson from the battery.
  - In this lecture, we only need connectors for the battery and Jetson; there is no need for the connector for LiDAR.
- **Wiring between Arduino and Receiver**
  - Receive manual commands from the transmitter and send auto/manual commands to the motors.



You need to solder this! (2nd)

Electronics

- **Wiring for power supply**
  - The Jetson NX needs the wire connection for power supply.
  - The LiPO battery can directly supply power for NX.

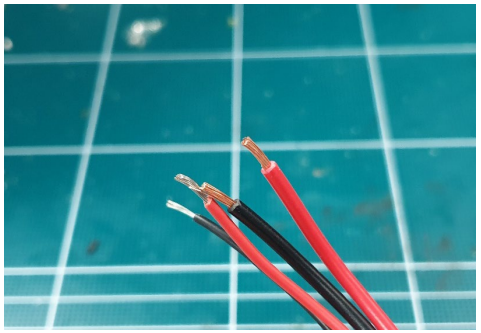


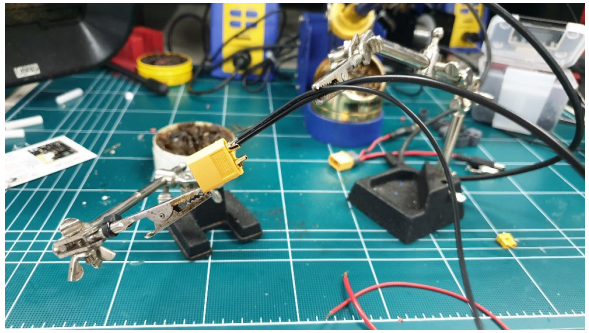
DC Jack

XT60 connector

Electronics

- **Wiring for power supply**
  - Small tutorial for wiring and sordering





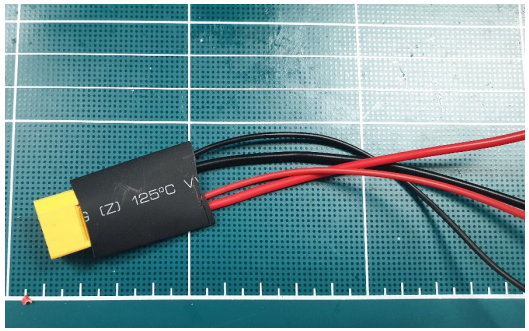
1. Pill the tips of wire using wire stripper

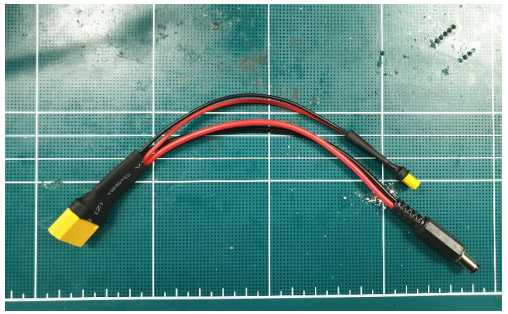
2. Smear soldering paste at the wire tips

3. Solder with helping hand

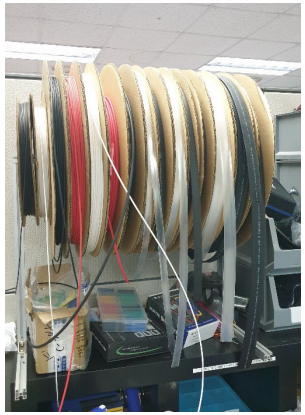
Electronics

- **Wiring for power supply**
  - Small tutorial for wiring and sordering





Done!



