



An Inexpensive Autonomous Mobile Robot for Undergraduate Education: Integration of Arduino and Hokuyo Laser Range Finders

Course Material

Yuki Ueyama

Dept. Mechanical Engineering, National Defense Academy of Japan

Course plan

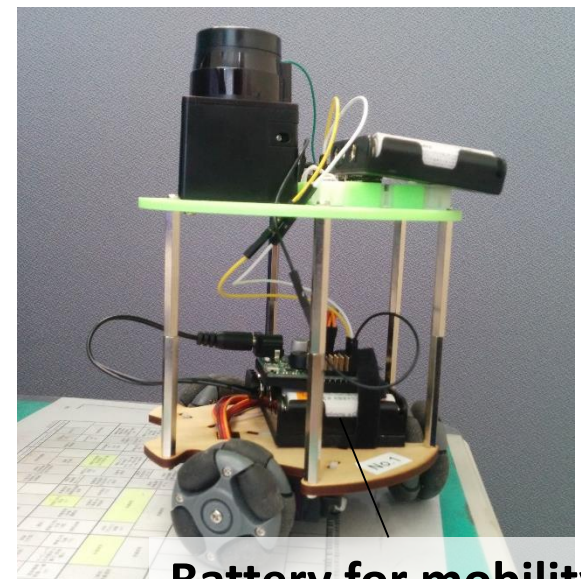
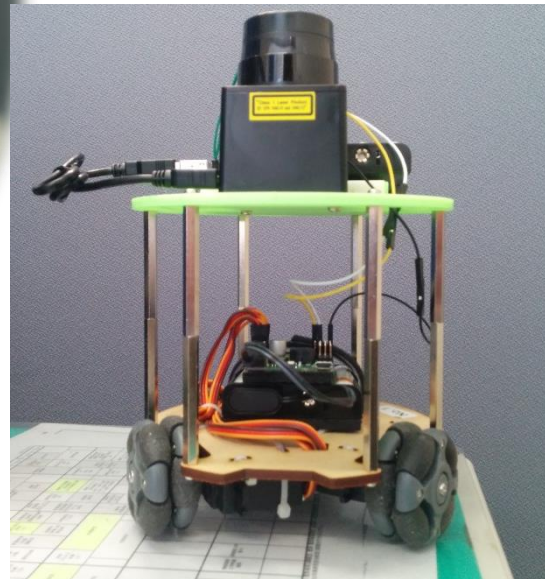
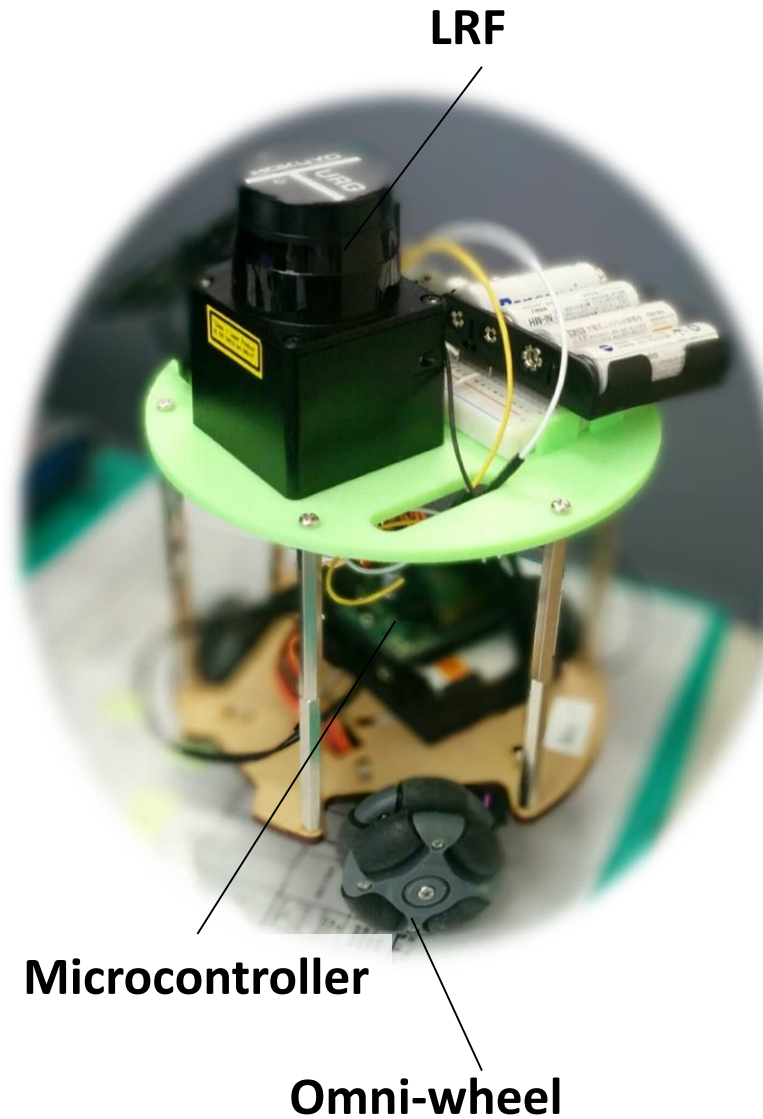
Week	Learning contents
1	Introduction
2~3	Programming basis
4	Feedback control using LRF
5~8	Project work
9	Preparing presentation
10	Presentation

Autonomous mobile vehicle

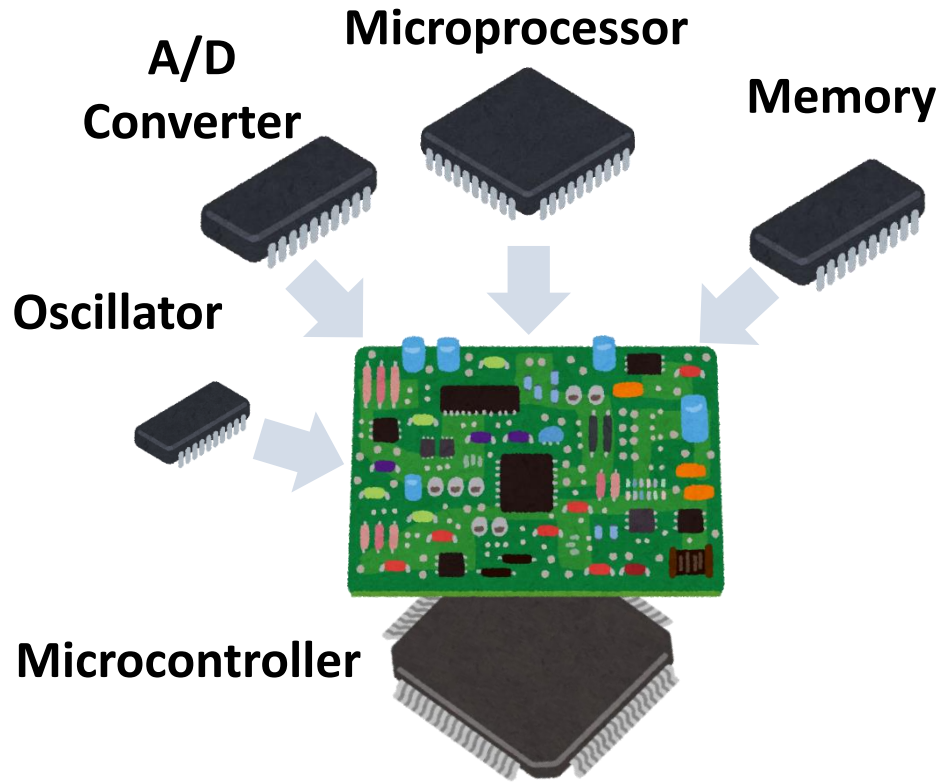


LiDAR (Light detection and ranging)
LRF (Laser range-finder)

