RumiCar

RumiCar Hands-On

Workshopping Self-Driving Car Algorithms

Lecturer: Rumika CHIBA
RumiCar Development Team
Project Owner

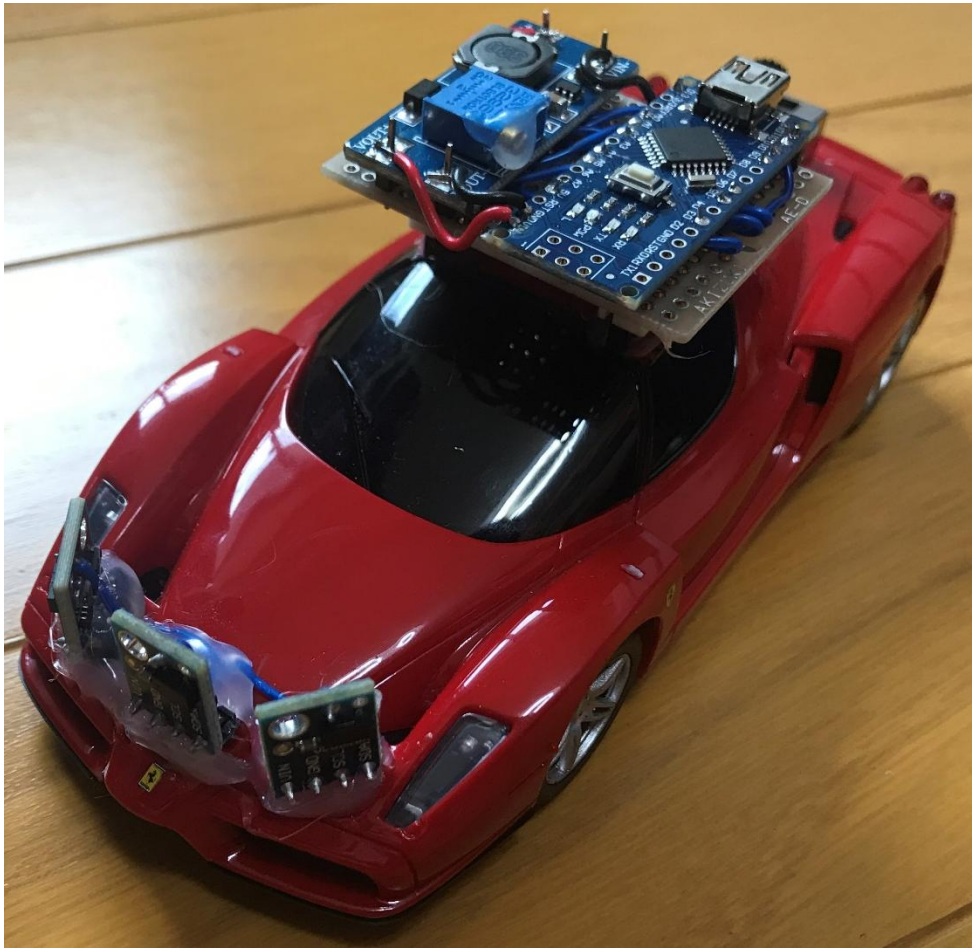
Index

The segments in blue will involve actual programming.

Résumé

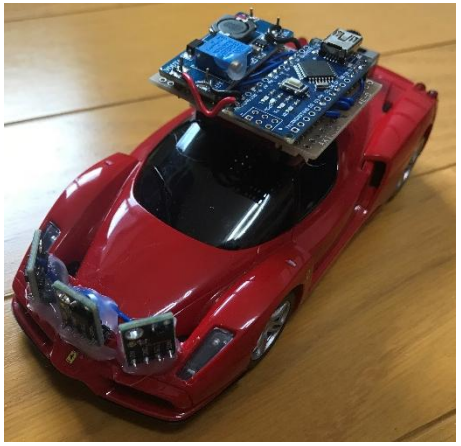
- How RumiCar Works
  - Sensor
  - Motor Driver
- Distance Measurement With Laser Sensors
- Motor Driver Control
- Basics of Autonomous Driving
- The Merits of Functionalization When Developing Programs

Introducing the RumiCar



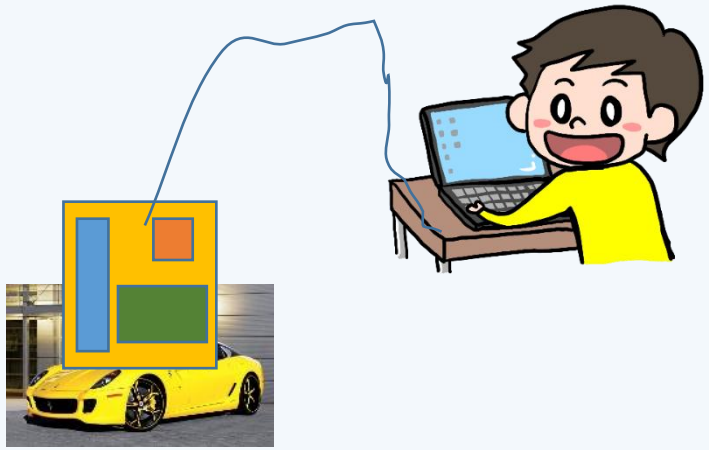
RumiCar is...

- A platform for developing autonomous driving programs by combining multiple car bodies and various types of compute modules.
- Equipped with a laser ranging module that can measure the distance to obstacles.
- Able to control motor rotation to alter vehicle speed.



Programming!

Use your own codes!



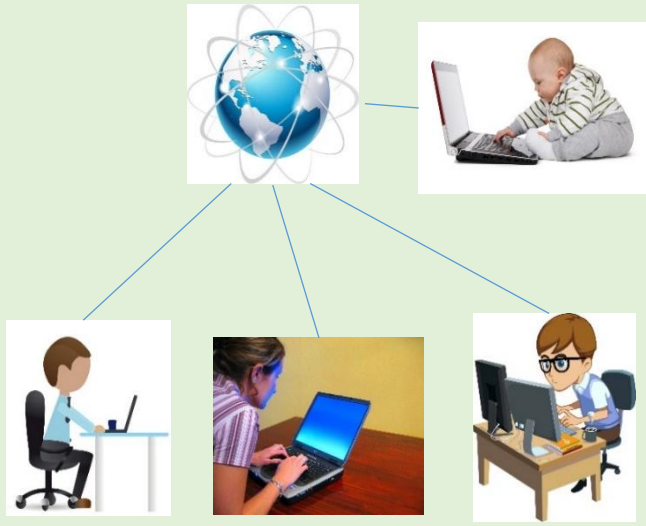
- RumiCar can be programmed and operated with your own computers!
- Surprise your friends with your codes!
- Maybe even AI??