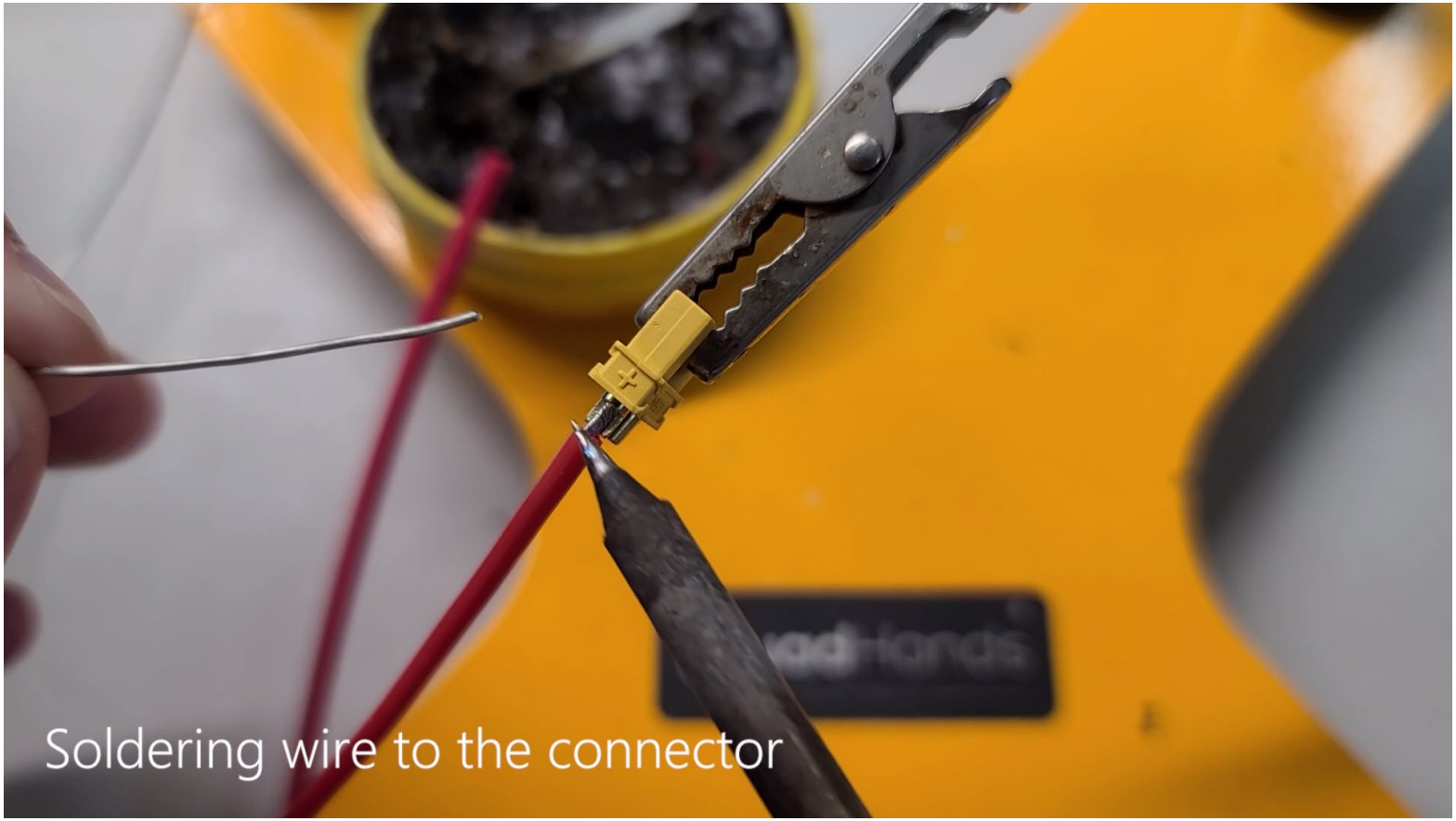
4. Cover the soldered part with heat shrink tube

5. Use a head gun for heat shrink tubing

Electronics

- **Wiring for power supply**
  - Watch the following reference video before starting soldering!