Robot used in this course



Microcontroller



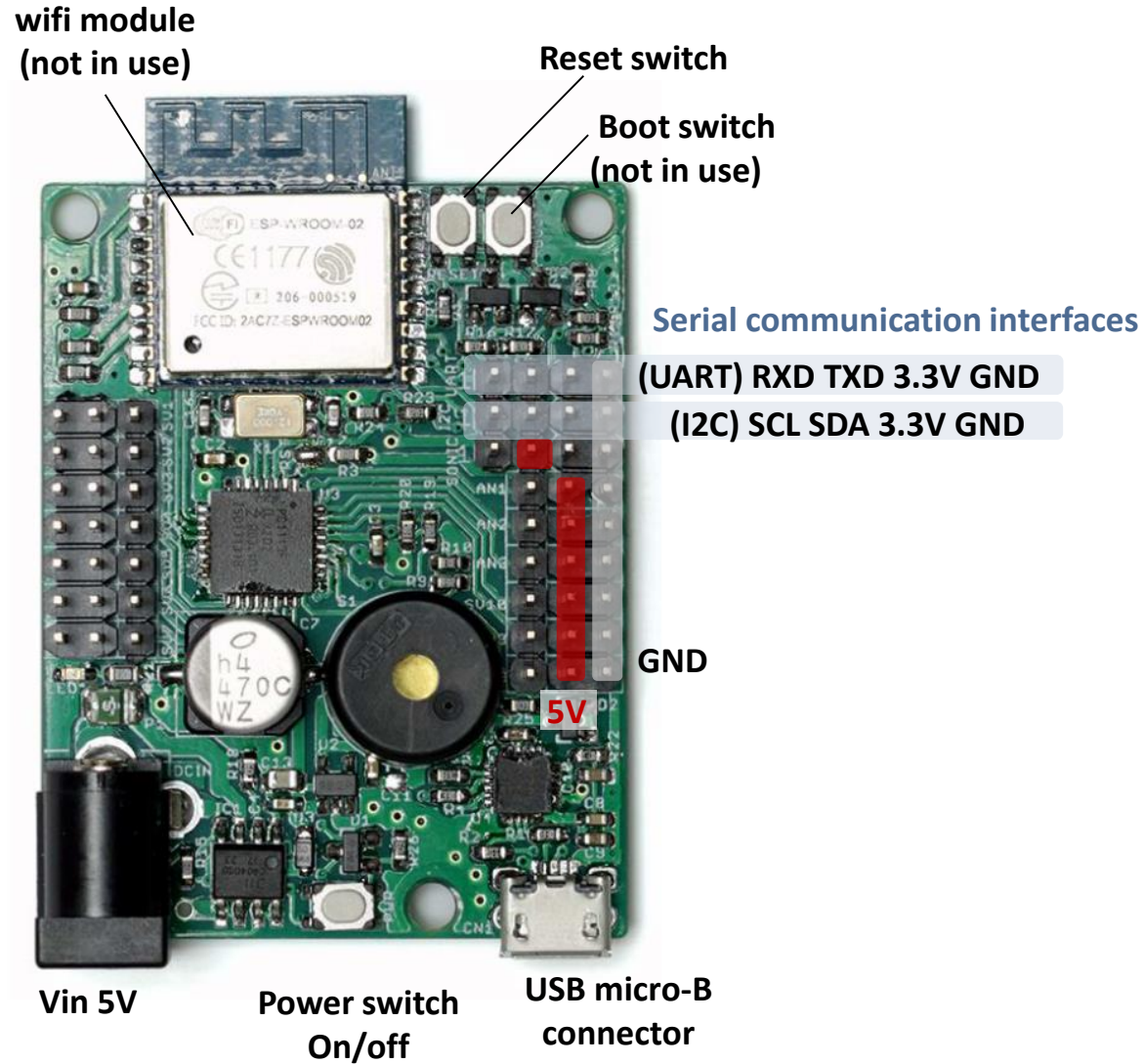
Arduino boards:



- Small computer integrated into an IC chip.
 - Processor, Memory, I/O
- Compact & low-cost.
- Embedded in appliances.

- Arduino is a family of single-board microcontrollers as open-source design.
- Easy to use for development and prototyping.


v-duino board (Compatible with Arduino)



Embedded programming

- Variable
 - Conditional statements (IF, ELSE)
 - Loop iteration (FOR, WHILE)
 - Function
 - Arrays
 - Servo motor
 - Serial communication
-

Arduino program (sketch) structure



```
SerialTest0 | Arduino 1.8.8
ファイル 編集 スケッチ ツール ヘルプ

SerialTest0 $

#include <vs-rc202.h>

void setup() {
  initLib();           //Initilize vs-rc202 library
  servoEnable(1, 1);    //Enable SV1 PWM
  setServoMovingTime(1000); //Set moving time to the target position
  Serial.begin(115200);  // 115200bpsでシリアルポートを開く
}

void loop() {
  setServoDeg(1, 0);    //Set SV1 servo target position
  moveServo();          //Start to move servo
  delay(1200);
  setServoDeg(1, 500);
  moveServo();
  delay(1200);
  /* シリアルモニタ
   に表示*/
  Serial.println("Hello World!");
}
```

Comments (not executed)

Comments (not executed)

Library (header file)

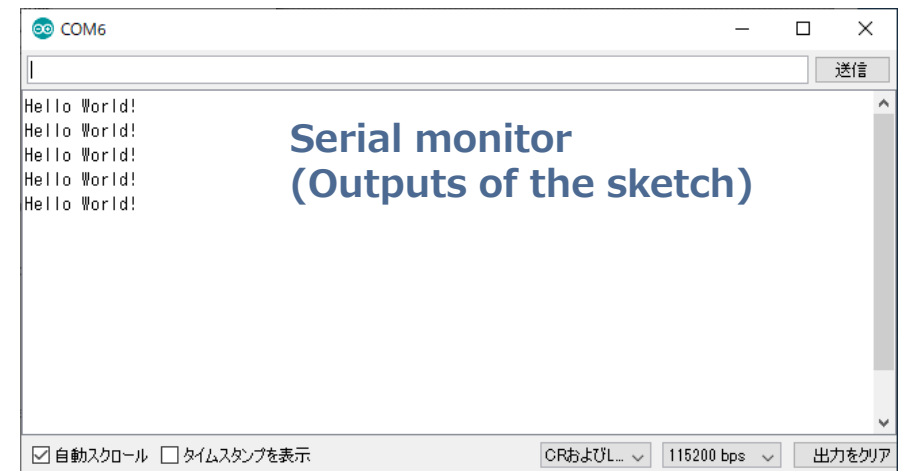
- Loads a set of functions for control motors.

setup() function

- Called when a sketch starts.

loop() function

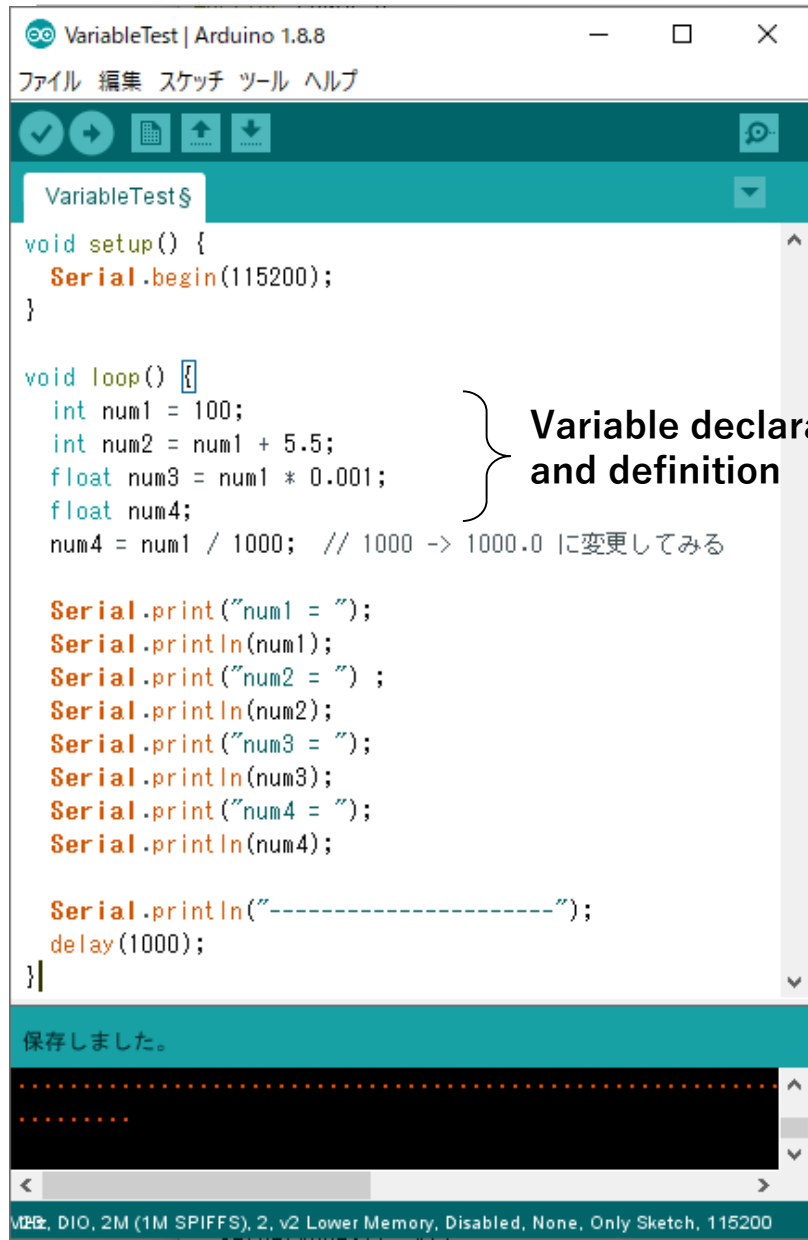
- Loops consecutively, after executing a setup() function.



```
COM6
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
```

