B 日程作業報告

1. 作業の引継ぎ

A 日程の作業報告から次のことから B 日程の作業を始めた。

a) 3D プリンタを使ったバンパーの作成
b) 標準機の組み立て
c) Arduino と Raspberry Pi の接続

2. 標準機の組み立て

標準機を組み立てるにあたってバンパーを 3D プリンタで作成した。Figure1 に完成したバンパーを示す。

Figure 1　完成したバンパー

このバンパーを標準機の上段シャーシに取り付ける際、固定する穴の位置が 90° ずれていた。そのため、上段シャーシに穴を空けなおした。

Figure2 に完成した MIRS 標準機を示す。

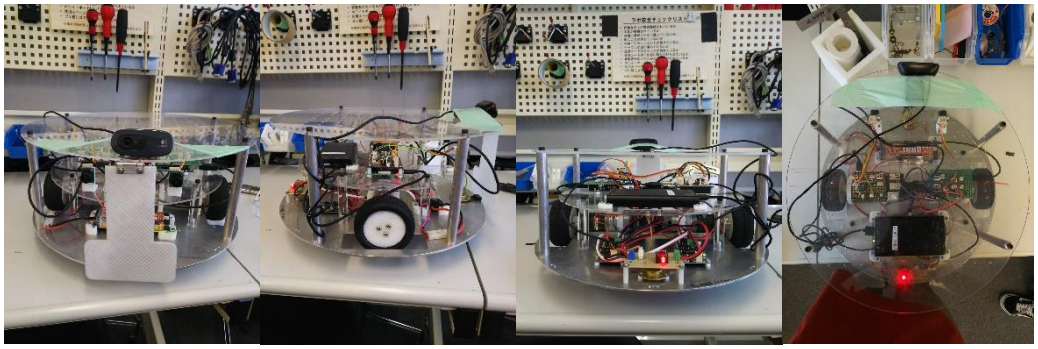

Figure 2　完成した標準機

3. 機能試験

機能試験では Arduino と Raspberry Pi を接続して機能の確認を行った。その結果、超音波センサとタッチセンサ、数字認識に動作不良が認められた。これらについて以下に詳しく記す。

3.1 超音波センサについて

超音波センサの動作不良について、test_uss を実行した結果、左の超音波センサが接続状態にならなかった。この問題に対して、以下のようなことを試した。

a) 配線の見直し
b) ケーブルの取り換え
c) 左右の超音波センサのケーブルのコネクタを交換
d) 接触不良の見直し
e) 超音波センサの交換

この結果、a)～d)からケーブルや接触の問題ではないとわかり、e)の超音波センサの交換を行ったところ動作が改善した。

このことから、動作不良の原因は超音波センサの故障だと判明した。

3.2 タッチセンサについて

タッチセンサの動作不良について、test_io を実行した結果タッチセンサが反応を示さなかった。この問題に対して、以下のようなことを試した。

a) タッチセンサから Raspberri Pi までの導通テスト
b) タッチセンサの交換
c) 配線の見直し

この結果、a,b からタッチセンサやケーブルの問題ではないとわかった。そのため、配線を見直してみたところ、標準プログラムでは IO18 ピンを使用することになっているが IO7 ピンを使用していた。

このことから、プログラムを変更すればよいとわかりタッチセンサのピンの指定を行っている io.c というプログラムを Table 1 のように変更した。

Table 1　io.c の変更

変更前(一部抜粋)	変更後(一部抜粋)

#include <stdio.h>
#include <wiringPi.h>
#include "io.h"
 	
// ピン配置
//static const int pin_sw_f　= 25;
//static const int pin_sw_l　= 21;
//static const int pin_sw_r　= 27;
static const int pin_sw_f　= 18;
static const int pin_sw_l　= 25;
static const int pin_sw_r　= 26;
 	
……
	#include <stdio.h>
	#include <wiringPi.h>
	#include "io.h"
	
	// ピン配置
	//static const int pin_sw_f　= 25;
	//static const int pin_sw_l　= 21;
	//static const int pin_sw_r　= 27;
	static const int pin_sw_f　= 7;
	static const int pin_sw_l　= 25;
	static const int pin_sw_r　= 26;
	