A Worldwide Community

Currently in Development

Download and exchange programs

- Upload programs to our server
- Download interesting programs to your own RumiCar
- Communicate with peers worldwide



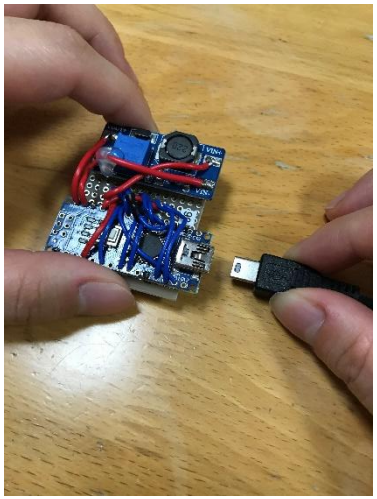
Learning Together

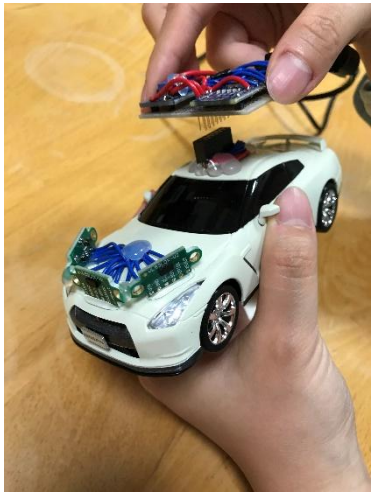
Currently in Development

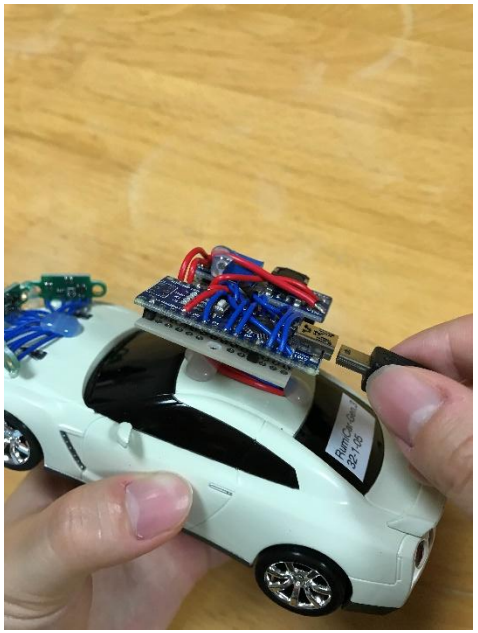
- Building the RumiCar
- Building Compute Modules
- Writing Programs

Preparation

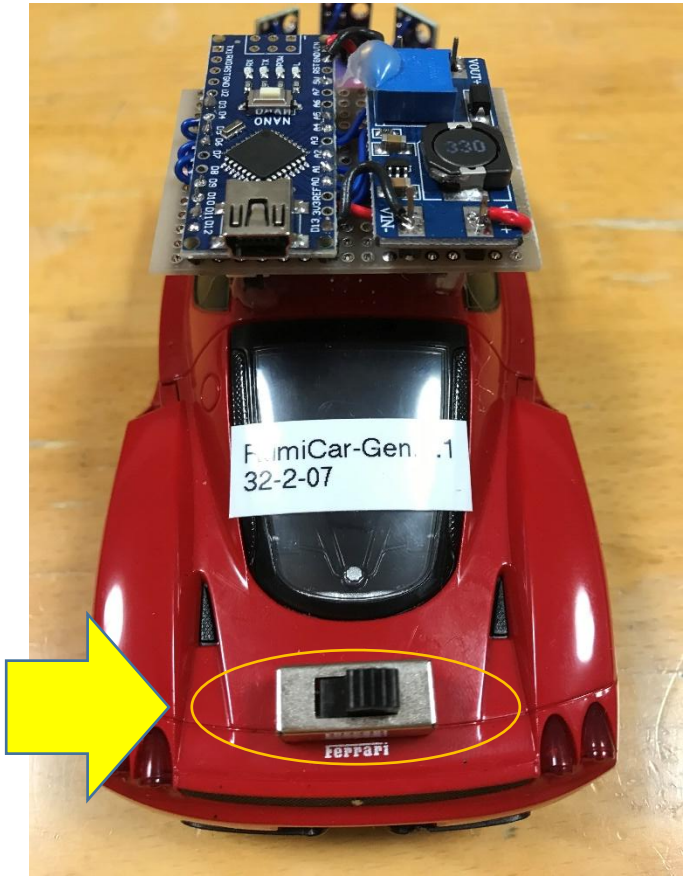
Cable Insertion / Removal







Configuring Controls



- Some models can toggle manual controls and computer controls.

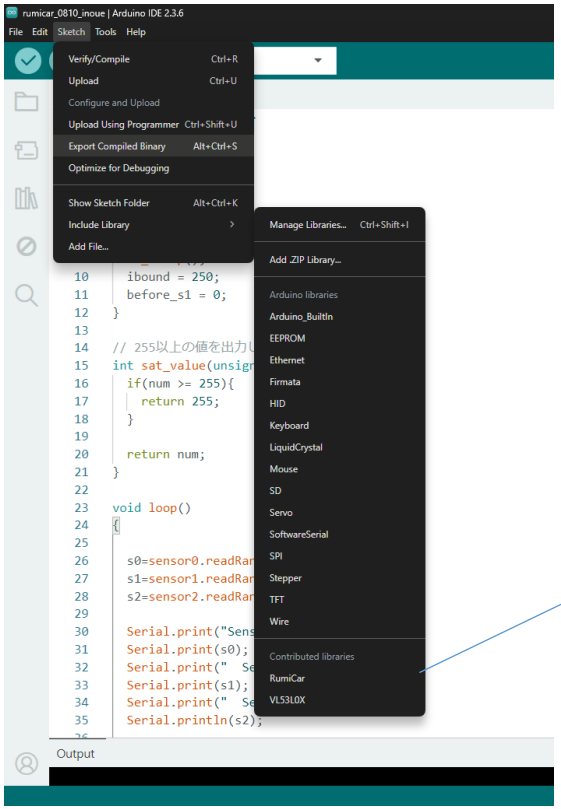
- Right for CM（Arduino / ESP32）controls
- Left for RC

Cable Insertion / Removal

- Connect USB cable to Compute Module
- Connect CM to car-body
- Don’t connect or disconnect USB to CM while CM is attached to car;
- CM connecter may break under excessive force

Install RumiCar Library (Confirm)