Serial monitor
(Outputs of the sketch)

Variable



```
void setup() {  
  Serial.begin(115200);  
}  
  
void loop() {  
  int num1 = 100;  
  int num2 = num1 + 5.5;  
  float num3 = num1 * 0.001;  
  float num4;  
  num4 = num1 / 1000; // 1000 -> 1000.0 に変更してみる  
  
  Serial.print("num1 = ");  
  Serial.println(num1);  
  Serial.print("num2 = ");  
  Serial.println(num2);  
  Serial.print("num3 = ");  
  Serial.println(num3);  
  Serial.print("num4 = ");  
  Serial.println(num4);  
  
  Serial.println("-----");  
  delay(1000);  
}
```

Variable declaration
and definition

int num1 = 100;

Type Variable's name Value
(Not necessary for
the declaration)

Type	Description
void	Represents the absence of type.
char	A single octet (one byte).
int	Integer value.
float	Floating point value.

- A variable is a storage location paired with an associated symbolic name.
- Required to choose a type of variable according to data.

IF/ELSE 1/2

Arduino 1.8.8

ファイル 編集 スケッチ ツール ヘルプ

✓ → 📄 ⬆ ⬇

IfTest

```
void setup() {
  Serial.begin(115200);
}

int counter = 0;

void loop() {
  Serial.print(counter);

  if(counter < 10){
    Serial.println(": 10未満");
  }
  else if(counter == 10){
    Serial.println(": 10と等しい");
  }
  else{
    Serial.println(": 10より大きい");
  }

  counter = counter + 1;
  delay(1000);
}
```

ボードへの書き込みが完了しました。

1024, DIO, 2M (1M SPIFFS), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200

Global variable

- Declared outside of a loop() function.

```
if (condition A) {
    // Do stuff if the condition A is true.
}
else if (condition B){
    // Do stuff only if the condition A is false,
    // and the condition B is true.
}
else {
    // Do stuff if both of the conditions A and B are false.
}
```

Multiple conditions :

(and)

```
if (condition A && condition B) {
    // Do stuff if both of the conditions A and B are true.
}
```

(or)

```
if (condition A || condition B) {
    // Do stuff
    // if at lease one of the conditions A and B is true.
}
```

IF, ELSE 2/2

Conditional expressions :

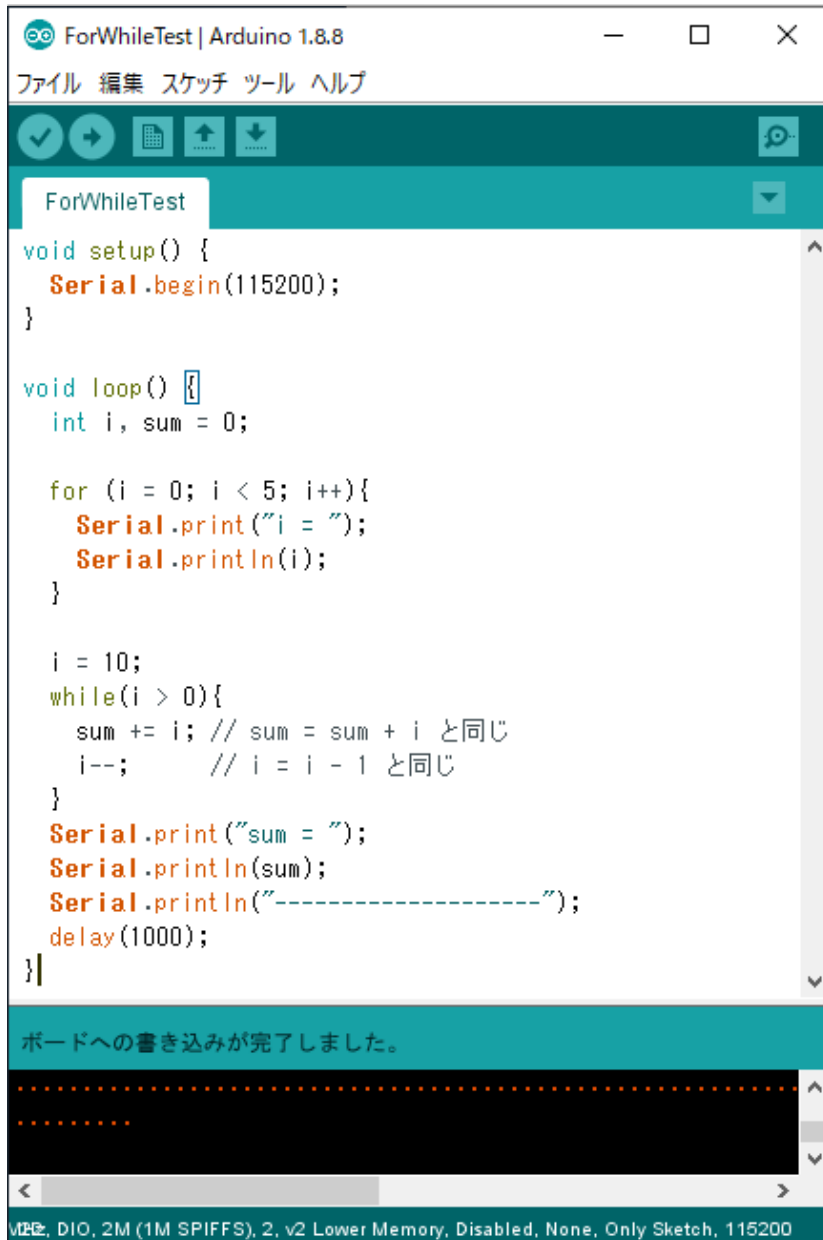
Exp	Description
A == B	Equal A to B.
A != B	Not equal A to B.
A < B	A is less than B.
A > B	A is grater than B.
A <= B	A is less than or equal to B.
A >= B	A is greater than or equal to B.

Switch case :

```
switch (variable) {  
    case value1 :  
        // Do stuff if variable = value1  
        break;  
    case value2 :  
        // Do stuff if variable = value2  
        break;  
    case value3 :  
        // Do stuff if variable = value3  
        break;  
        .  
        .  
        .  
    default :  
        // Do stuff if there is no match above cases  
}  

```

FOR, WHILE



The screenshot shows the Arduino IDE interface. The title bar reads 'ForWhileTest | Arduino 1.8.8'. The menu bar includes 'ファイル', '編集', 'スケッチ', 'ツール', and 'ヘルプ'. The toolbar contains icons for checking, running, serial monitor, and uploading. The sketch name 'ForWhileTest' is shown in a dropdown menu. The main text area contains the following code:

```
void setup() {  
  Serial.begin(115200);  
}  
  
void loop() {  
  int i, sum = 0;  
  
  for (i = 0; i < 5; i++){  
    Serial.print("i = ");  
    Serial.println(i);  
  }  
  
  i = 10;  
  while(i > 0){  
    sum += i; // sum = sum + i と同じ  
    i--;    // i = i - 1 と同じ  
  }  
  Serial.print("sum = ");  
  Serial.println(sum);  
  Serial.println("-----");  
  delay(1000);  
}
```

At the bottom, a status bar indicates 'ボードへの書き込みが完了しました。' (Upload completed). The bottom-most status bar shows hardware details: 'USB, DIO, 2M (1M SPIFFS), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200'.

