



Course Material

Yuki Ueyama

Dept. Mechanical Engineering, National Defense Academy of Japan

Course plan

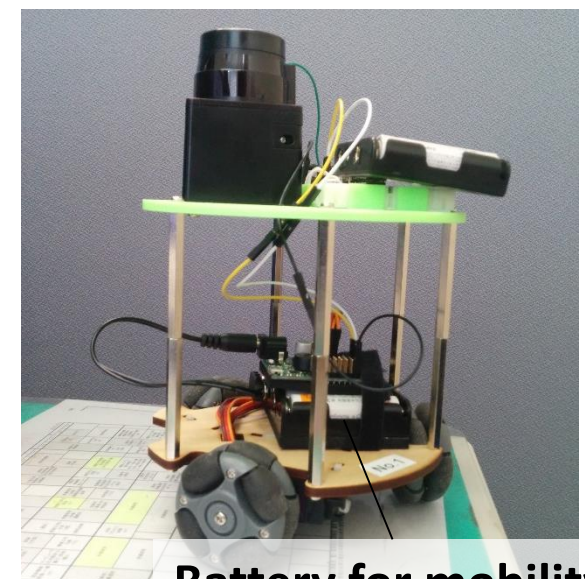
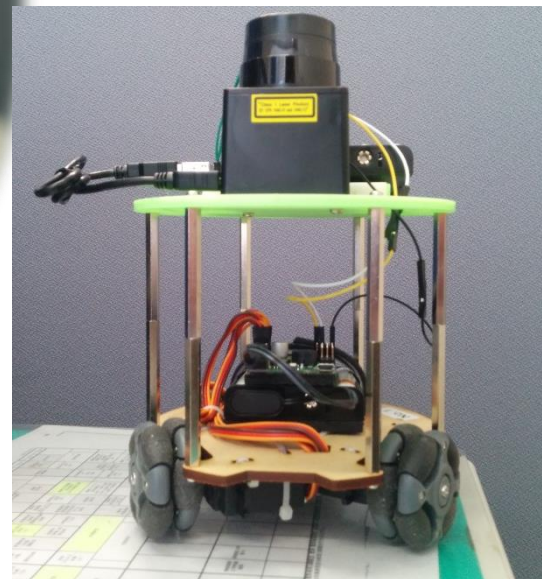
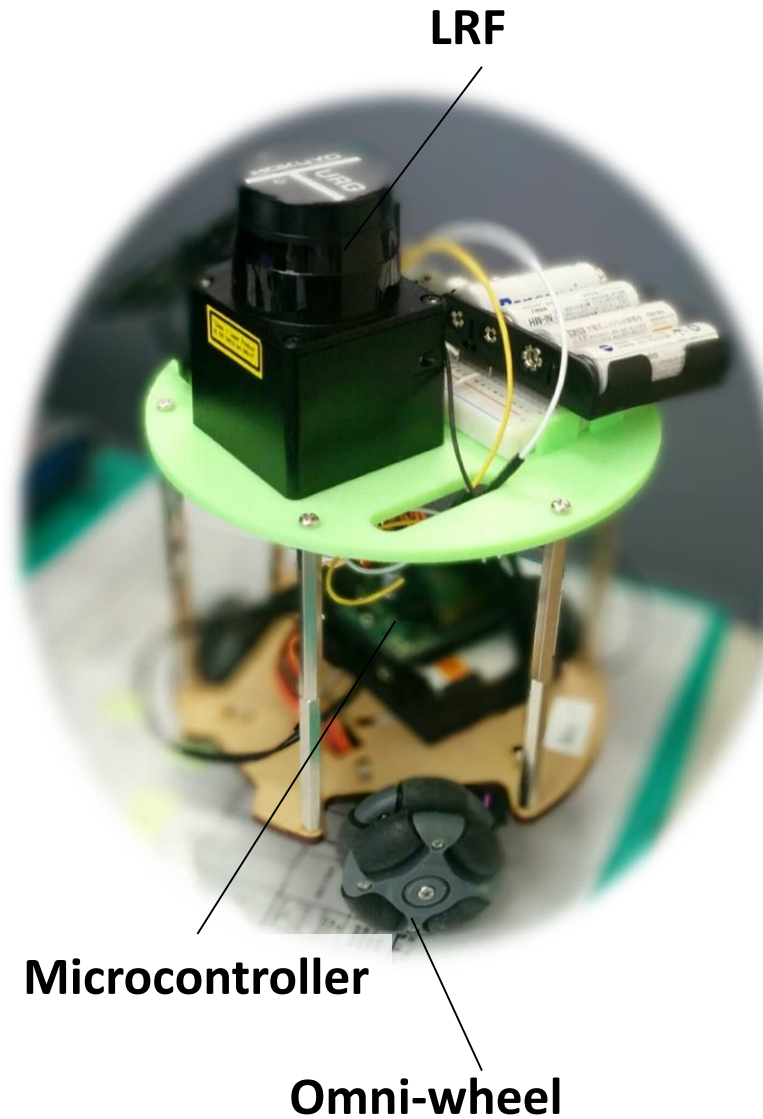
Week	Learning contents
1	Introduction
2~3	Programming basis
4	Feedback control using LRF
5~8	Project work
9	Preparing presentation
10	Presentation

Autonomous mobile vehicle



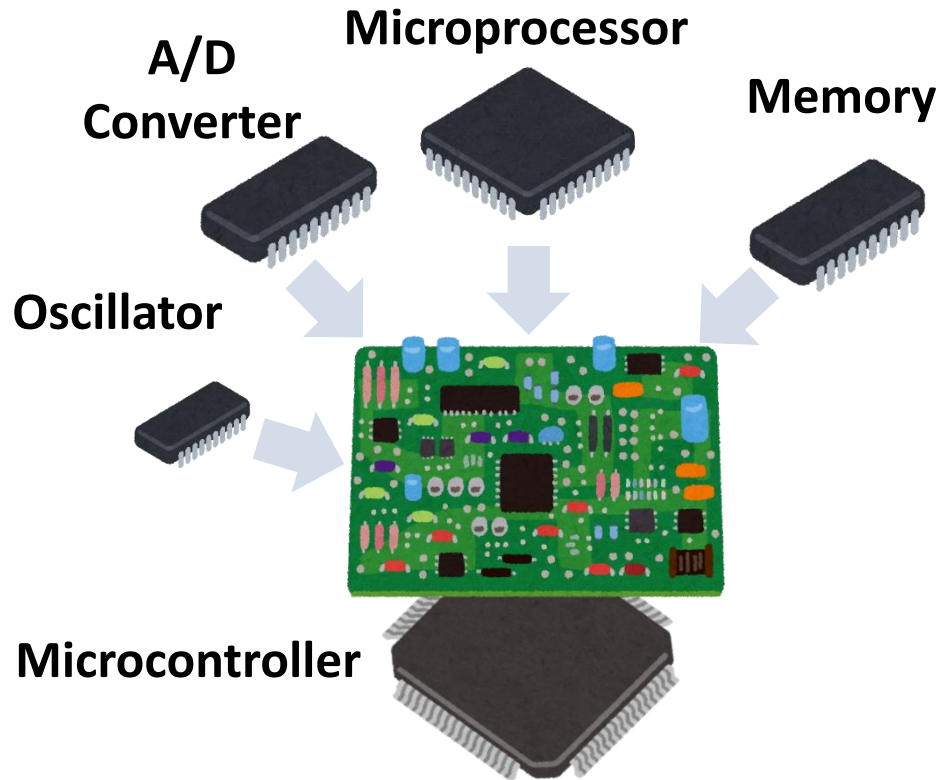
LiDAR (Light detection and ranging)
LRF (Laser range-finder)