	……

このようにプログラムを変更した結果、正常に動作するようになった。

3.3　数字認識について

数字認識の動作不良について、test_number を実行した結果「7」のみ認識しなかった。この問題に対して、プログラムで数字「7」を判定するパターンの設定が間違っているのではないかと考えた。そのため、画像の読み取り方を検証した結果、Figure3 のように読み取っていることがわかった。

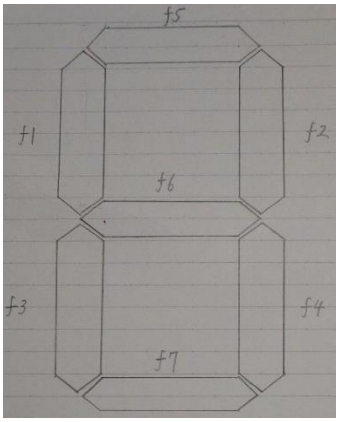

Figure 3　画像の読み取り方

これを踏まえて、数字の認識を行っているプログラム number_search.cpp の変更を Table 2 に示す。

Table 2　数字判定の変更

変更前(一部抜粋)

……
// 数字判別
if(f1==1 && f2==1 && f3==1 && f4==1 && f5==1 && f6==1 && f7==0) number = 0;
if(f1==0 && f2==1 && f3==1 && f4==0 && f5==1 && f6==1 && f7==1) number = 2;
if(f1==0 && f2==1 && f3==0 && f4==1 && f5==1 && f6==1 && f7==1) number = 3;
if(f1==1 && f2==1 && f3==0 && f4==1 && f5==0 && f6==0 && f7==1) number = 4;
if(f1==1 && f2==0 && f3==0 && f4==1 && f5==1 && f6==1 && f7==1) number = 5;
if(f1==1 && f2==0 && f3==1 && f4==1 && f5==1 && f6==1 && f7==1) number = 6;
if(f1==1 && f2==1 && f3==0 && f4==1 && f5==1 && f6==0 && f7==0) number = 7;
if(f1==1 && f2==1 && f3==0 && f4==1 && f5==1 && f6==1 && f7==1) number = 9;
……

変更後(一部抜粋)

……
// 数字判別
if(f1==1 && f2==1 && f3==1 && f4==1 && f5==1 && f6==1 && f7==0) number = 0;
if(f1==0 && f2==1 && f3==1 && f4==0 && f5==1 && f6==1 && f7==1) number = 2;
if(f1==0 && f2==1 && f3==0 && f4==1 && f5==1 && f6==1 && f7==1) number = 3;
if(f1==1 && f2==1 && f3==0 && f4==1 && f5==0 && f6==0 && f7==1) number = 4;
if(f1==1 && f2==0 && f3==0 && f4==1 && f5==1 && f6==1 && f7==1) number = 5;
if(f1==1 && f2==0 && f3==1 && f4==1 && f5==1 && f6==1 && f7==1) number = 6;
if(f1==0 && f2==1 && f3==0 && f4==1 && f5==1 && f6==0 && f7==0) number = 7;
if(f1==1 && f2==1 && f3==0 && f4==1 && f5==1 && f6==1 && f7==1) number = 9;
……

このようにプログラムを変更したところ「7」を認識するようになった。

4. 統合試験

4.1 Makefile について

統合試験及び追加課題のプログラムを Makefile でコンパイルできるように変更した。Table 3 に変更内容を示す

Table 3　Makefile への書き足し

変更前(一部抜粋)

```
……7 行目から……
# 生成するプログラム
PRGS :=
PRGS += test_io test_uss test_request test_direction
PRGS += test_capture test_number test_dir_num
PRGS += test_server test_position
PRGS += pilot
……
……一番最後の行……
test_server: $(OBJS_SVR) $(DIR_OBJ)/test_server.o
        $(CC) $(LDFLAGS) $^ -o $@

test_position: $(OBJS_POS) $(DIR_OBJ)/test_position.o
        $(CC) $(LDFLAGS) $^ -o $@
```