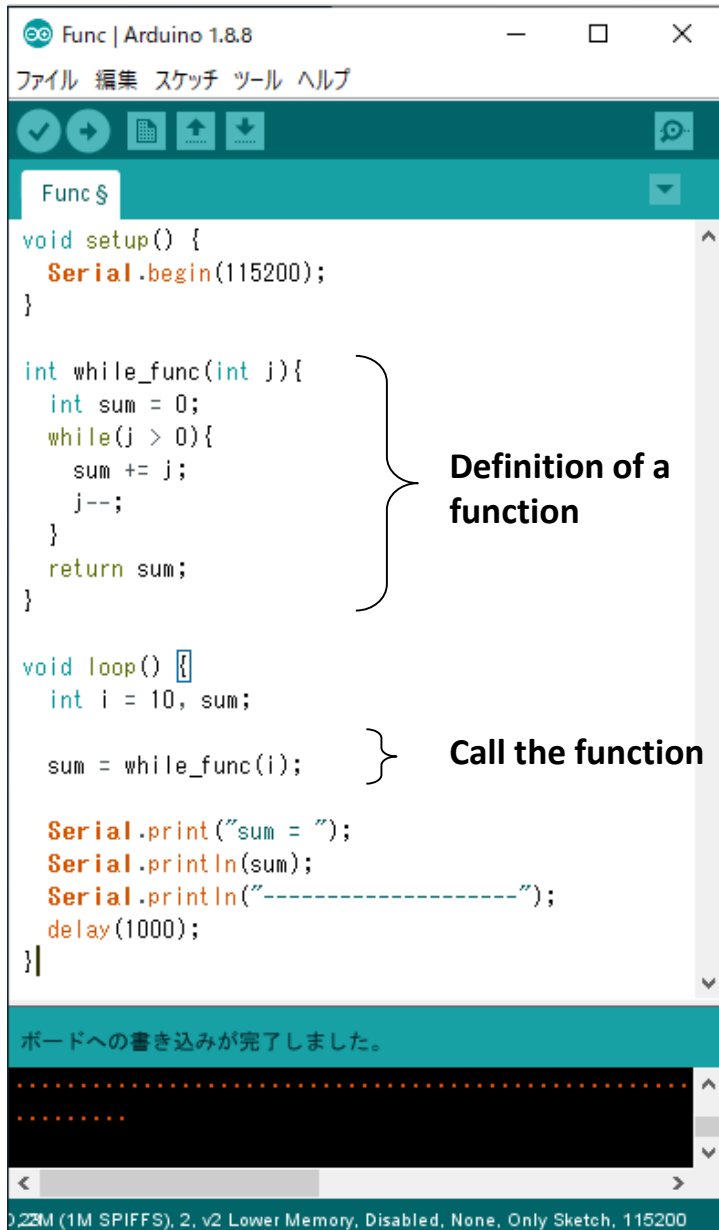
for loop :

```
for (initialization; condition ; increment) {  
    // Do stuff repeatedly for the condition is true.  
    .  
    .  
    .  
}
```

while loop :

```
while (condition A) {  
    // Do stuff repeatedly while the condition A is true.  
    .  
    .  
    .  
}
```

Function



The screenshot shows the Arduino IDE interface with a sketch named 'Func'. The code is as follows:

```
void setup() {  
  Serial.begin(115200);  
}  
  
int while_func(int j){  
  int sum = 0;  
  while(j > 0){  
    sum += j;  
    j--;  
  }  
  return sum;  
}  
  
void loop() {  
  int i = 10, sum;  
  
  sum = while_func(i);  
  
  Serial.print("sum = ");  
  Serial.println(sum);  
  Serial.println("-----");  
  delay(1000);  
}
```

Annotations in the image:

- A bracket on the right side of the `while_func` function definition is labeled "Definition of a function".
- A bracket on the right side of the `while_func(i)` call in the `loop` function is labeled "Call the function".

At the bottom, a status bar indicates "ボードへの書き込みが完了しました。" (Upload to board completed.) and a progress bar shows the upload progress.

Function with return value :

```
Return-type function (argument1, argument2, ...) {  
    // statements  
    .  
    .  
    .  
    return value;  
}
```

Function without return value :

```
void function (argument1, argument2 , ...) {  
    // statements  
    .  
    .  
    .  
}
```

Arrays

int num[3] = {1, 2, 3};

Type Array name[size] Initial values
(Required unique name) (Not necessary for declaration)

num[0]	num[1]	num[2]
1	2	3

If the size of the array is 3, the array is start from index 0 as num[0] to last to index 2 as num[2].

Two-dimensional array :

int num[2][2] = {{1, 2}, {3, 4}};

num[0][0]	1	2	num[0][1]
num[1][0]	3	4	num[1][1]

Servo motor using serial comm

```
SerialTest | Arduino 1.8.8
ファイル 編集 スケッチ ツール ヘルプ

SerialTest$

#include <vs-rc202.h>

void setup() {
  initLib();           //Initialize vs-rc202 library
  servoEnable(1, 1);    //Enable SV1 PWM
  setServoMovingTime(1000); //Set moving time to the target posit
  Serial.begin(115200);  // 115200bpsでシリアルポートを開く
}

int spd = 0;

void loop() {
  char sgn;           // + or -

  if (Serial.available() > 0) { // 受信したデータが存在する
    sgn = Serial.read();

    if (sgn == '+') { // 入力された文字が+のとき
      spd = spd + 100;
    } else if (sgn == '-') { // 入力された文字が-のとき
      spd = spd - 100;
    }
  }

  setServoDeg(1, spd); //回転速度をspdに設定
  moveServo();
  delay(1200);

  Serial.print("回転速度:"); // 受信データを表示
  Serial.println(spd);
}

630Hz, 40MHz, DIO, 2M (1M SPIFFS), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200
```

Press 'Enter' after input '+' or '-'.

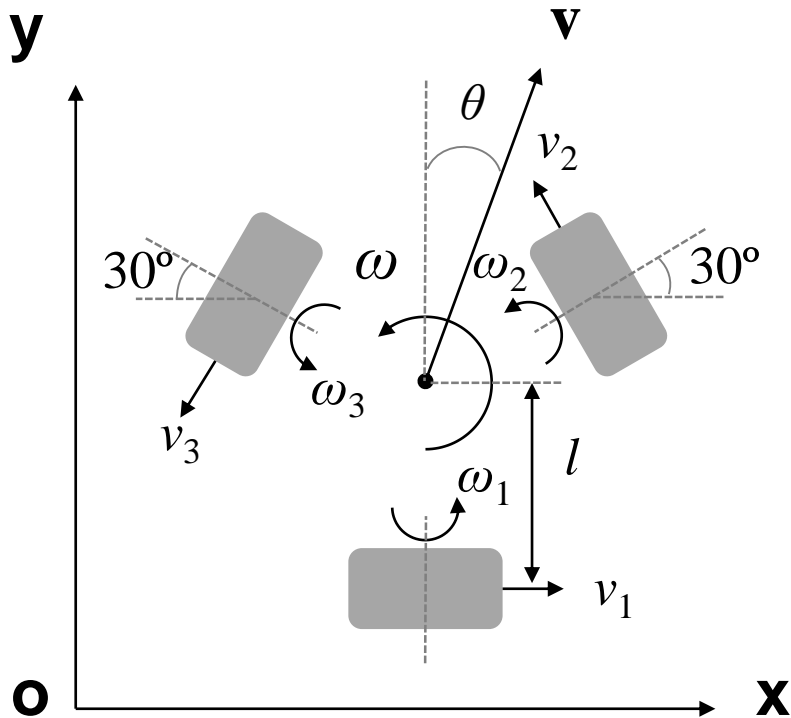
```
COM6
+| 送信

回転速度:0
回転速度:100
回転速度:100
回転速度:100
回転速度:100
回転速度:100
回転速度:0
回転速度:0
回転速度:0
回転速度:0
回転速度:0
回転速度:0
回転速度:0
回転速度:0

☒ 自動スクロール ☐ タイムスタンプを表示
CRおよびL... 115200 bps 出力をクリア
```

Kinematics of the mobile robot 1/2

Velocity in the body coordinates :



$$\mathbf{v} = \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} v \sin \theta \\ v \cos \theta \end{bmatrix}$$

Velocity of the wheels :