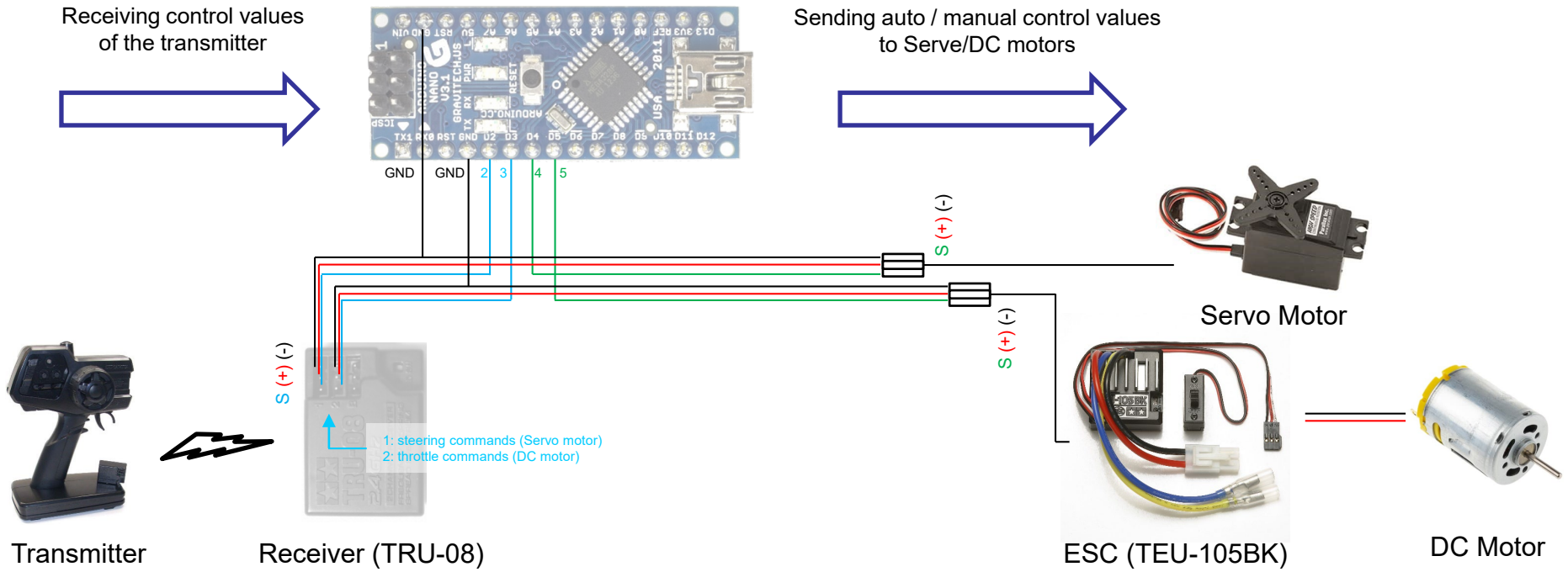
https://www.youtube.com/watch?v=aZLutrw3_g4



Electronics

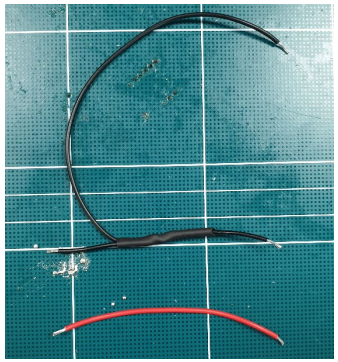
➢ Wiring between Arduino and Receiver

- ❑ For switching manual and auto control, the Arduino should be placed between **the wire** receiving the control value of the transmitter from the receiver and **the wire** sending to the Servo/DC motors .
- ❑ Arduino can read the control values from the receiver and selectively send the auto control (from Jetson, ROS) or manual control (from Transmitter) commands to the motors according to the '/auto_mode' ROS message (see the Software configuration PPT)

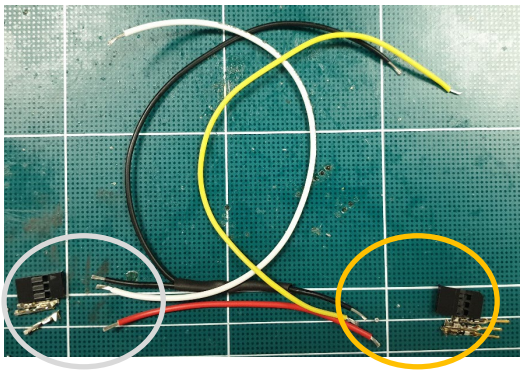


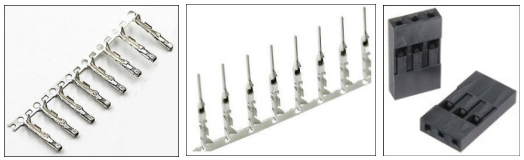
Electronics

- **Wiring between Arduino and Receiver**
  - We need two of the below wire set (for both Servo and DC motors).