- Check Sketch > Include Library

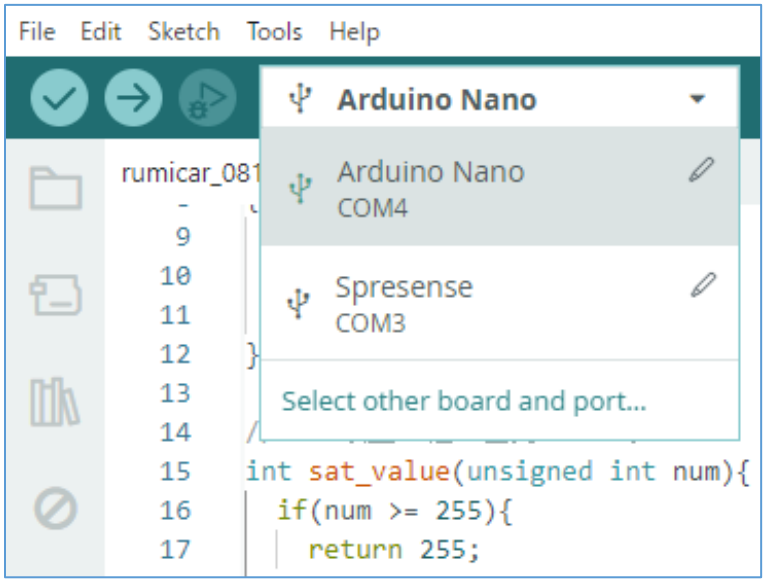


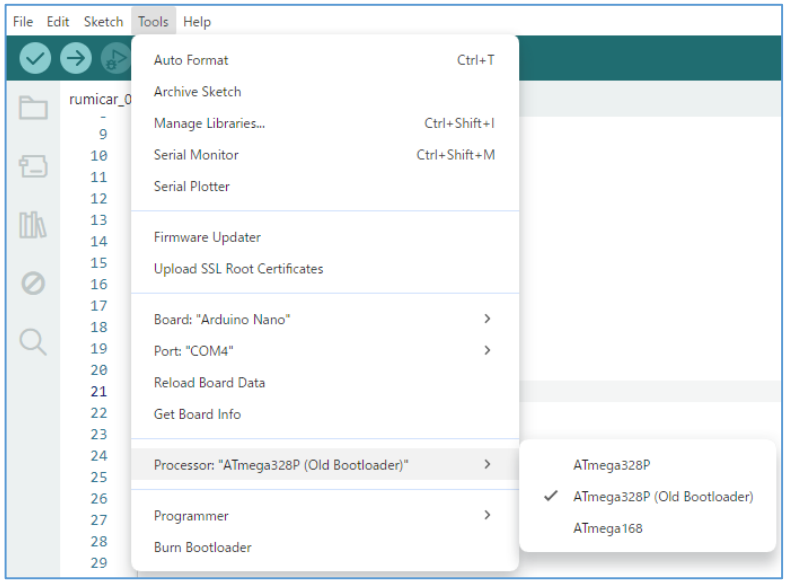
Incorrect
VL53L0X installed
RumiCar absent

Correct
VL53L0X and RumiCar both installed

Arduino IDE Settings

- Board and Serial Port: Arduino Nano
- Processor：
  - New Arduino Nano CM (Type-C): ATmega328P
  - Older Arduino Nano CM (Micro-B, Mini-B, etc.): ATmega328P (Old Bootloader)





Confirm Settings

Item	Setting	Notes
Board	Arduino Nano	Arduino UNO not compatible
Processor	New Arduino Nano CM (Type-C)	Select ATmega328P
	Older Arduino Nano CM (Micro-B, Mini-B, etc.)	Select ATmega328P (Old Bootloader)
Serial Port	Shown in Device Manager	May change if board is exchanged
Writing Device	USBasp	Possibly changed from default by user
Serial Port Baud Rate	9600bps	Arduino Nano required settings: 9600bps Check in Arduino IDE's Serial Monitor

Distance Measurement

Laser Ranging Module

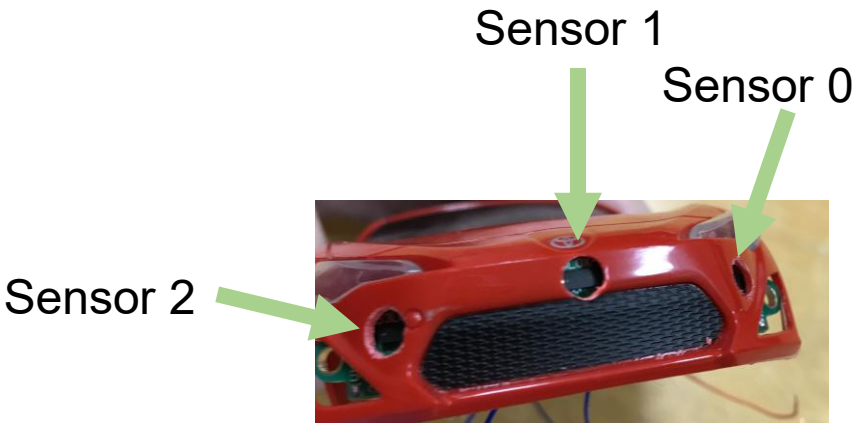
Laser Ranging Module

- 3 modules in front of vehicle

Left: 0

Center: 1

Right: 2



Ranging Methods

- Here is an example of a command the ranging modules use to obtain measurements. The results are an integer value (int) in millimetres:

  readRangeSingleMillimeters()

- This is the command to obtain the measurements of sensor 1:

  i = sensor1.readRangeSingleMillimeters();

Exercise-1

Ranging

Exercise-1.1

Ranging with the Central Sensor

Exercise-1.1
Ranging with the Central Sensor (Prep)

- Turn RumiCar's power off
- Remove CM
- Connect CM to USB Cable
- Connect USB cable to computer
- Reattach CM to RumiCar

Exercise-1.1
Ranging with the Central Sensor

- In Arduino IDE, open ”Exercise”
- At the bottom, add this code to display measurements

```
225
226 void loop()
227 {
228
229 }
```