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} r\omega_1 \\ r\omega_2 \\ r\omega_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & l \\ -\sin 30^\circ & \cos 30^\circ & l \\ -\sin 30^\circ & -\cos 30^\circ & l \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix}$$

r : Radius of the wheel

Kinematics of the mobile robot 2/2

Relationship between the world and body coordinates :

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} \cos \phi & \sin \phi \\ -\sin \phi & \cos \phi \end{bmatrix} \begin{bmatrix} v_x^w \\ v_y^w \end{bmatrix}$$

ϕ : Heading direction of the robot

Velocity in the body coordinates

Velocity in the world coordinates

Velocity of the wheels :

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & l \\ -\sin 30^\circ & \cos 30^\circ & l \\ -\sin 30^\circ & -\cos 30^\circ & l \end{bmatrix} \begin{bmatrix} \cos \phi & \sin \phi & 0 \\ -\sin \phi & \cos \phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x^w \\ v_y^w \\ \omega \end{bmatrix}$$

$$= \begin{bmatrix} \cos \phi & \cos \phi & l \\ -\sin(\phi + 30^\circ) & \cos(\phi + 30^\circ) & l \\ \sin(\phi - 30^\circ) & -\cos(\phi - 30^\circ) & l \end{bmatrix} \begin{bmatrix} v_x^w \\ v_y^w \\ \omega \end{bmatrix}$$

where

$$\phi = \int \omega dt$$

Feedforward control

- Make a sketch to circle around an obstacle.

Feedback control using LRF

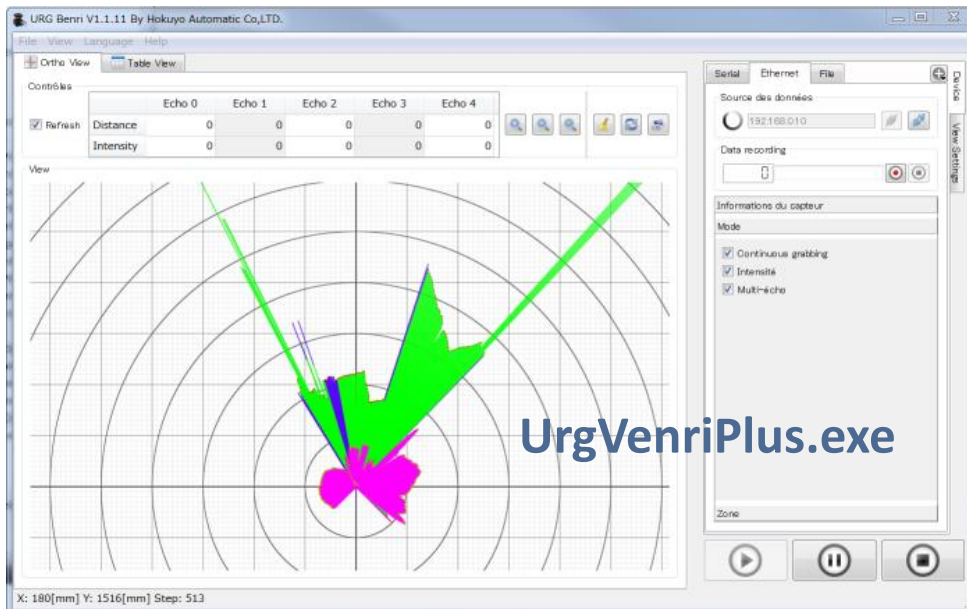
- Operation verification of Hokuyo LRF
 - How to use breadboards
 - Connect the LRF to Arduino
 - Control the robot using the LRF
-

Hokuyo LRF

URG-04LX-UG01 (Hokuyo automatic)

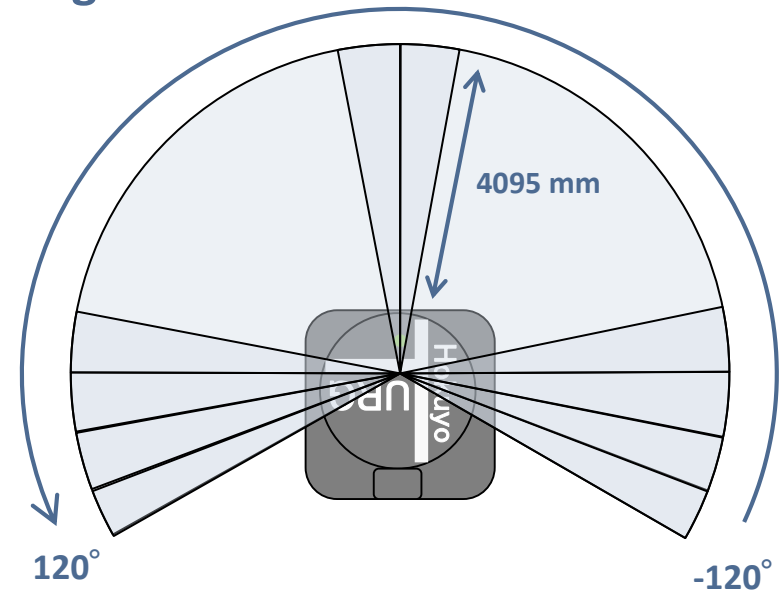


Operation verification :

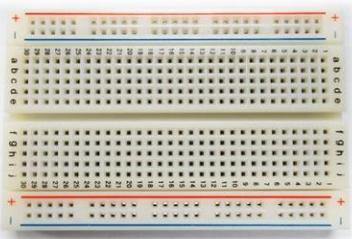


Measureable range:

Scanning direction

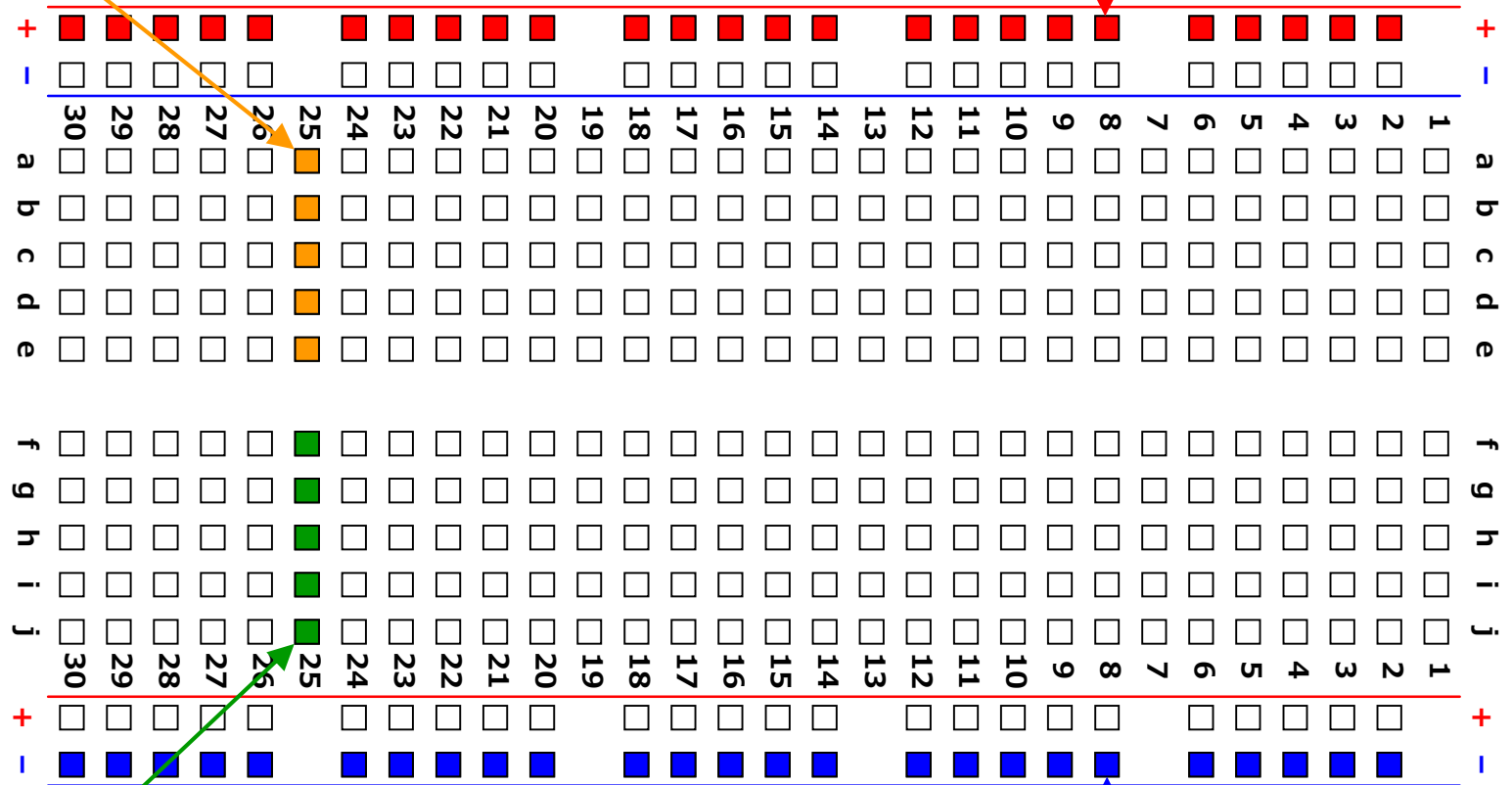


Breadboard



Conducting (same current flow)

Conducting



Not conducting between lines of (a)-(e) and (f)-(j).