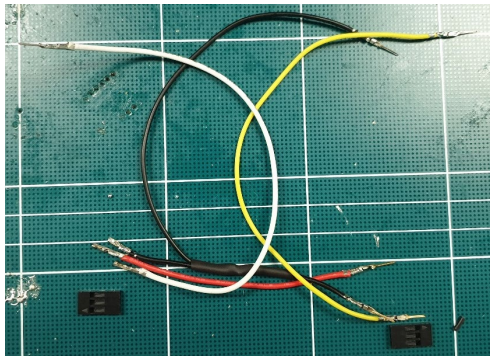


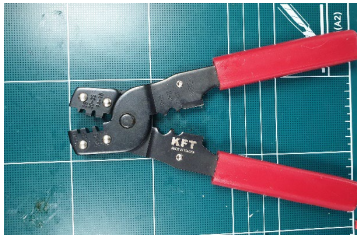
1. Wire for (+) and (-)





2. Wire for receiver and motor with Jumper wire cable housing Female and Male pin header connectors.





3. Use a climping tool for the pin connectors.

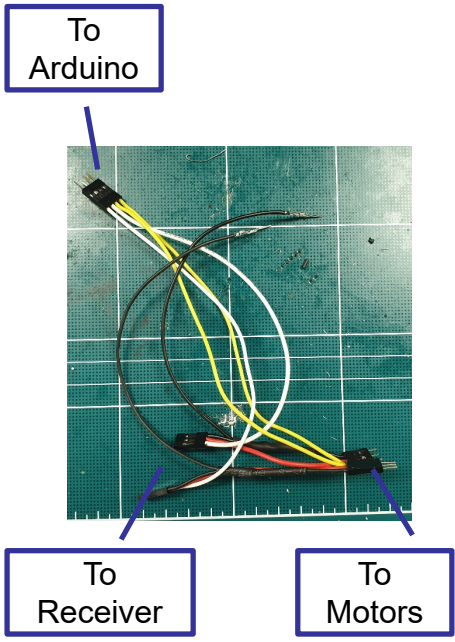
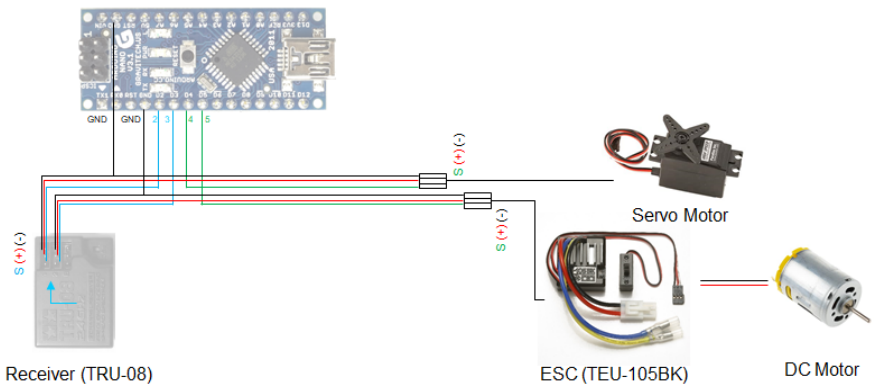
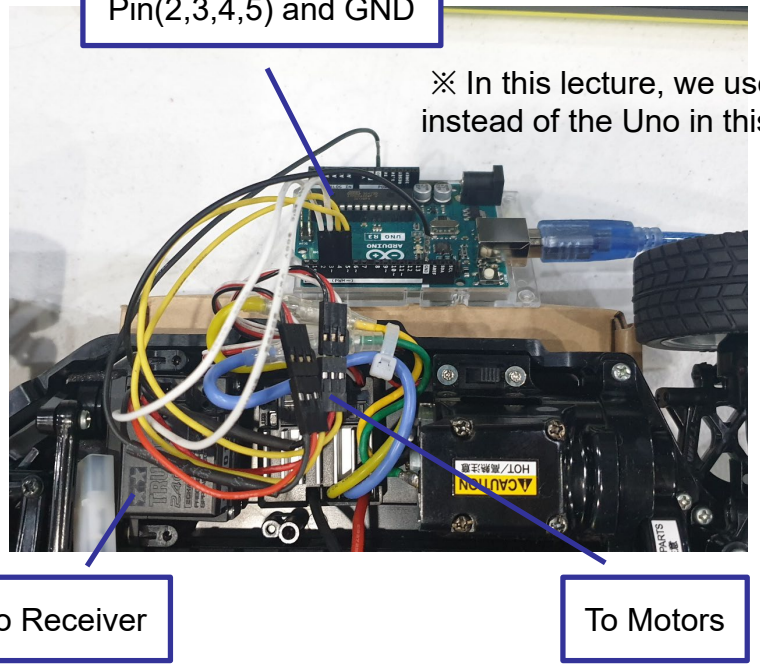


Image for two wire sets

Electronics

- **Wiring between Arduino and Receiver**
  - Connect the wire sets with the receiver and motors.