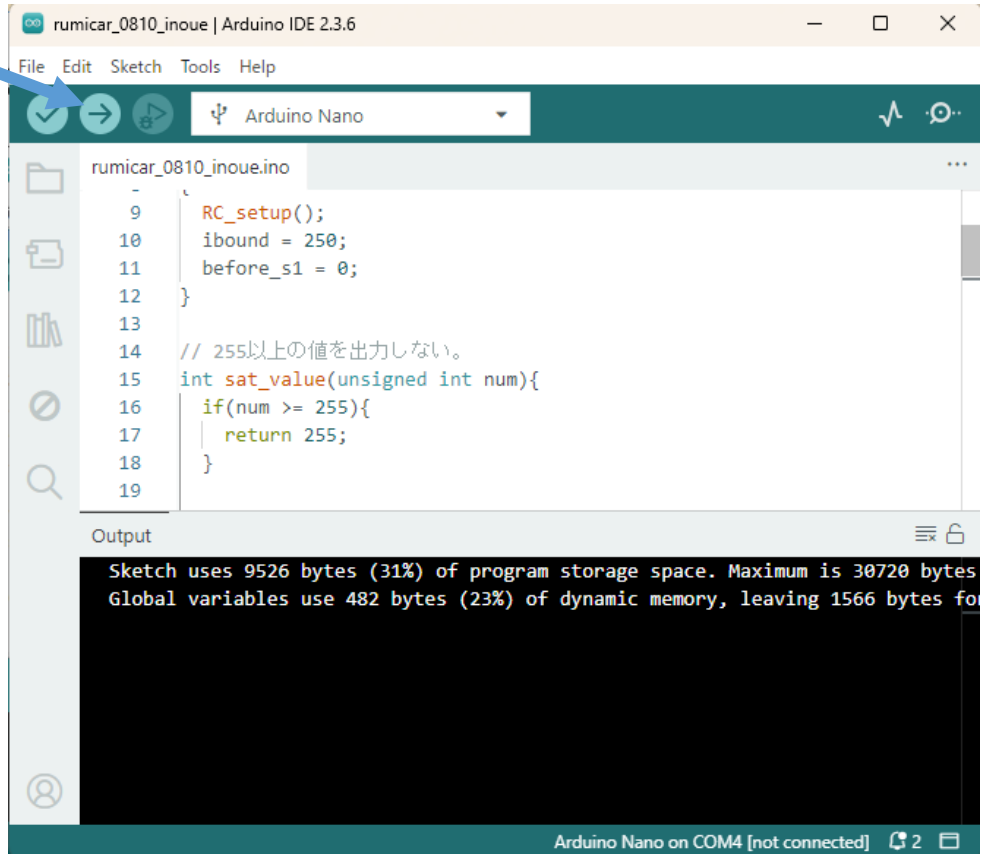
```
225
226 void loop()
227 {
228 Serial.println(sensor1.readRangeSingleMillimeters());
229 }
```

- The Serial.print and Serial.println commands displays data on the serial monitor
- Serial.print had no line breaks
- Serial.println starts a new line

Exercise-1.1
Ranging with the Central Sensor

- Compile / Write to Board

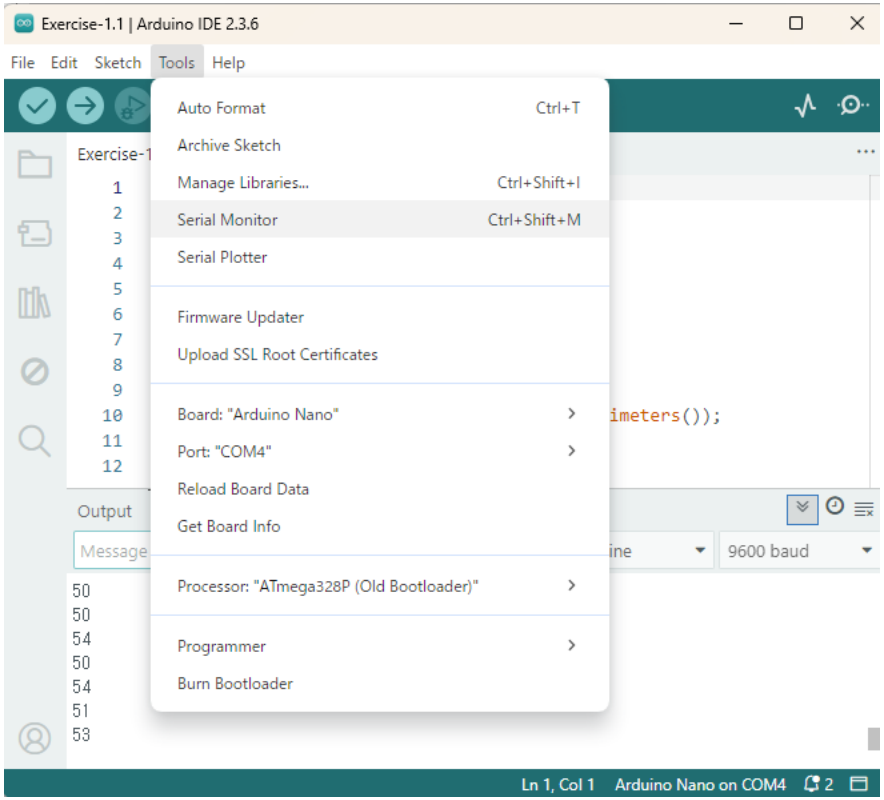
Click



Exercise-1.1
Ranging with the Central Sensor (Cont'd)

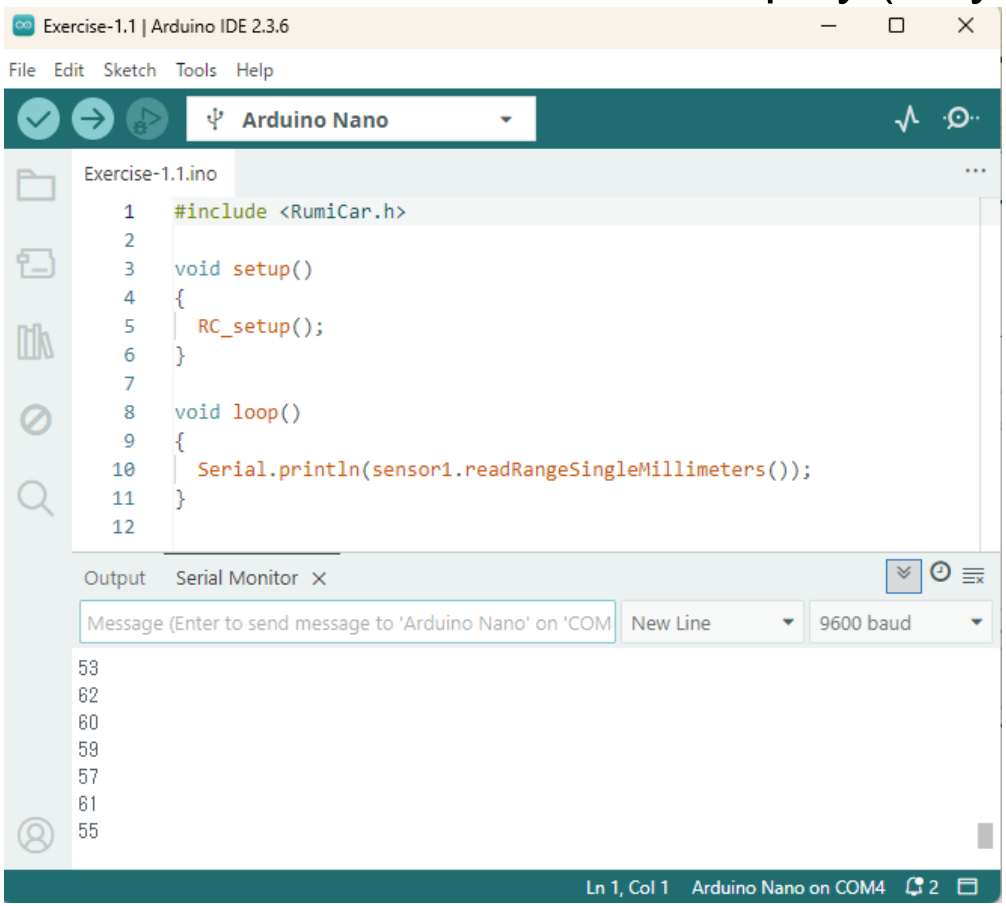
- Display values on serial monitor
  - Tools -> Serial Monitor