Conducting

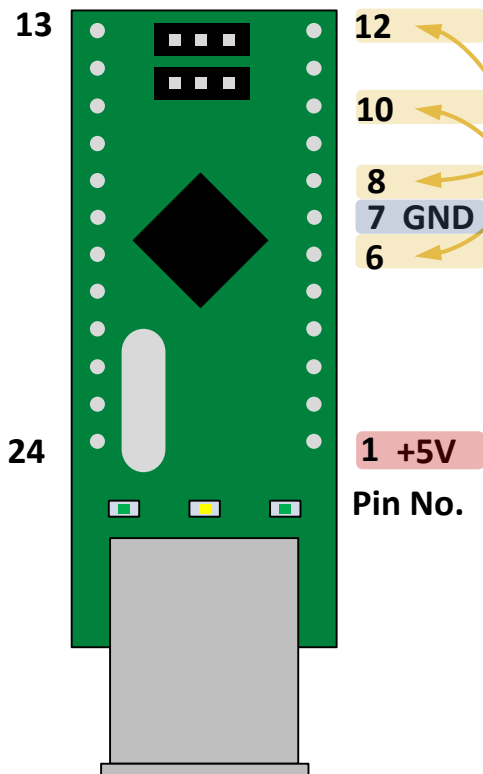
conduction

Connect LFR to v-duino (1) 1/3

Convert USB to serial comm interface (UART) :

USB host driver (VDIP1)

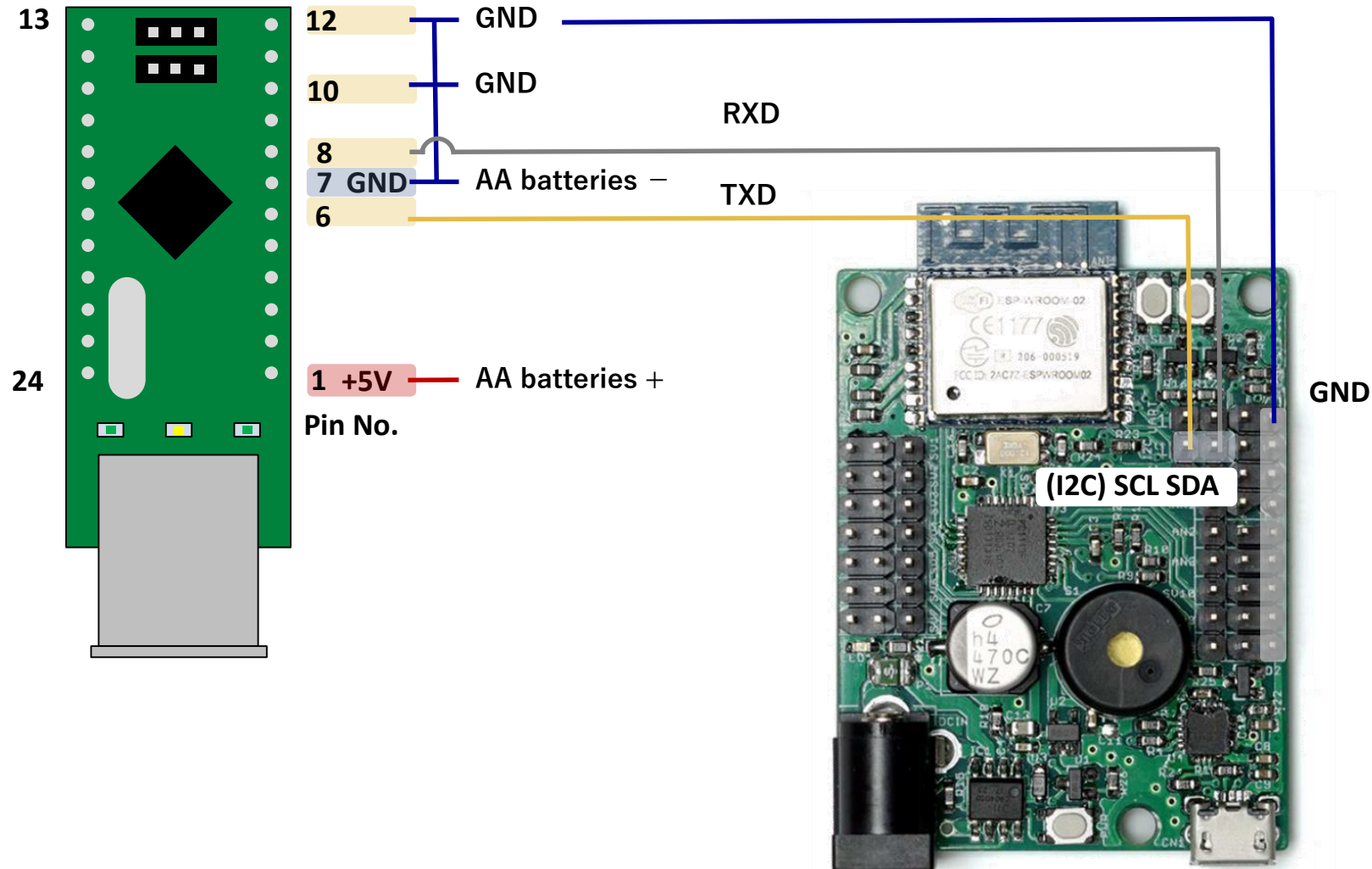
I/O pin configuration for UART interface



No.	Name	Type	Description	
6	TXD	Output	Transmit asynchronous data output	
8	RXD	Input	Receive asynchronous data output	
9	RTS#	Output	Request to send control signal	Not in use
10	CTS#	Input	Clear to send control signal	
11	DTR#	Output	Data terminal ready control signal	Not in use
12	DSR#	Input	Data set ready control signal	
13	DCD#	Input	Data carrier detect control input	Not in use
14	RI#	Input	Ring indicator control input	
15	TXDEN#	Input	Enable transmit data for RS485 designs	