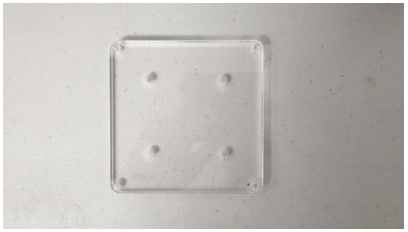
Hardware Configuration

Chassis

Chassis

- **Chassis**
  - Chassis is a structure for mounting the sensor and boards on a vehicle platform.
  - It has to be designed differently according to the platform shape.
  - We use lightweight, durable Acrylic plates.

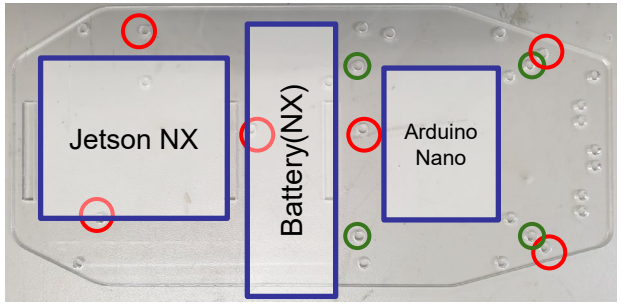
- **Build Chassis**
  - Setting up sensors, boards, battery placement on an acrylic plate.
  - Positioning holes for mounting electronics on the chassis.
  - Positioning holes for mounting chassis on the RC platform.
  - Design chassis using Solidworks & Cut the acrylic plate with laser cutter.
  - Mount the sensors, boards, battery on the chassis plate.
  - Drill holes for mounting the laser-cut plate on the bottom of the RC platform.
  - Tighten the chassis plate and RC platform using supports.



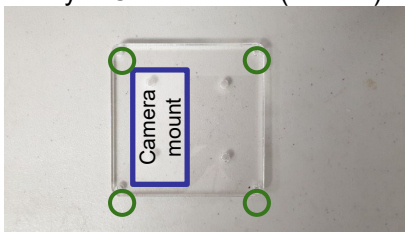
Acrylic Chassis Plates

3D Printed Mount (for Camera)

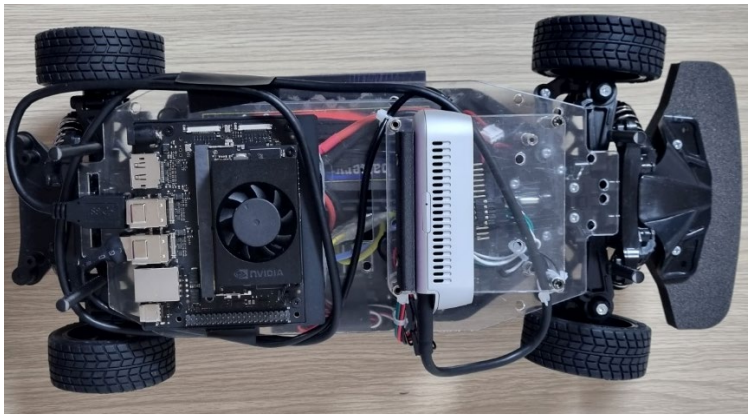
Chassis



Acrylic Chassis Plate (Bottom)



Acrylic Chassis Plate
(Middle, for Camera)



※ in the above figure, the battery (NX) has not been placed yet.

O: For mounting chassis on the RC platform.
O: For mounting the bottom and middle chassis plates.