Or just click this



Exercise-1.1
Ranging with the Central Sensor (Cont'd)

- Check Baud rate if there are issues with the display (they may be set to 9600 bps)



Exercise-1.1
Ranging with the Central Sensor (Cont'd)

- If issues with the display persist, check the instructions under "Configure Settings"

Exercise-1.2

Ranging with 3 Sensors

Exercise-1.2
Ranging with 3 Sensors

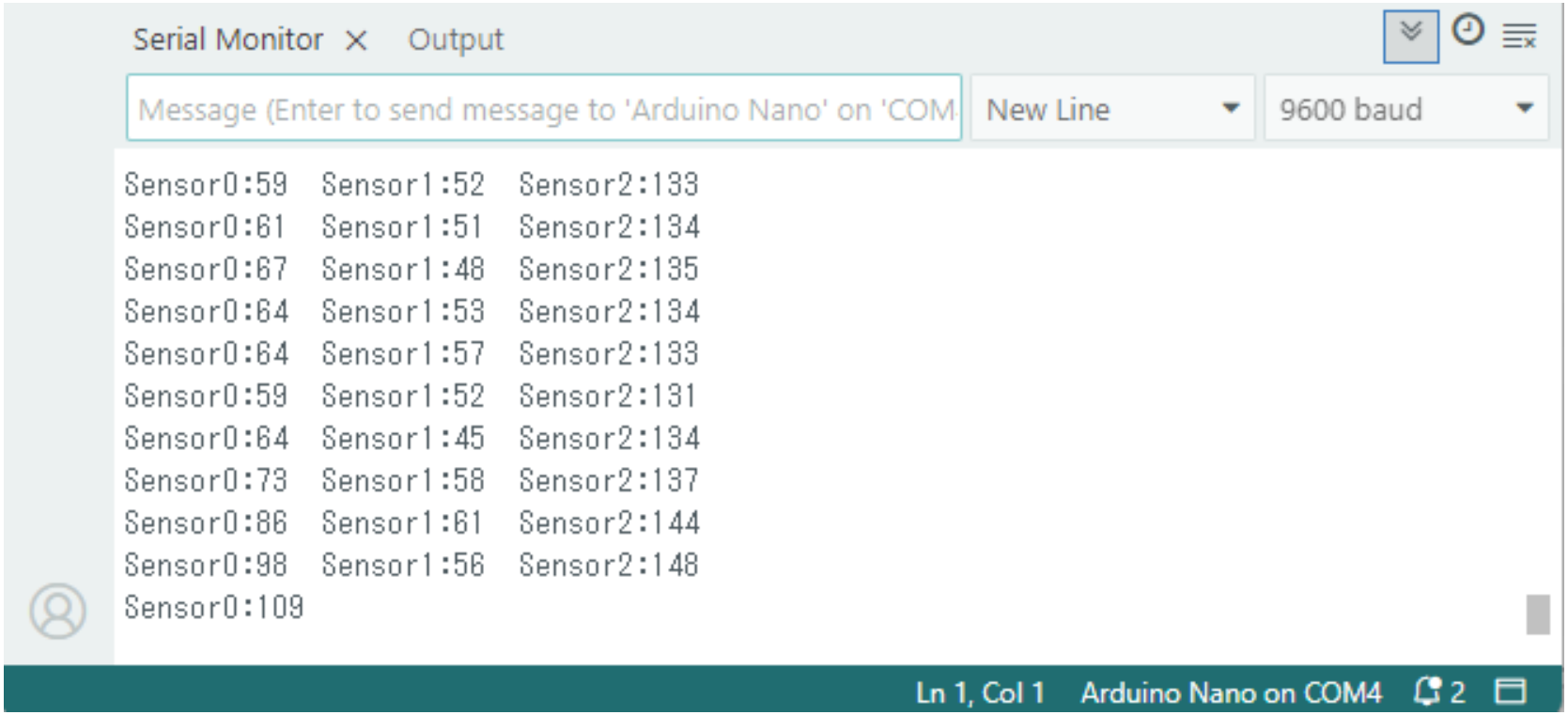
- Open ”Exercise-1.2”

Sensor	Sensor location
Sensor0	Left
Sensor1	Center
Sensor2	Right

```
123
124 void loop()
125 {
126   Serial.print("Sensor0:");
127   Serial.print(sensor0.readRangeSingleMillimeters());
128   Serial.print("  Sensor1:");
129   Serial.print(sensor1.readRangeSingleMillimeters());
130   Serial.print("  Sensor2:");
131   Serial.println(sensor2.readRangeSingleMillimeters());
132 }
```

Exercise-1.2
Ranging with 3 Sensors (Cont'd)