Robot used in this course



Battery for mobility

Microcontroller



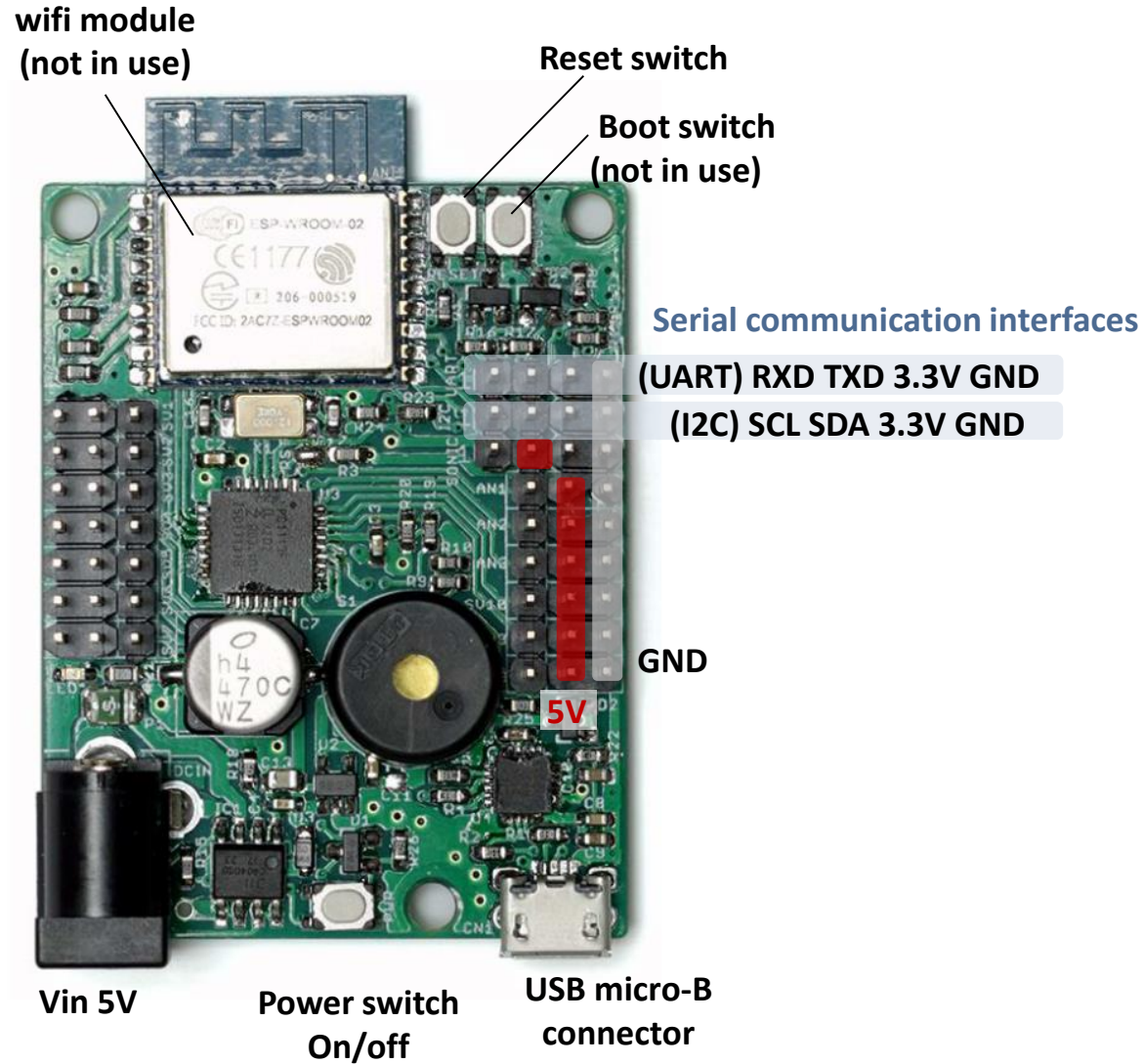
Arduino boards:



- 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

- Variables
 - IF, ELSE statements
 - FOR, WHILE statements
 - Function
 - Array
 - 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!");
}
```

SerialTest0, 26 MHz, 40MHz, DIO, 2M (1M SPIFFS), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200

Library (header file)

- Loads a set of functions for control motors.

setup() function

- Called when a sketch starts.

loop() function

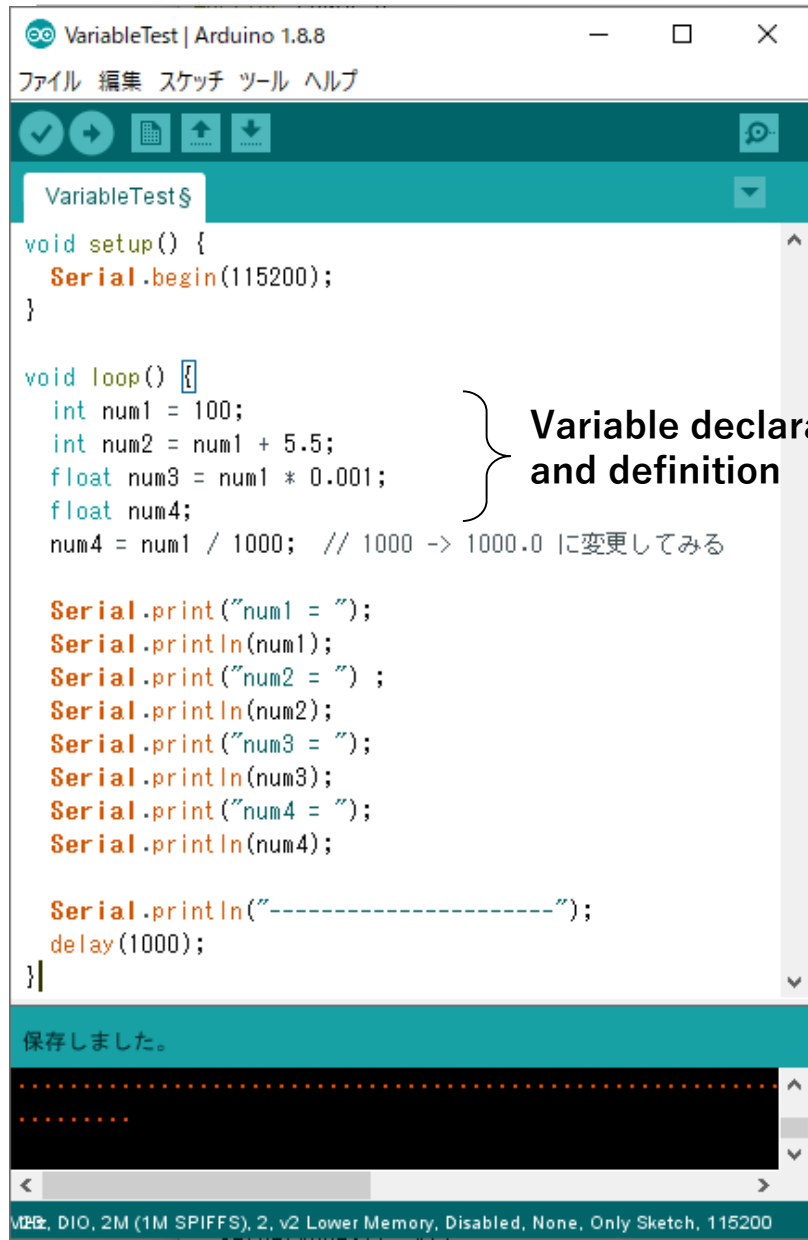
- Loops consecutively, after executing a setup() function.

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

Serial monitor
(Outputs of the sketch)

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


Variable



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

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);
}
```