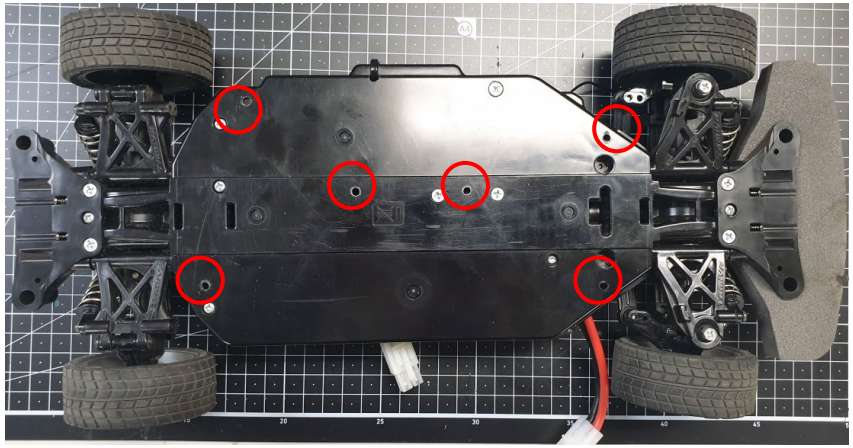
※ This is a reference placement, so feel free to arrange the devices in different ways as you see fit.

➢ Build Chassis

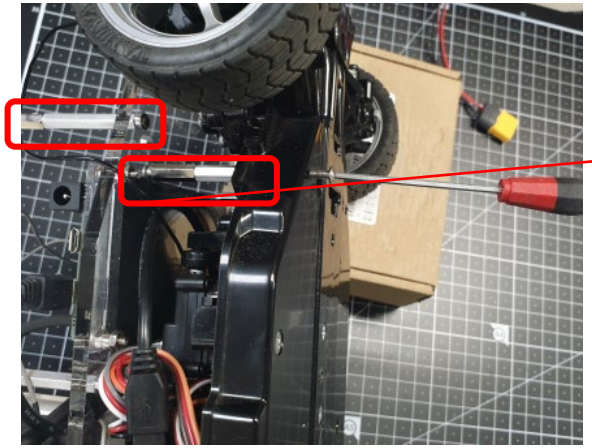
- ❑ Setting up sensors, boards, battery placement on an acrylic plate.
- ❑ Positioning holes for mounting electronics on the chassis.
- ❑ Positioning holes for mounting chassis on the RC platform.
- ❑ Design chassis using Solidworks & Cut the acrylic plate with laser cutter.
- ❑ Mount the sensors, boards, battery on the chassis plate.
- ❑ Drill holes for mounting the laser-cut plate on the bottom of the RC platform.
- ❑ Tighten the chassis plate and RC platform using supports.

3D Printed Mount (for Camera)

Chassis



Drill holes on the bottom of the RC platform



Tighten the chassis plate and RC platform

- **Build Chassis**
  - ❑ Setting up sensors, boards, battery placement on an acrylic plate.
  - ❑ Positioning holes for mounting electronics on the chassis.
  - ❑ Positioning holes for mounting chassis on the RC platform.
  - ❑ Design chassis using Solidworks & Cut the acrylic plate with laser cutter.
  - ❑ Mount the sensors, boards, battery on the chassis plate.
  - ❑ Drill holes for mounting the laser-cut plate on the bottom of the RC platform.
  - ❑ Tighten the chassis plate and RC platform using supports.

PCB support

PCB support metal

Notice

Notice

➢ Re-arrange Desktop PC Assignment

Teams						allocated PCs	
Team number	**member 1**	**member 2**	**member 3**	**member 4**	**member 5**		
1	김휘동	김건호	김민진	양지원	김태하	1	2
2	류현	박종은	양기표	이현석	정채윤		2
3	최서연	최현욱	차혜민	이라이언	Manit Kapoor	4	3
4	안승현	박성민	신혜리	안민형	박성혁		2
5	Hooman Khosravi	Elena Kuznetsova	Haard Shah	Saikhanbileg Sugar	Akhdan Dzaky Maulana	4	3
6	임진섭	이시우	천수범	이원재	조규석	4	3
7	이은성	박준호	이우빈	정우현	전영은	4	3
8	이세연	박문수	정재원	박준수		4	3
9	박성주	박지혁	채동준	Kochari Asgarov		3	

➢ Weekly Progress Report

- ❑ Each team is required to submit a progress report on their final project in KLMS (**every Friday 23:59**).
- ❑ Each week in class, we will guid you through the minimum milestones for the system configuration and design of your project. The following week, you will submit a progress report.

Weekly Progress Report

- **Hardware Configuration**
  - Assembly
  - Wiring

Figure of hardware configuration

- **Software Configuration**
  - Setting up Jetson NX
  - Installation of sensor interfaces
  - Installation of control interfaces

Figure of software configuration

Q & A