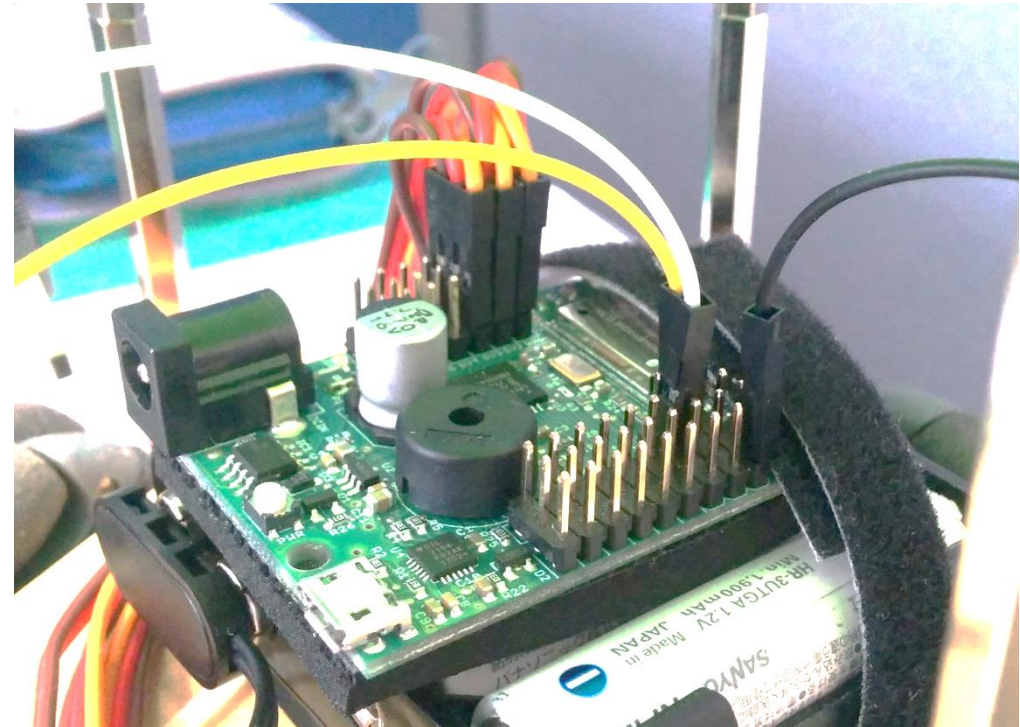
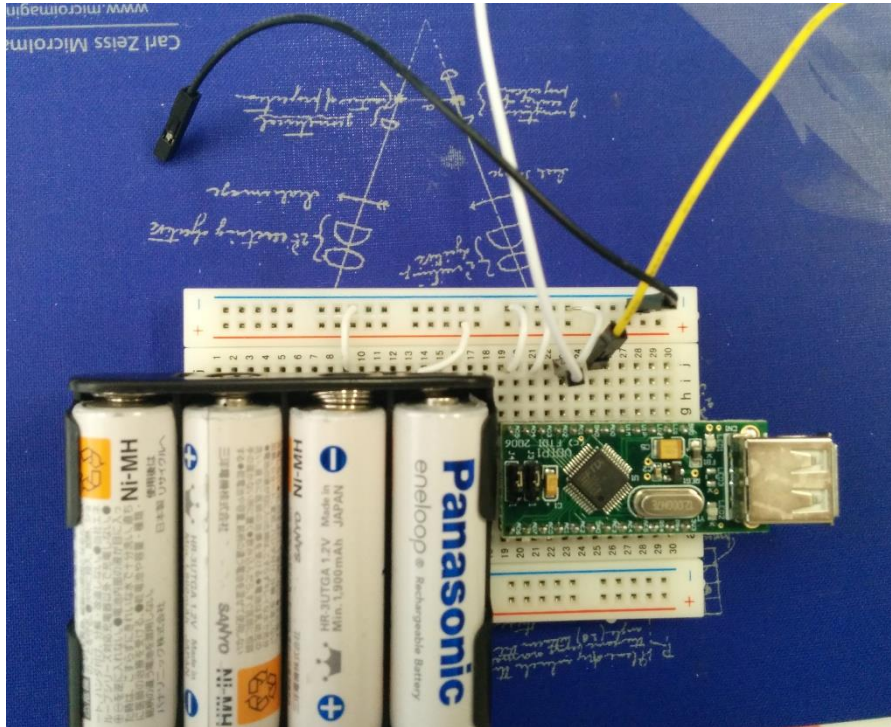
Connect LFR to v-duino (1) 2/3

Software serial comm :

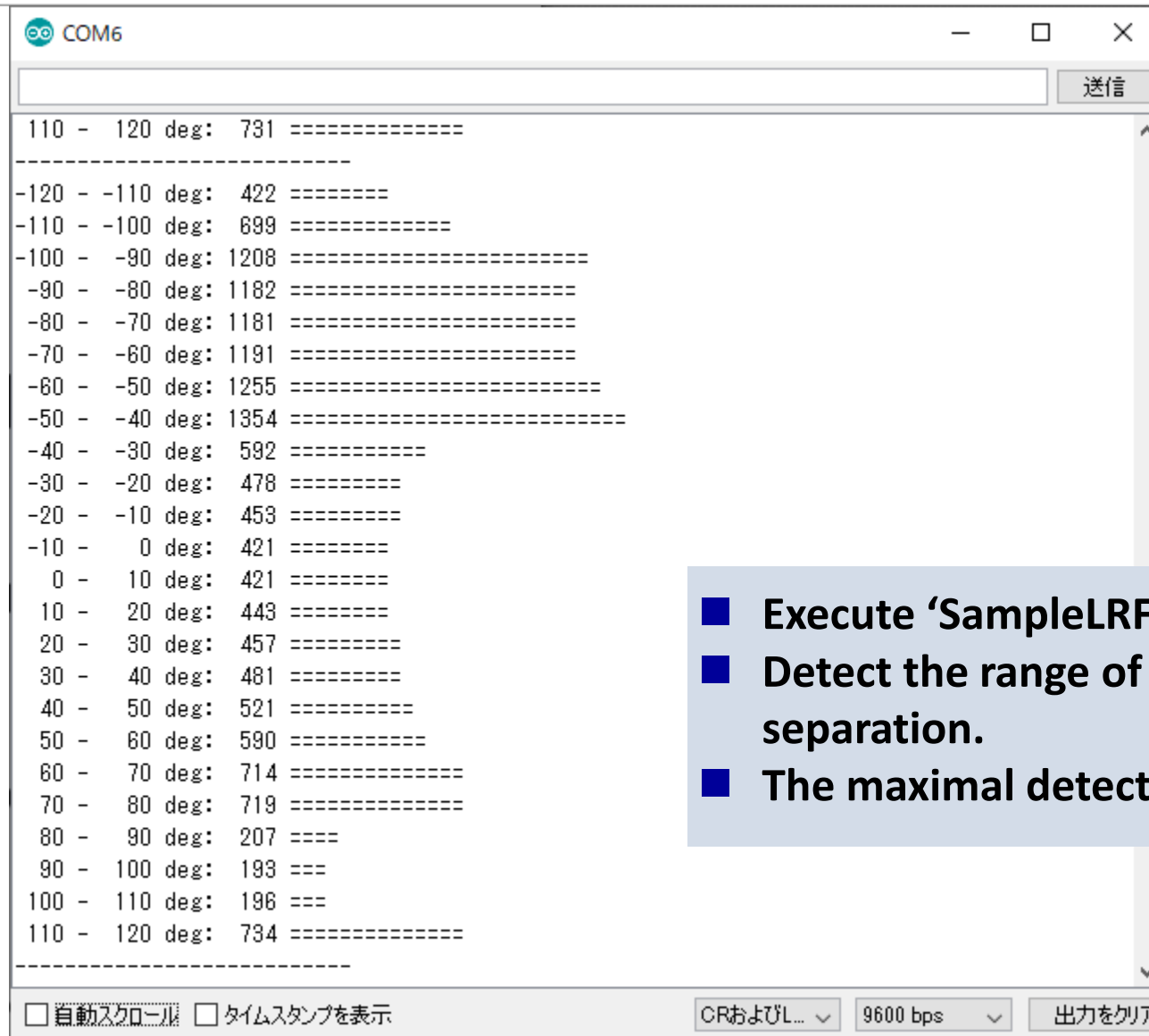


Connect LFR to v-duino (1) 3/3

Software serial comm :



Receive data from the LRF



The screenshot shows a terminal window titled 'COM6' with a '送信' (Send) button. The terminal displays a list of data points received from an LRF, organized by angle ranges. The data is as follows:

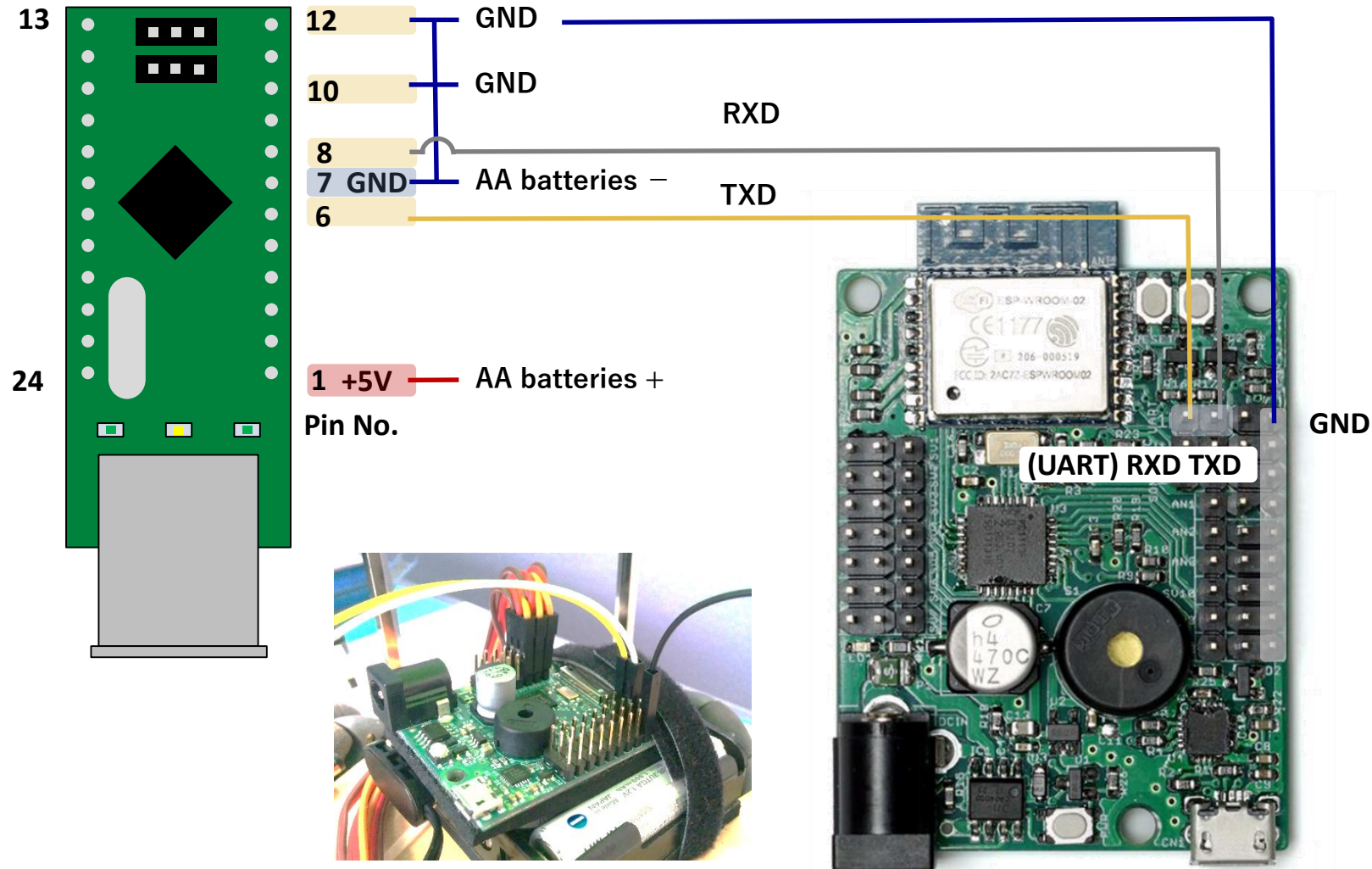
Angle Range (deg)	Value
110 - 120	731
-120 - -110	422
-110 - -100	699
-100 - -90	1208
-90 - -80	1182
-80 - -70	1181
-70 - -60	1191
-60 - -50	1255
-50 - -40	1354
-40 - -30	592
-30 - -20	478
-20 - -10	453
-10 - 0	421
0 - 10	421
10 - 20	443
20 - 30	457
30 - 40	481
40 - 50	521
50 - 60	590
60 - 70	714
70 - 80	719
80 - 90	207
90 - 100	193
100 - 110	196
110 - 120	734

At the bottom of the window, there are checkboxes for '自動スクロール' (Auto Scroll) and 'タイムスタンプを表示' (Show Timestamp), a dropdown menu for 'CRおよびL...', a baud rate selector set to '9600 bps', and a '出力をクリア' (Clear Output) button.

- Execute 'SampleLRF.ino'.
- Detect the range of -120° ~ 120° with 10° separation.
- The maximal detectable length is about 4 m.

Connect LFR to v-duino (2)

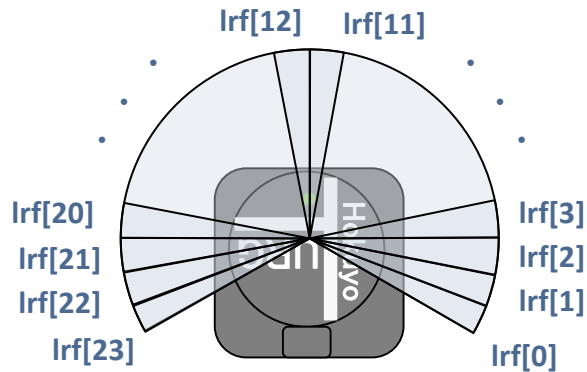
Hardware serial comm :



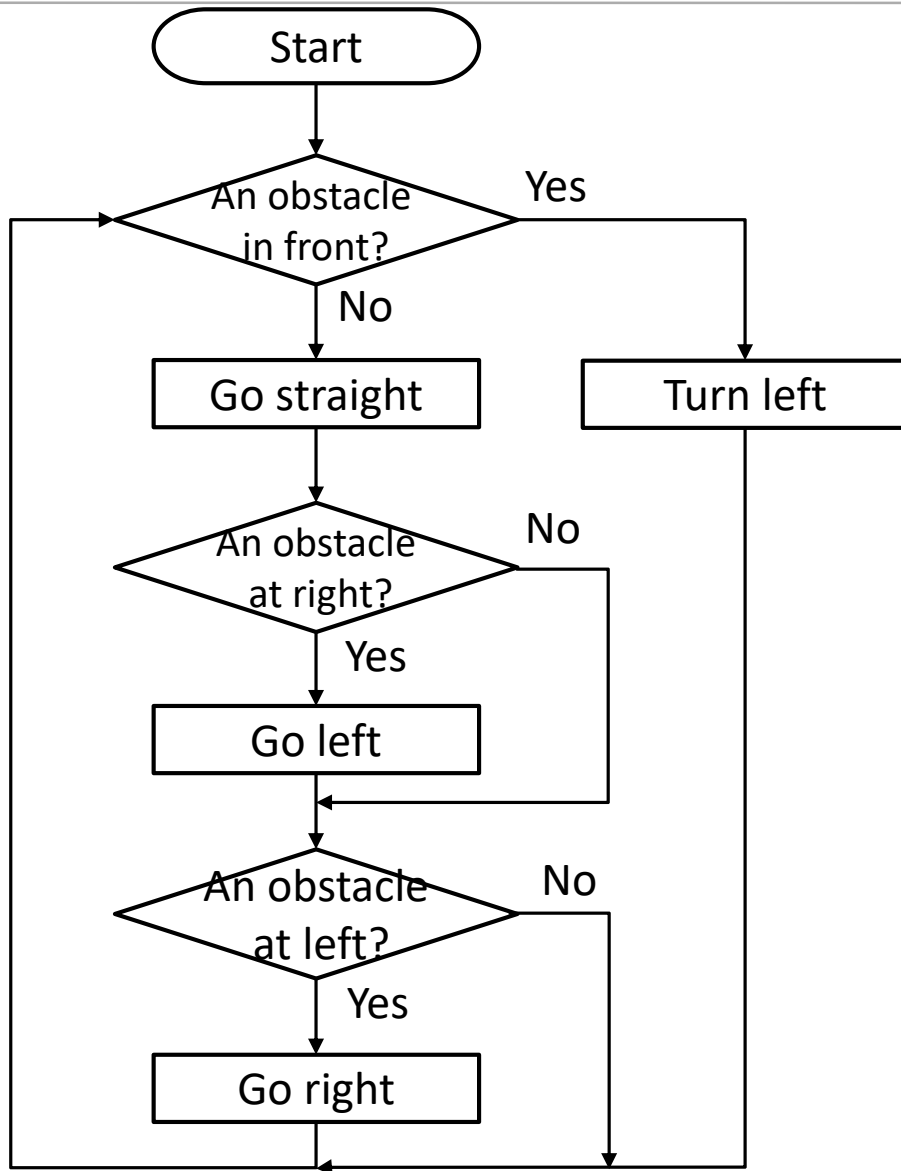
Feedback control using LRF

- Make a sketch referring to “SampleObstacleAvoidance.ino” to circle around an obstacle using the LRF.

Data assignment of the LRF :



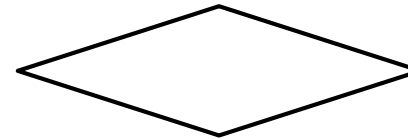
Flowchart



Start/end



Process



Decision (yes/no)

■ **Make a flowchart of your sketch circling an obstacle.**

Final Project

■ TBA