保存しました。

VR2, DIO, 2M (1M SPIFFS), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200

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 KB, 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 least 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	aとbが等しくない
A < B	aがbより小さい
A > B	aがbより大きい
A <= B	aの値がbの値以下
A >= B	aの値がbの値以上

Switch :

```
switch (変数) {  
    case 値1 :  
        変数 = 値1のときに実行される命令  
        break;  
    case 値2 :  
        変数 = 値2のときに実行される命令  
        break;  
    case 値3 :  
        変数 = 値3のときに実行される命令  
        break;  
        .  
        .  
        .  
    default :  
        すべてのケースに一致しないとき実行される命令  
}
```

繰り返し処理



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

```
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 to board completed.)

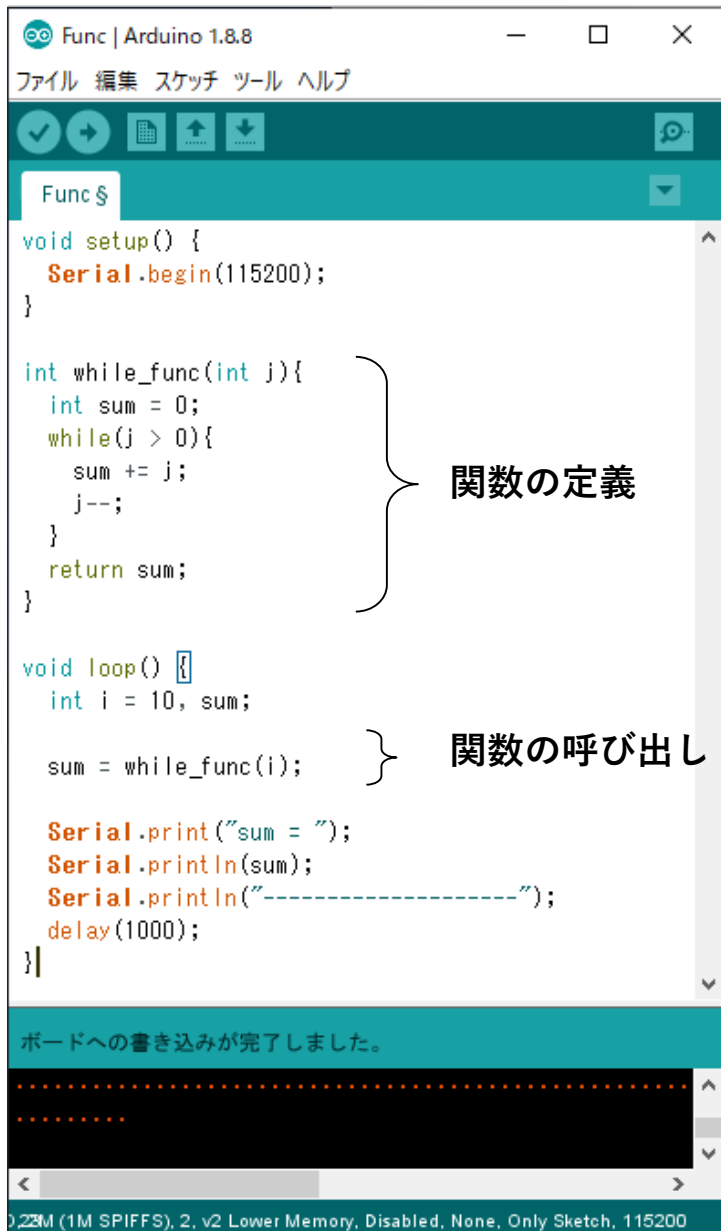
for文 :

for (初期値; 条件式; 増減式) {
 条件式が満たされている時に実行される命令
 .
 .
 .
}

while文 :

while (条件式) {
 条件式が満たされている時に実行される命令
 .
 .
 .
}

関数



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 "関数の定義" (Function Definition).
- A bracket on the right side of the `while_func(i)` call in the `loop` function is labeled "関数の呼び出し" (Function Call).

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 serial monitor window is visible.

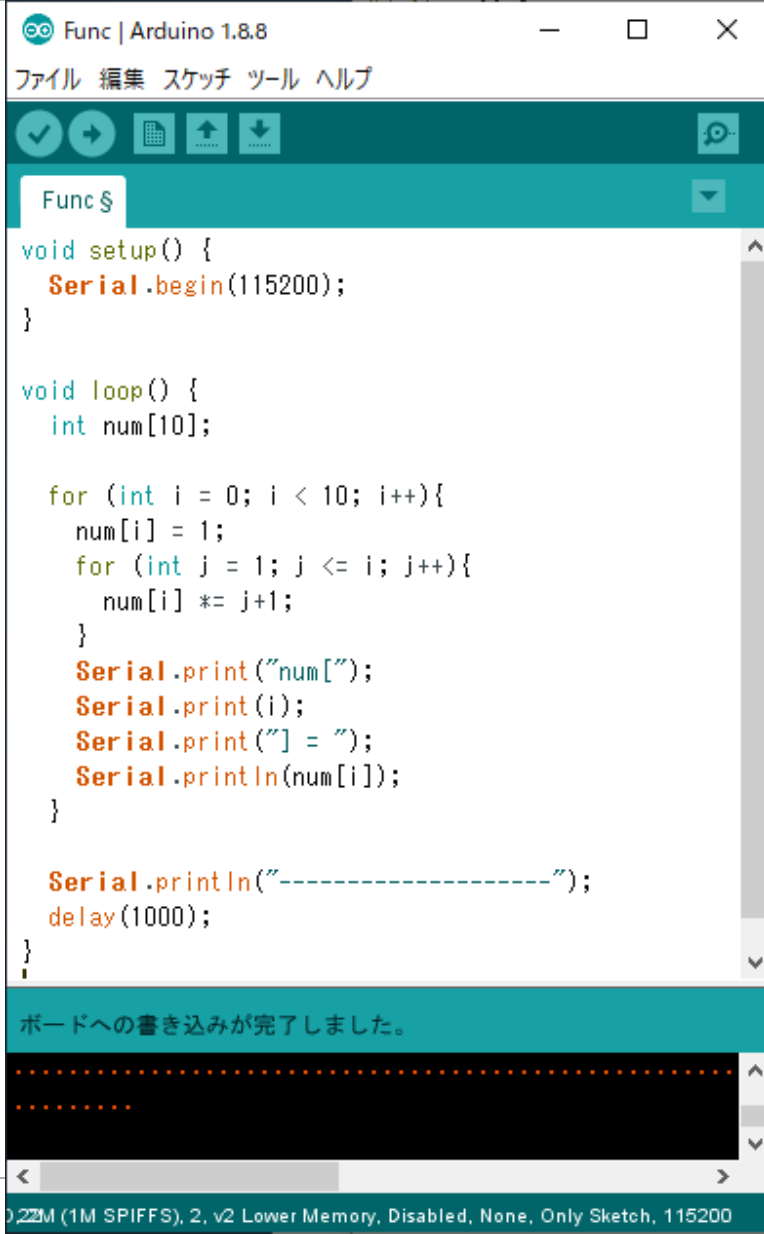
戻り値のある関数 :

```
型名 関数名(型名 引数1, 型名 引数2, . . . ) {  
    処理  
    .  
    .  
    .  
    return 戻り値;  
}
```

戻り値のない関数 :