- Display measurement values on Serial Monitor
  Tools -> Serial Monitor

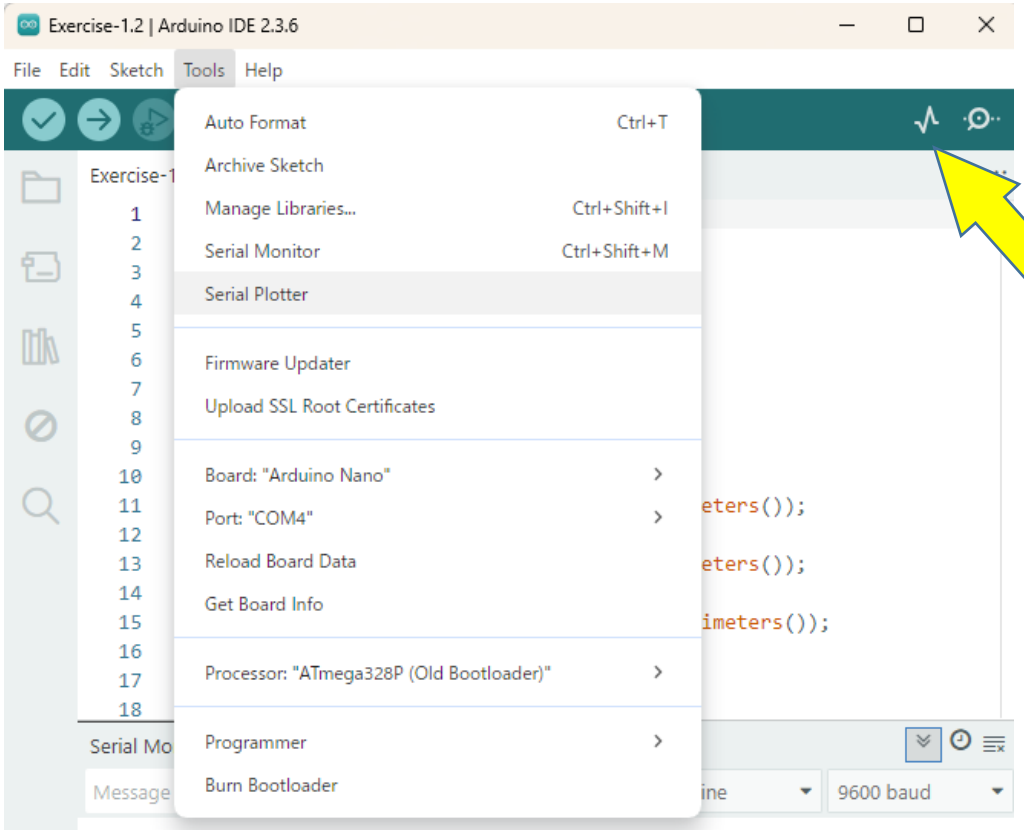


Exercise-1.3

Using the Serial Plotter

Exercise-1.3
Using the Serial Plotter

- This exercise will still use the programs from Exercise-1.2
- Tools -> Serial Plotter



Or just click this

Motor Control

Moving the Handle and Driving

Exercise-2

Motor Control

Exercise-2.1

Steering

Exercise-2.1
Steering (Prep)

- Turn RumiCar's power off
- Remove the CM
- Connect the CM to USB cable
- Connect USB cable to computer
- Open file "Exercise-2.1"

Exercise-2.1
Steering (Program)

```
210
211 void loop()
212 {
213   RC_steer(RIGHT);
214   delay(500);
215   RC_steer(LEFT);
216   delay(500);
217 }
```

- The “delay” command makes the car wait for the specified time
- The delay parameters are measured in milliseconds
- This program holds the steering wheel in one direction for 0.5 seconds (500 milliseconds).

- Steer right
- Keep steering right for 0.5 seconds
- Steer left
- Keep steering left for 0.5 seconds

Repeat

Exercise-2.1
Steering (Test)

- Compile and write program to CM
- Detach USB cable from CM
- Attach CM to RumiCar
- Turn RumiCar's power on
- Is the handle going left and right?

Exercise-2.2

Speed Control

Exercise-2.2
Speed Control (Prep)

- Turn RumiCar's power off
- Remove the CM
- Connect the CM to USB cable
- Connect USB cable to computer
- Open file "Exercise-2.2"

Exercise-2.2
Speed Control (Program)

```
210
211 void loop()
212 {
213   RC_drive(FORWARD,255);
214   delay(500);
215   RC_drive(FORWARD,200);
216   delay(500);
217   RC_drive(FORWARD,150);
218   delay(500);
219 }
```

- RC_drive's second argument i.e. 255, 200, 150 are PWM values
- 255 is the maximum value; 128 is approximately half of max.
- Lowering the values causes the wheels to turn slower

Exercise-2.2
Speed Control (Testing)

- Detach USB cable from CM
- Connect CM to RumiCar
- Instead of letting it run, keep holding RumiCar in your hands
- Turn RumiCar's power on
- Are the wheels turning differently?
- Gently place RumiCar onto the desk

Exercise-2.3

Forward and Reverse

Exercise-2.3
Forward and Reverse (Prep)

- Turn RumiCar's power off
- Remove the CM
- Connect the CM to USB cable
- Connect USB cable to computer
- Open file "Exercise-2.3"

Exercise-2.3
Forward and Reverse (Script)

```
210
211 void loop()
212 {
213   RC_drive(FORWARD,255);
214   delay(500);
215   RC_drive(REVERSE,255);
216   delay(500);
217 }
```

- Forward movement when RC_drive's first argument is FORWARD
- Backward if first argument is REVERSE

- Go forward
- Continue forward for 0.5 seconds
- Go backwards
- Continue backwards for 0.5 seconds

Repeat

Exercise-2.3
Forward and Reverse (Test)

- Compile and write program to CM
- Detach USB cable from CM
- Connect CM to RumiCar
- Instead of letting it run, keep holding RumiCar in your hands
- Turn RumiCar's power on
- Are the wheels continuously moving forward and backward ?
- Gently place RumiCar onto the desk

Exercise-2.3
Forward and Reverse (Advanced)

- Advanced Development Question: How can I avoid slipping as much as possible while repeating forward and reverse motion?

Exercise-2.4

Zig-Zagging Movement

Exercise-2.4
Zig-Zagging Movement (Prep)

- Turn RumiCar's power off
- Remove the CM
- Connect the CM to USB cable
- Connect USB cable to computer
- Open file "Exercise-2.4"

Exercise-2.4
Zig-Zagging Movement (Script)

```
210
211 void loop()
212 {
213   RC_drive(FORWARD,255);
214   RC_steer(RIGHT);
215   delay(500);
216   RC_steer(LEFT);
217   delay(500);
218 }
```

- Move forward
- Steer right
- Keep steering right for 0.5 seconds
- Steer left
- Keep steering left for 0.5 seconds

Repeat

Exercise-2.4
Zig-Zagging Movement (Test)

- Compile and write program to CM
- Detach USB cable from CM
- Connect CM to RumiCar
- Instead of letting it run, keep holding RumiCar in your hands
- Turn RumiCar's power on
- Are the wheels moving as predicted?
- Gently place RumiCar onto the desk

Autonomous Driving Basics

Sensor-Controlled Driving

Autonomous Driving Basics

- Autonomous driving combines preexisting data with information obtained from sensors for automatic and optimal driving.
- In this chapter, we will consider a simple, basic example.

Autonomous Driving Basics

- Establish the autonomous car is under the following conditions:
  - If an obstacle is detected in front of the vehicle while driving forward, the vehicle stops safely to avoid colliding with the obstacle.
  - If the vehicle detects an obstacle and stops, it resumes moving forward when the obstacle is gone.

Exercise-3

Developing an Autonomous Driving Program

Exercise-3.1

Stopping the Car Safely

Exercise-3.1
Stopping the Car Safely

- Establish the following simple conditions:
  - RumiCar's central sensor (Sensor1) detects obstacles, and moves forward when no obstacle is detected.
  - The car stops in front of the obstacle (e.g. 10cm away), and resumes moving forward when the obstacle is gone.