```
void 関数名(型名 引数1, 型名 引数2, . . . ) {  
    処理  
    .  
    .  
    .  
}
```

配列



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

Func $
void setup() {
  Serial.begin(115200);
}

void loop() {
  int num[10];

  for (int i = 0; i < 10; i++){
    num[i] = 1;
    for (int j = 1; j <= i; j++){
      num[i] *= j+1;
    }
    Serial.print("num[");
    Serial.print(i);
    Serial.print("] = ");
    Serial.println(num[i]);
  }

  Serial.println("-----");
  delay(1000);
}

ボードへの書き込みが完了しました。
-----
-----
0.22M (1M SPIFFS), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200
```

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

型 配列名[要素数] 初期値
(他と被っていないけれ (宣言したときにはな
ば、なんでもよい) くてよい)

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

要素数が3でも、num[0]から始まるため、最後の要素はnum[2]となる。

2次元配列 :

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

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

シリアル通信によるモータの制御

```
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();           //Initilize 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();

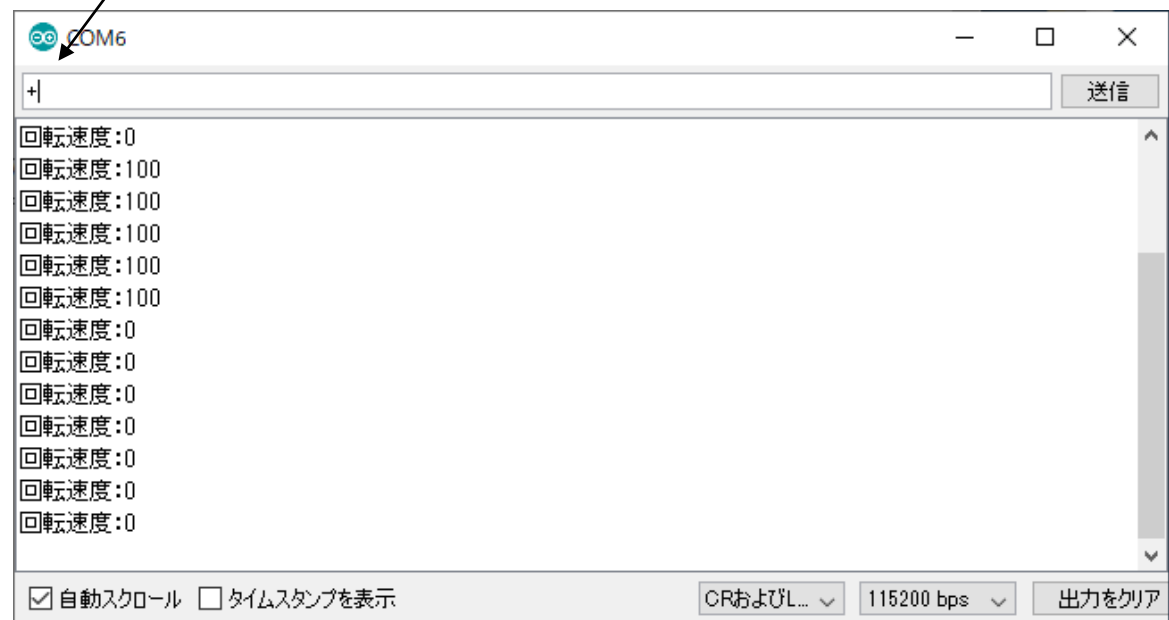
    if (sgn == '+') { // 入力された文字が+のとき
      spd = spd + 100;
    } else if (sgn == '-') { // 入力された文字が-のとき
      spd = spd - 100;
    }
  }

  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
```

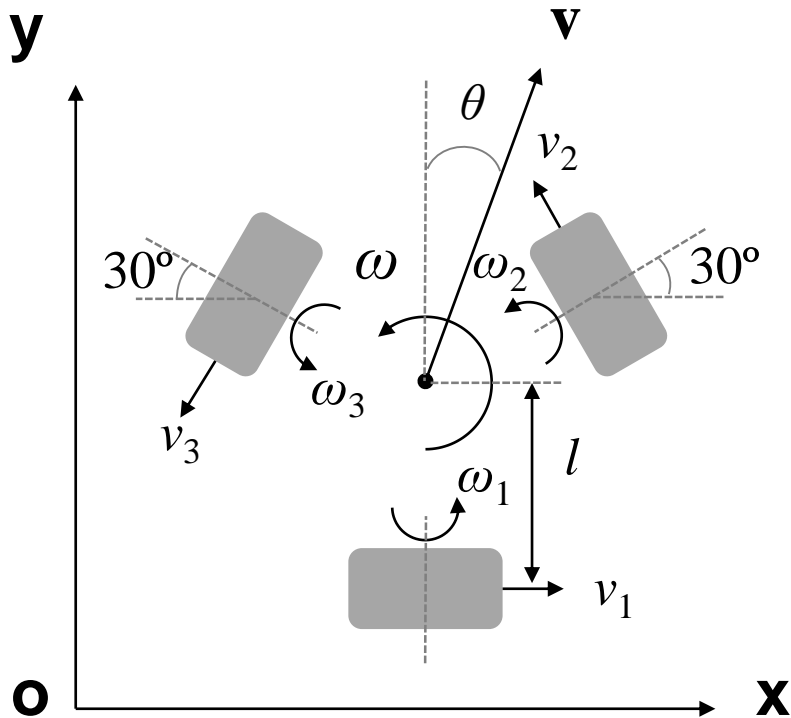
ここに“+”または“-”を入力して、Enterキーを押す。



Kinematics of the mobile robot 1/2

機体座標系での速度：

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



各車輪の移動速度：

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

r ：車輪の半径

Kinematics of the mobile robot 2/2

絶対座標系と機体座標系での速度の関係 :

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

ϕ : ロボットの向き

機体座標系での速度

絶対座標系での速度

各車輪の移動速度 :

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

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

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

ロボットのフィードフォワード制御

- サンプルプログラム「`omnirover3wd_auto_motion.ino`」を参考にして、障害物の周囲を1周するプログラムを作成しなさい。

測域センサを使用した制御

- 測域センサの特性
 - ブレッドボードの使い方
 - 測域センサとマイコンとの接続
 - 測域センサを使用したロボットの制御
-

測域センサ



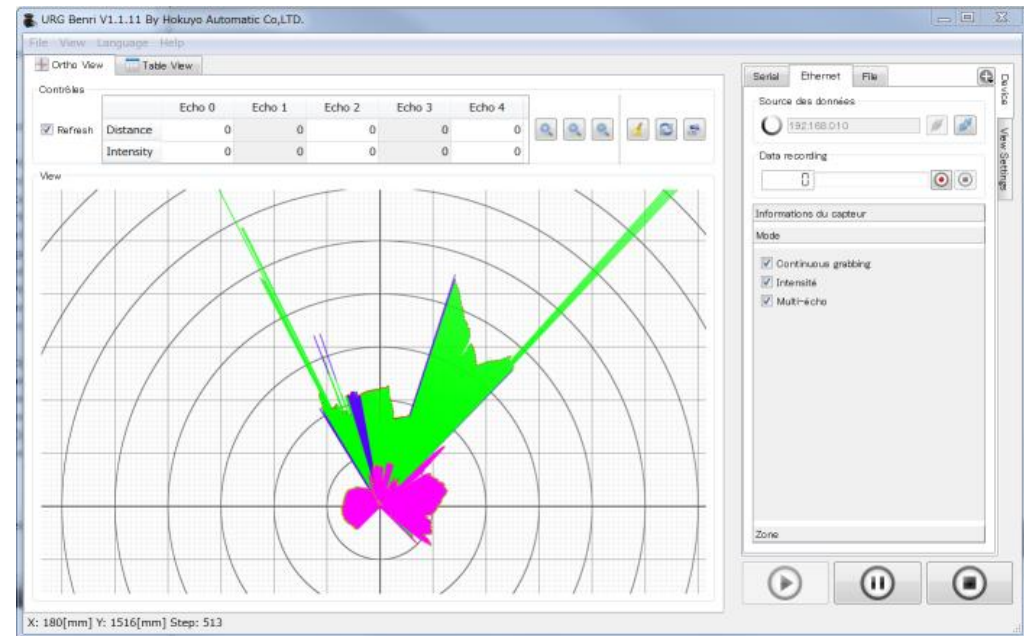
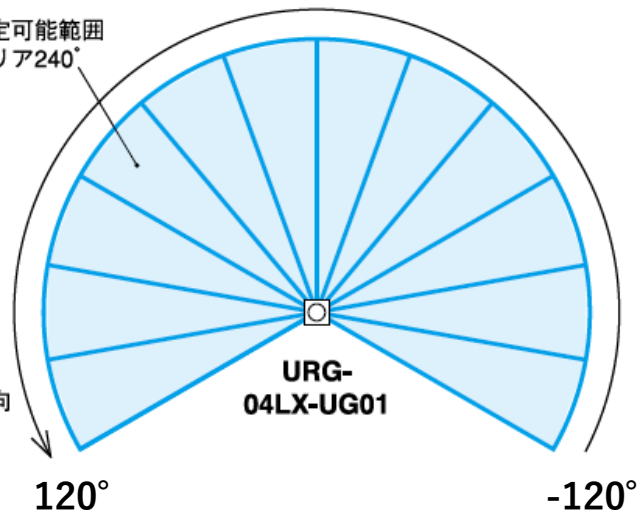
北陽電機 URG-04LX-UG01

動作確認

- デスクトップからUrgBenriPlusを起動

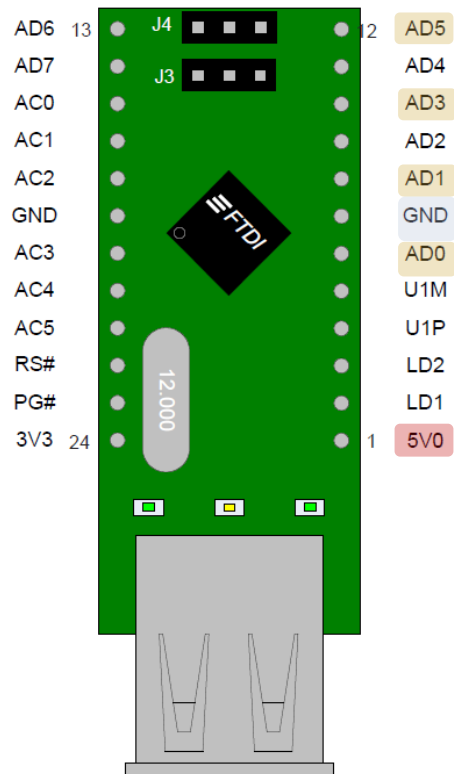
スキャン設定可能範囲
及び検出エリア240°

スキャン方向
(Top-view)



測域センサとv-duinoの接続① 1/3

USBからシリアル通信（UART）への変換：

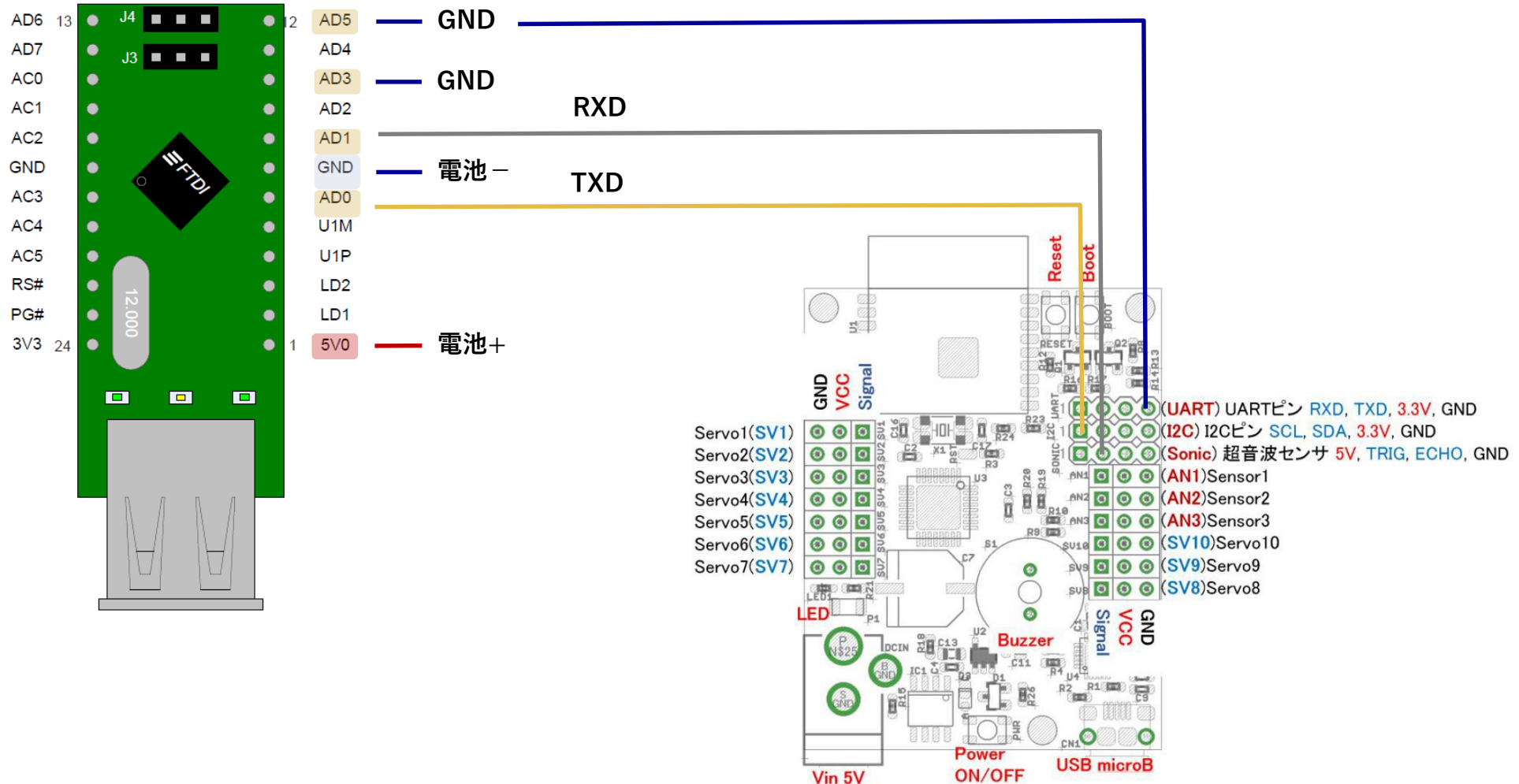


Pin No.	Name	Type	Description
6	TXD	Output	Transmit asynchronous data output
8	RXD	Input	Receive asynchronous data input
9	RTS#	Output	Request To Send Control Output / Handshake signal
10	CTS#	Input	Clear To Send Control Input / Handshake signal
11	DTR#	Output	Data Terminal Ready Control Output / Handshake signal
12	DSR#	Input	Data Set Ready Control Input / Handshake signal
13	DCD#	Input	Data Carrier Detect Control Input
14	RI#	Input	Ring Indicator Control Input. When the Remote Wake Up option is enabled in the EEPROM, taking RI# low can be used to resume the PC USB Host controller from suspend
15	TXDEN#	Input	Enable Transmit Data for RS485 designs