Exercise-3.1
RumiCar's Development Considerations

- Measurement Errors
  - In the sample program, the VL53L0X is set to high-speed measurement mode, which sacrifices some accuracy for high-speed measurement. Therefore, an error of 1~2cm will occur during distance measurement. Account for this margin when ranging.
- Processing Unmeasurable Results
  - Results of around 8190 show up when no laser reflection is obtained, i.e. no obstacle can be detected within the measurement range.

Exercise-3.1
RumiCar's Development Considerations

- Detecting the Ground
  - The sensor emits a conical laser. Because of the small size of the RumiCar, the sensors on the car are not well positioned and the cone-shaped laser beam will eventually hit the ground or floor before the distance measurement limit. This distance is about 30cm for RumiCar model 32. Therefore, the detection distance of an obstacle should be set at less than 30cm.
- Braking
  - Even if you order the car to stop or go backwards from a running state, it will not be able to stop immediately due to inertia (momentum). If possible, consider more efficient ways to slow down or stop.

Exercise-3.1
Summary of Programming Requirements

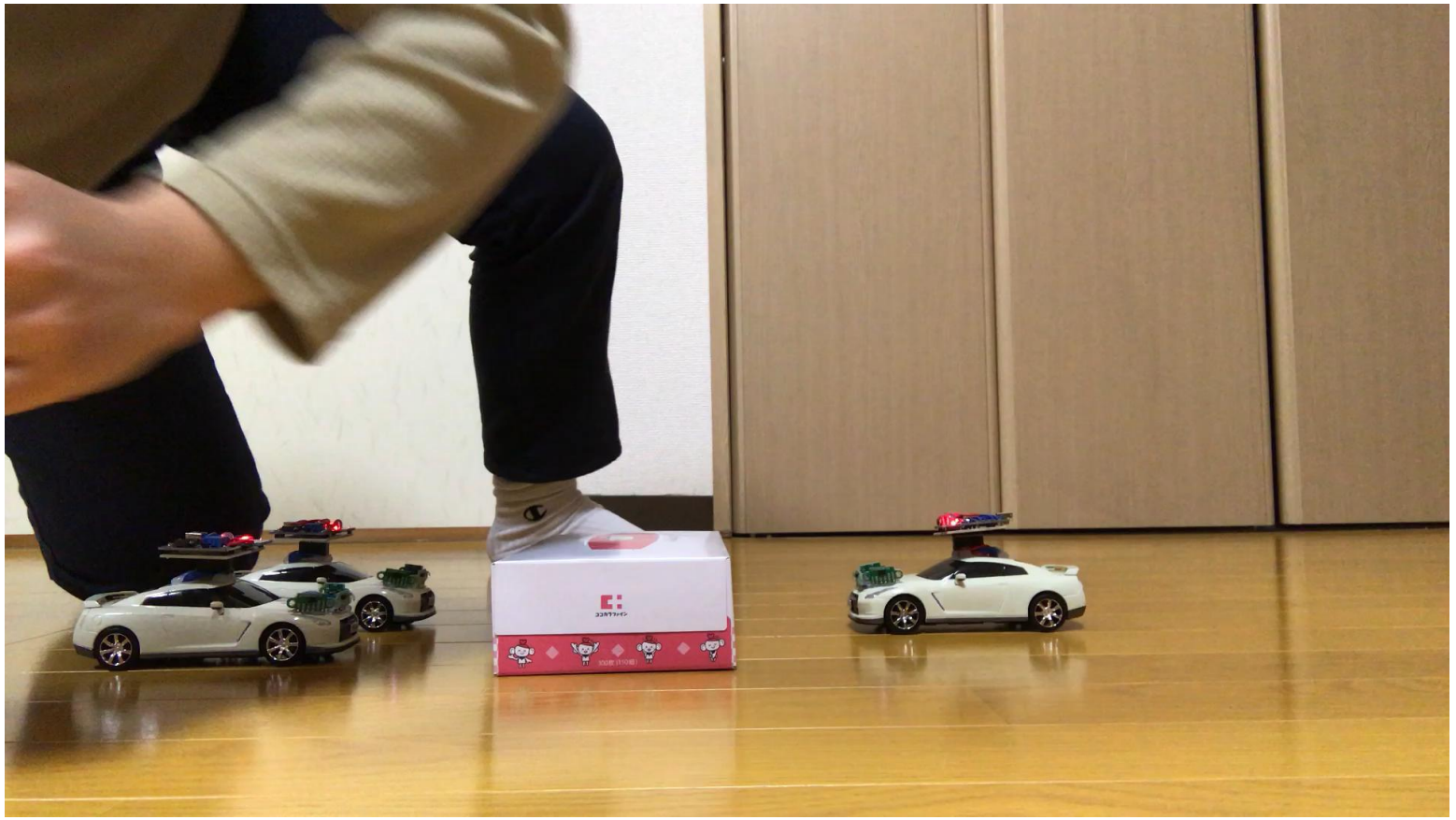
Item	Condition
Sensor used	Only use the central sensor (Sensor 1)
Operation	Stop when an obstacle is detected within the set distance
Set distance	Set to 10 cm. However, considering margin of error, the actual range is 10 ± 2 cm
Reliable distance	Within 30 cm
Value when measurement fails	8190, but be aware it may include errors

Exercise-3.1
Development Tips & Tricks

- If the condition for forward movement is simply that there are no obstacles in front of the car, RumiCar will drive an indefinite distance, e.g. right off a desk. Therefore you may want to add a condition such as only moving forward if there is an obstacle within 30 cm. This will reduce the risk of inadvertently damaging the RumiCar during development.

Exercise-3.1 https://youtu.be/95pc_4Wf14U
Safely Stopping the Car （Video）

Exercise-3.1 https://youtu.be/95pc_4Wf14U
Safely Stopping the Car （Video）



Exercise-3.1
Safely Stopping the Car (Prep)

- Turn RumiCar's power off
- Remove the CM
- Connect the CM to USB cable
- Connect USB cable to computer
- Open file "Exercise-3.1"

Exercise-3.1
Safely Stopping the Car (Prep)

```
210
211 void loop()
212 {
213 int ispeed = 255;
214 int idist1;
215 idist1=sensor1.readRangeSingleMillimeters();
216 if ( idist1 < 300 ){
217   if ( idist1 > 120 ){
218     RC_drive(FORWARD,ispeed);
219   }else if (idist1 < 80){
220     RC_drive(REVERSE,ispeed);;
221     }else{
222     RC_drive(BRAKE,ispeed);
223     }
224 }else{
225     RC_drive(FREE,ispeed);
226   }
227 }
```

- Is there an obstacle within 30 cm?
- (If not, do nothing)
- （If yes, so the following)
  - Obstacle is more than 12cm away->Forward
  - Obstacle is less than 8cm away->Back
  - Greater than 8cm and less than 12cm->Brake

Repeat

Exercise-3.1
Stopping the Car Safely (Test)