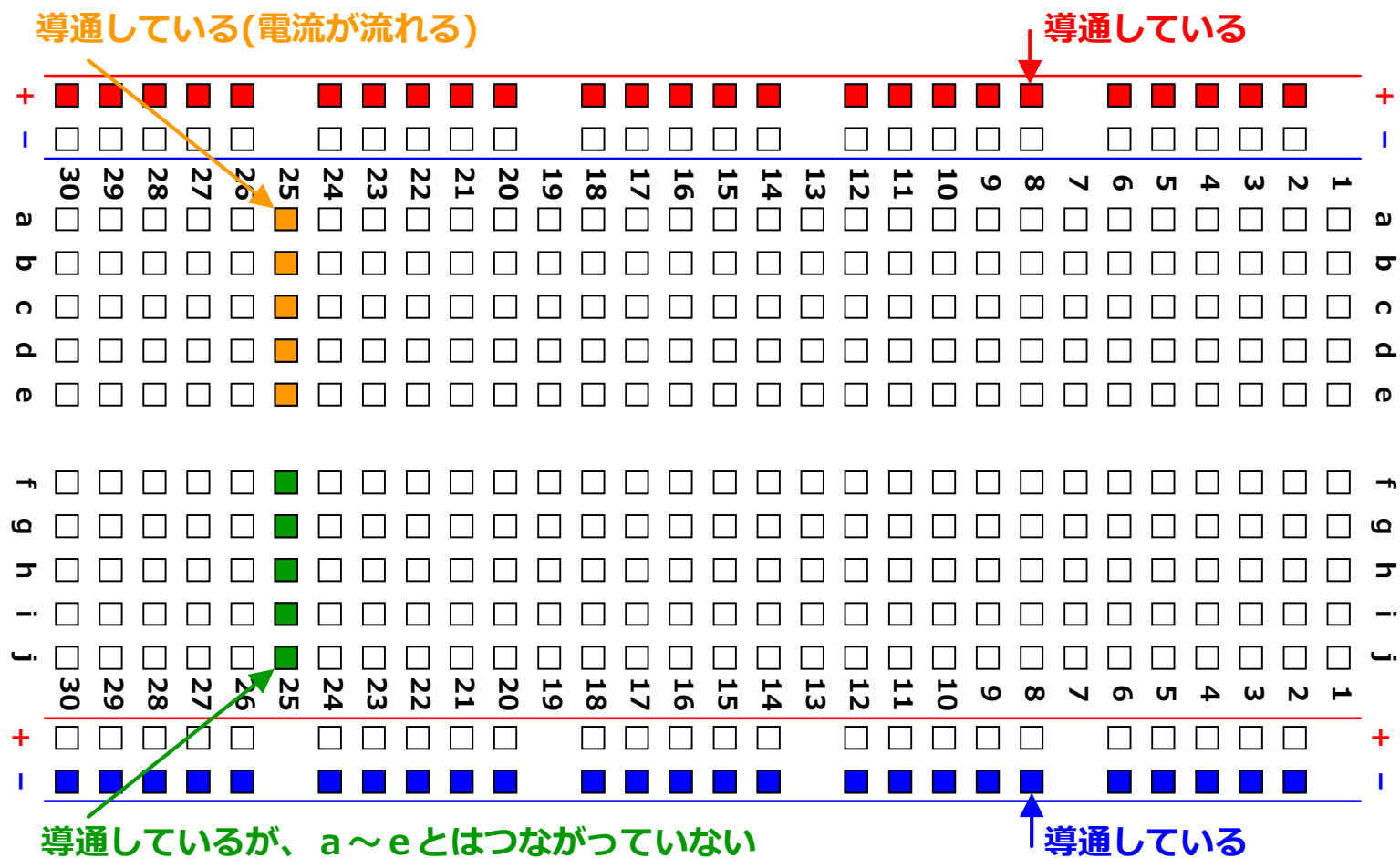
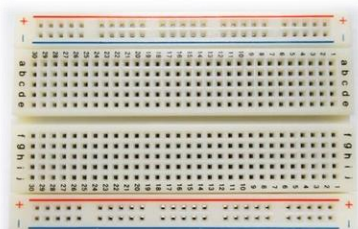
Table 3.4 - Default I/O Pin Configuration – UART Interface

測域センサとv-duinoの接続① 2/3

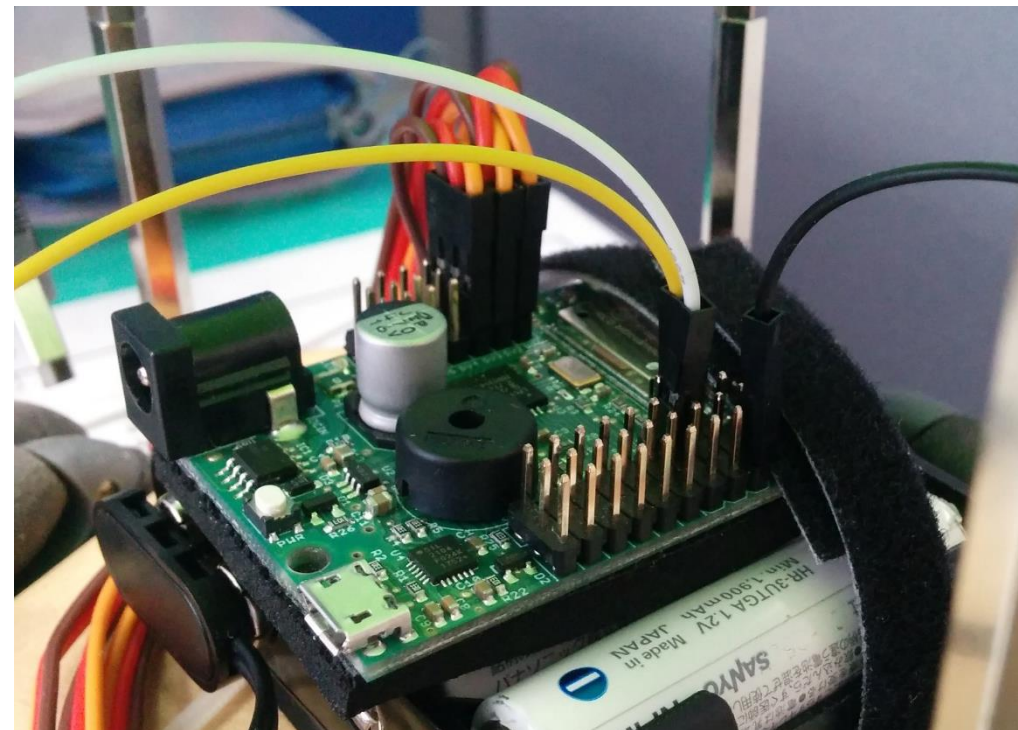
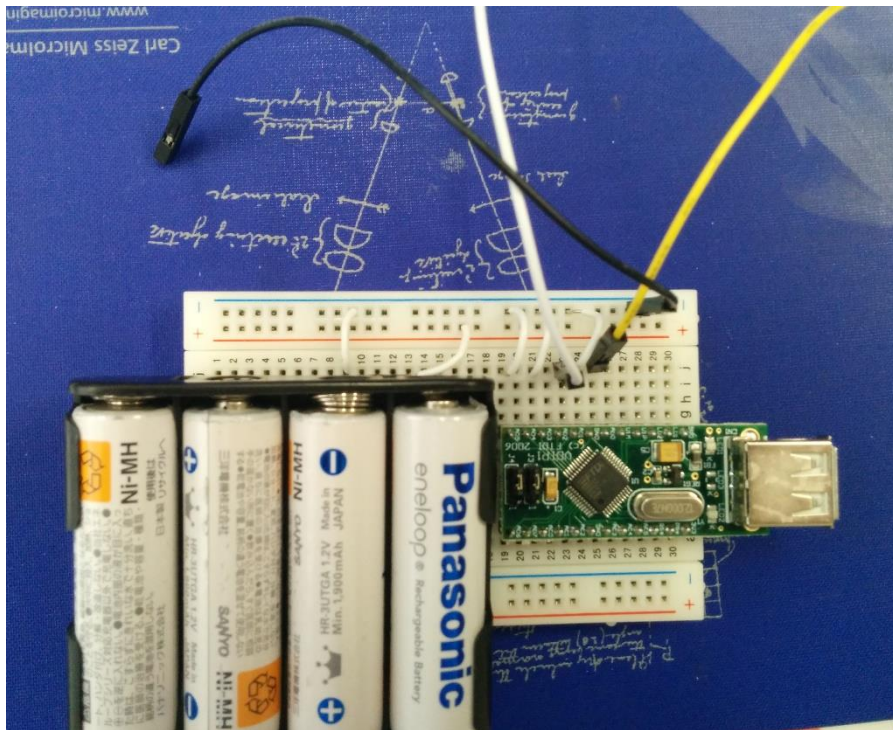
ソフトウェアシリアル :



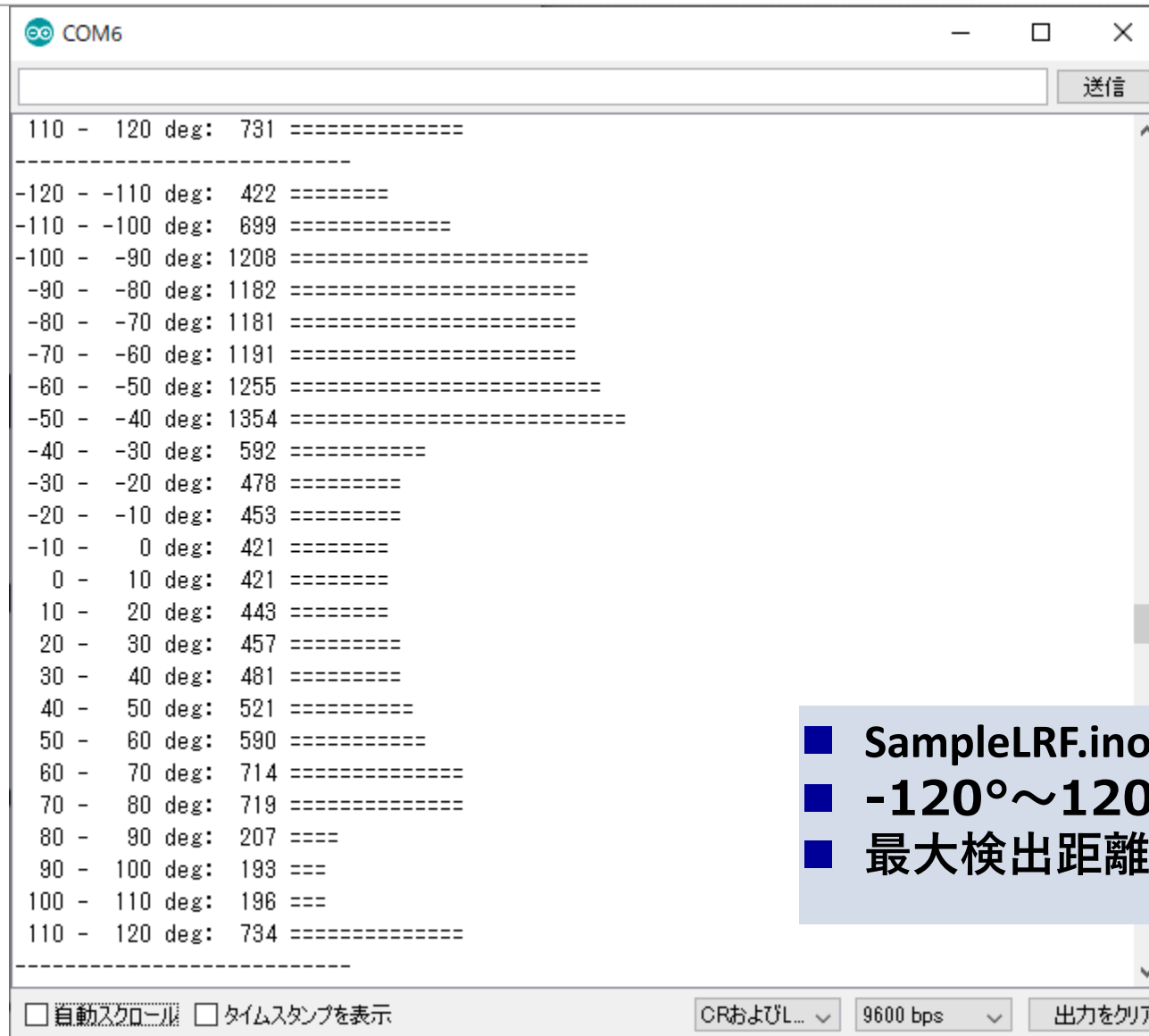
ブレッドボード



測域センサとv-duinoの接続① 3/3



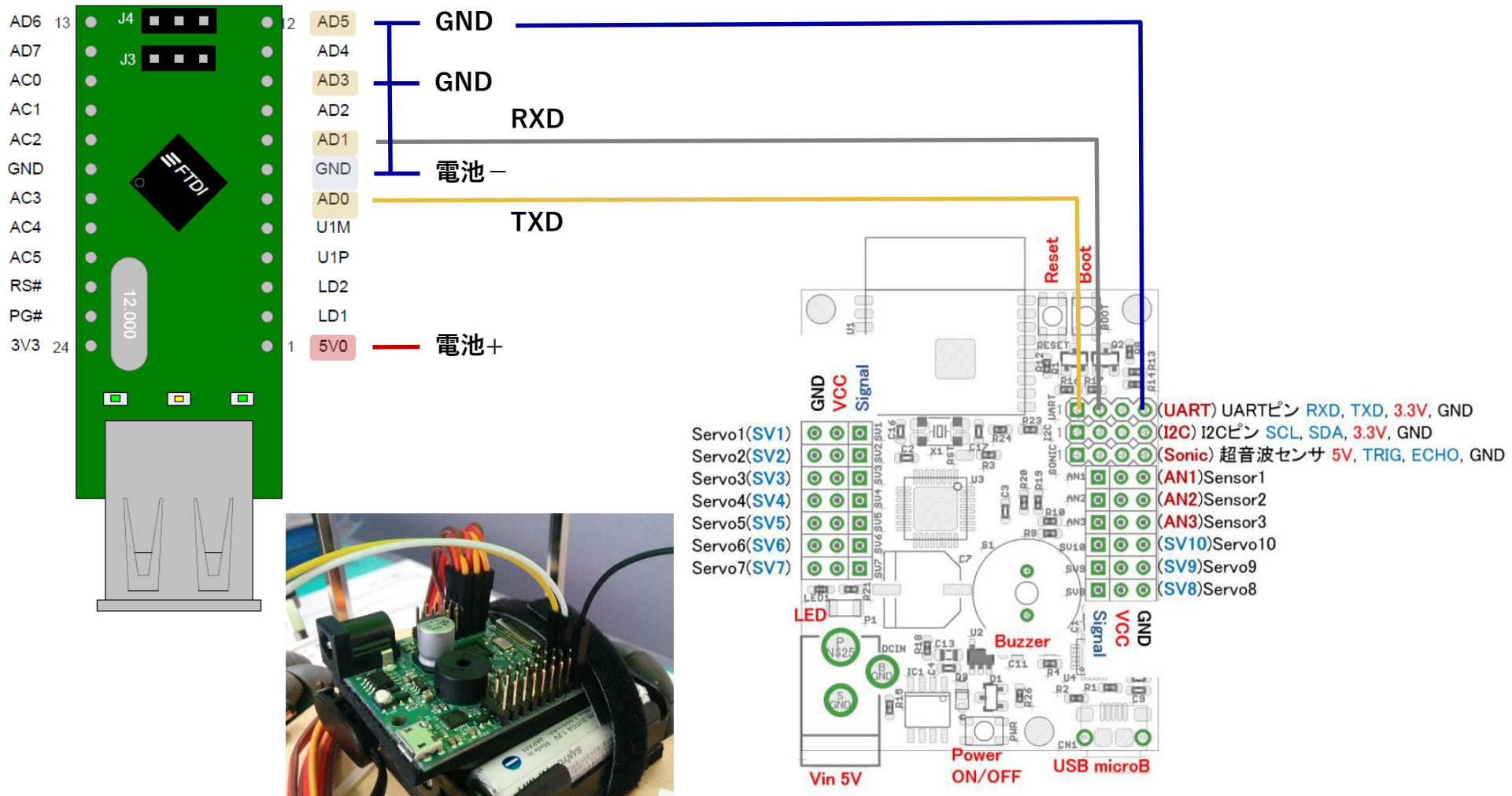
測域センサの値の確認



```
COM6
送信
110 - 120 deg: 731 =====
-----
-120 - -110 deg: 422 =====
-110 - -100 deg: 699 =====
-100 - -90 deg: 1208 =====
-90 - -80 deg: 1182 =====
-80 - -70 deg: 1181 =====
-70 - -60 deg: 1191 =====
-60 - -50 deg: 1255 =====
-50 - -40 deg: 1354 =====
-40 - -30 deg: 592 =====
-30 - -20 deg: 478 =====
-20 - -10 deg: 453 =====
-10 - 0 deg: 421 =====
0 - 10 deg: 421 =====
10 - 20 deg: 443 =====
20 - 30 deg: 457 =====
30 - 40 deg: 481 =====
40 - 50 deg: 521 =====
50 - 60 deg: 590 =====