- Compile and write program to CM
- Detach USB cable from CM
- Connect CM to RumiCar
- Instead of letting it run, keep holding RumiCar in your hands
- Turn RumiCar's power on
- Place your hand closer to or away from the central sensor. Is RumiCar moving as predicted?
- Gently place RumiCar onto the desk

Exercise-3.2

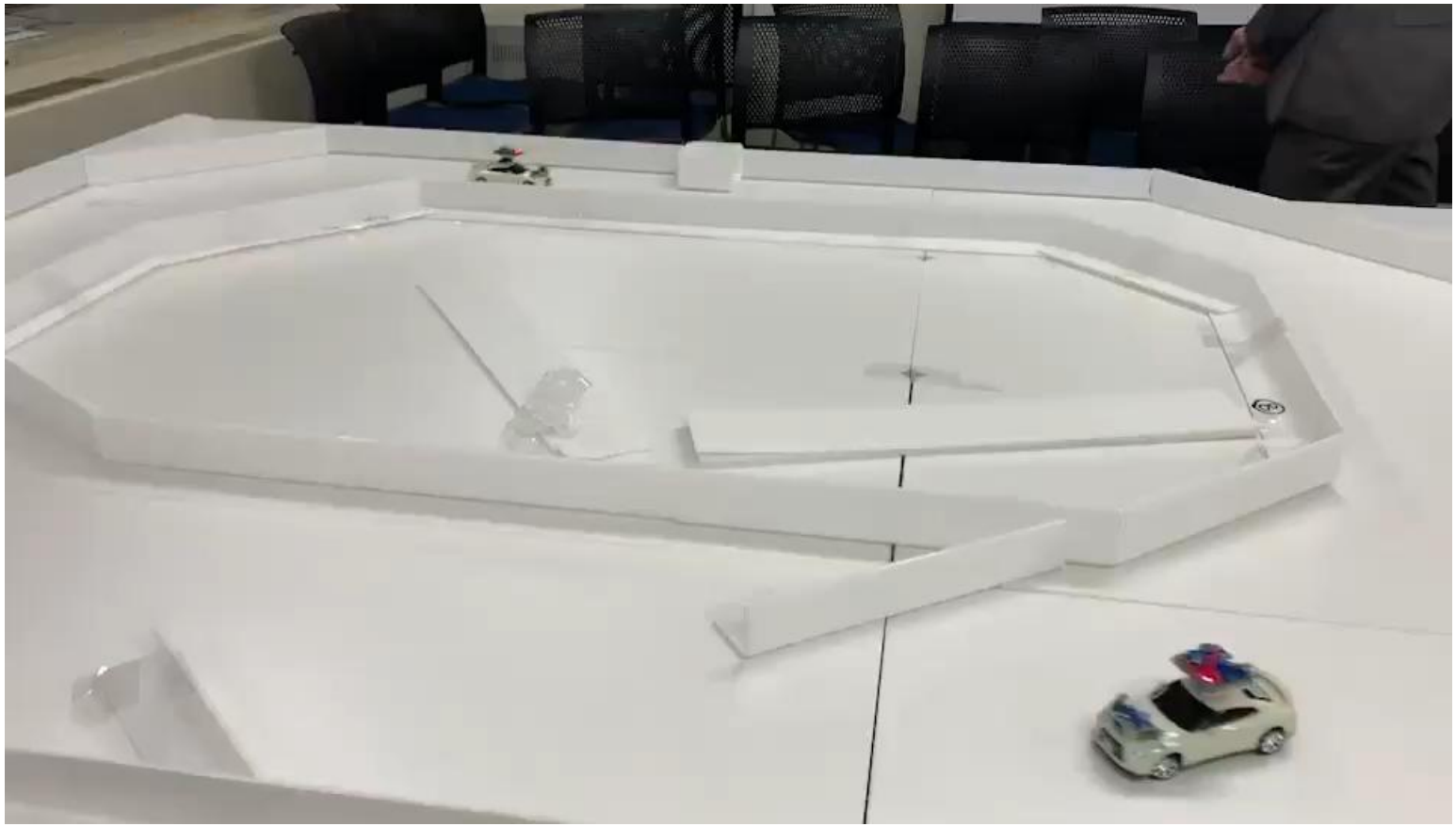
Driving in Urban Areas

Exercise-3.2
Driving in Urban Areas

- Set the following simple conditions:
  - Run on a path with walls on both sides, which represents an urban area.
  - Measure the distance between the two walls and drive in the center of the path.
  - Stop when within a certain distance (e.g. 10 cm) from car ahead, and resume forward movement when car moves farther away

Exercise-3.2 Driving in Urban Areas （Video）

https://youtu.be/jyq4Ph0mCDM



Exercise-3.2
Driving in Urban Areas (Prep)

- Turn RumiCar's power off
- Remove the CM
- Connect the CM to USB cable
- Connect USB cable to computer
- Open file "Exercise-3.2"

Exercise-3.2
Driving in Urban Areas (Script)

```
210
211 void loop()
212 {
213
214 int ibound =250;
215 int s0, s1, s2;
216 s0=sensor0.readRangeSingleMillimeters();
217 s1=sensor1.readRangeSingleMillimeters();
218 s2=sensor2.readRangeSingleMillimeters();
219
220 if(s1<100){
221   RC_drive(REVERSE,150);
222 }else if (s1<150){
223   RC_drive(FORWARD,150);
224 }else if (s1<250){
225   RC_drive(FORWARD,200);
226 }else{
227   RC_drive(FORWARD,255);
228 }
229  if(s0>s2){
230   RC_steer(LEFT);
231   }else{
232     RC_steer(RIGHT);
233   }
234 }
```

- Try to change the speed parameter (150, 200, 255)!
- Let's try changing the distance parameters (100, 150, 250)!
- What do you think will happen?

- Obstacle is less than 10cm->slow backward
- Obstacle is less than 15cm->slow forward
- Obstacle is less than 25cm->slowly forward
- Other than the above, full speed forward

- Compare the distance between the left and right walls.
- Steer away from the closer wall (towards the farther wall)

Exercise-3.2
Driving in Urban Areas (Testing)

- Compile and write program to CM
- Detach USB cable from CM
- Connect CM to RumiCar
- Instead of letting it run, keep holding RumiCar in your hands
- Turn RumiCar's power on
- Gently place RumiCar onto the course

Final Notes

RumiCar Info

RumiCar Info

- RumiCar Web Site
  - https://www.rumicar.com/
- Facebook Group
  - https://www.facebook.com/groups/rumicar
- RumiCar Team E-Mail
  - info@RumiCar.com
- YouTube
  - https://www.youtube.com/@RumiCar
- GitHub
  - https://github.com/RumiCar-group/RumiCar

Thank you for participating in the RumiCar Hand-on

We’re done!