60 - 70 deg: 714 =====
70 - 80 deg: 719 =====
80 - 90 deg: 207 ===
90 - 100 deg: 193 ==
100 - 110 deg: 196 ==
110 - 120 deg: 734 =====
-----
☐ 自動スクロール ☐ タイムスタンプを表示
COMポート: COM6 波特率: 9600 bps 出力をクリア
```

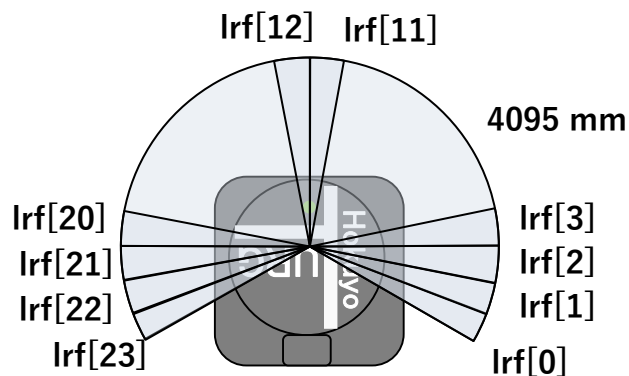
- SampleLRF.inoを実行
- -120°～120°を10°間隔で検出
- 最大検出距離は約4 m

測域センサとv-duinoの接続②

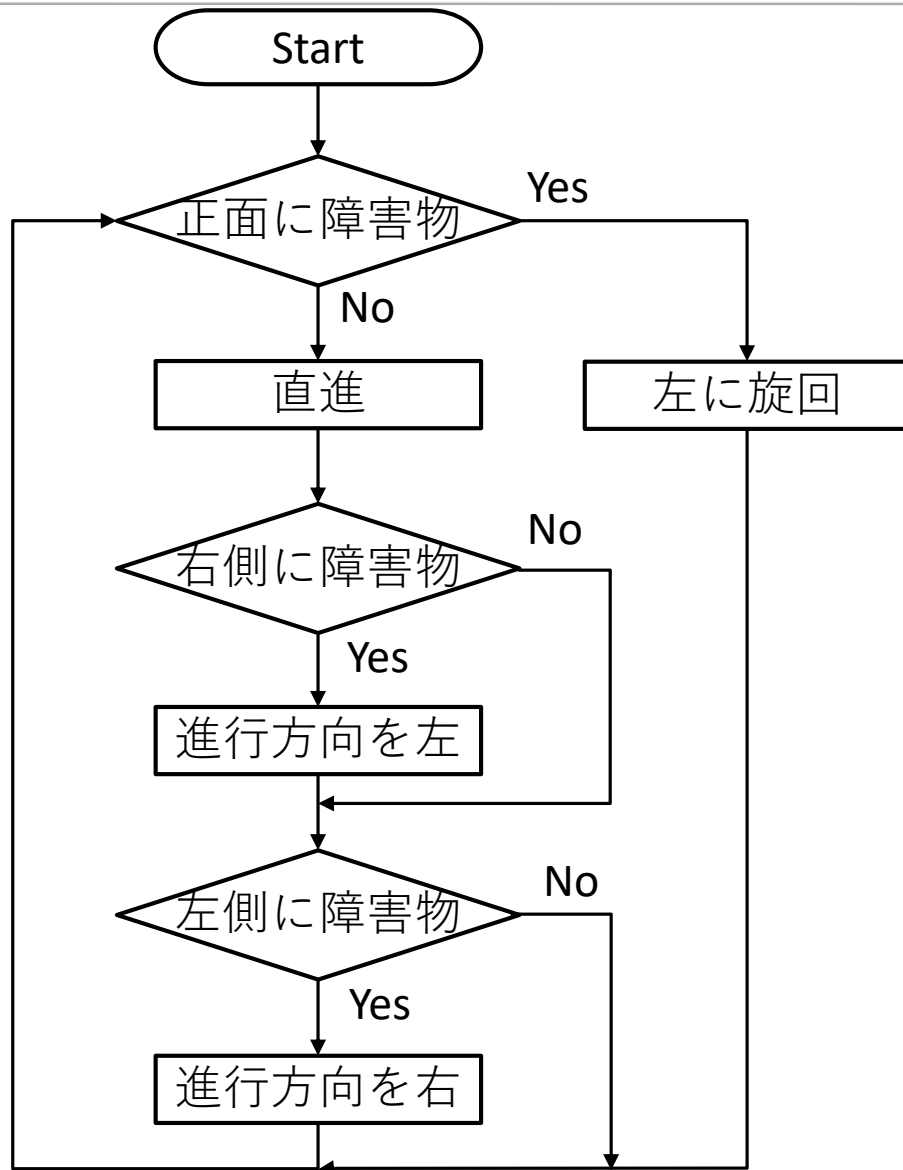


測域センサを使った制御

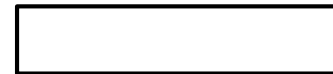
- サンプルプログラム「SampleObstacleAvoidance.ino」を参考に、測域センサのデータを使用して障害物の周囲を1周するプログラムを作成しなさい。



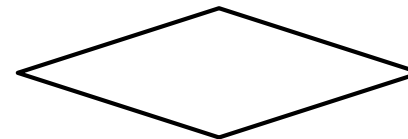
フローチャート



プログラムの開始



処理



条件 (if文)

- 障害物の周囲を一周するプログラムについて、フローチャートを作成しなさい。

Final Project

■ TBA