変更後(一部抜粋)

```
……7 行目から……
# 生成するプログラム
PRGS :=
PRGS += test_io test_uss test_request test_direction straight number avoidance avoidance_2
PRGS += test_capture test_number test_dir_num
PRGS += test_server test_position
PRGS += pilot
……
……一番最後の行に書き足し……
test_server: $(OBJS_SVR) $(DIR_OBJ)/test_server.o
        $(CC) $(LDFLAGS) $^ -o $@

test_position: $(OBJS_POS) $(DIR_OBJ)/test_position.o
$(CC) $(LDFLAGS) $^ -o $@

straight: $(OBJS_RUN) $(DIR_OBJ)/straight.o
        $(CC) $(LDFLAGS) $^ -o $@

number: $(OBJS_RUN) $(OBJS_CAP) $(OBJS_NUM) $(DIR_OBJ)/number.o
        $(CC) $(LDFLAGS) $^ -o $@

avoidance: $(OBJS_RUN) $(DIR_OBJ)/avoidance.o
        $(CC) $(LDFLAGS) $^ -o $@

avoidance_2: $(OBJS_RUN) $(DIR_OBJ)/avoidance_2.o
        $(CC) $(LDFLAGS) $^ -o $@
```

4.2 直進性能試験

直進性能試験の詳しいプログラムは straight.c を参照する。Table 4 に試験の結果を示す。

Table 4　直進性能試験

走行速度	左右のずれ	距離のずれ	調整内容	実施者	確認教員
30cm/s	右に 3.8cm	-1.5cm	ギア比を 1/27 のものにする タイヤの回転比 L_R_RATIO を 1.84 にする タイヤ半径 R_TIRE を 4.47 にする ゲイン Ks_d を 50 にする	位田直弥 菊池愛結 榊原里樹 村山慶将	青木先生 牛丸先生
60cm/s	無し	2.5cm	ギア比を 1/27 のものにする タイヤの回転比 L_R_RATIO を 1.83 にする タイヤ半径 R_TIRE を 4.45 にする ゲイン Ks_d を 50 にする	位田直弥 菊池愛結 榊原里樹 村山慶将	青木先生 牛丸先生
特記事項	straight.c という名前でプログラムを作成した ギア比，L_R_RATIO，R_TIRE は Arduino の define.h にある モータのギア比は 1/27 であった L_R_RATIO は左右のずれの調整 R_TIRE は走行距離の調整 Ks_d は Arduino の run_ctrl にある Ks_d は PID 制御の積分ゲイン				

4.3 数字認識性能試験

数字認識性能試験の詳しいプログラムは number.c を参照する。Table 5 に試験の結果を示す。

Table 5　数字認識性能試験

試験回数	認識精度	調整内容	実施者	確認教員
2	100%	Raspberry Pi のプログラムで数字とカメラの距離指定を調整 カメラの角度を調整 Arduino の run_ctrl でゲイン Kr を-0.5 にする 「7」の認識の調整	榊原里樹 位田直弥	香川先生
特記事項	number.c という名前でプログラムを作成した 最初に距離が長いところから始めるか短いところから始めるか選択できる ゲイン Kr は回転角度の調整 「7」の認識は Raspberry Pi の number_search.cpp というプログラムの数字判別という部分の 7 に対応する f1 を f1 == 1 となっているのを f1 == 0 に変える			

4.4 障害物回避試験

障害物回避試験の詳しいプログラムは avoidance.c を参照する。Table 6 に試験の結果を示す。

Table 6　障害物回避試験

超音波による回避行動	タッチセンサによる回避行動	調整内容	実施者	確認教員
合格	合格	回避行動の際の「左回転、右回転をランダムに行う」「回転角度を 90~180 度でランダムにする」に苦戦したが、time 関数を使用することで実現することができた	村山慶将 菊池愛結	牛丸先生 大沼先生
特記事項	avoidance.c という名前でプログラムを作成した ランダム角度回転により壁に対して小さい角度で直進する時、超音波センサが読み取れず壁にぶつかることがあった（鋭角でも回避するプログラムが B 日程の加点課題となった）			

5. 追加課題

追加課題の内容は障害物回避試験の改善である。障害物回避試験では壁に対して小さい角度で直進したとき超音波センサがうまく反応せず壁に衝突してしまっていた。これを改善し、完全に壁に衝突せずコース内を自由に動き回れるようにするというのが追加

課題の内容である。

この課題に対して Figure 4 に示すように、センサの値から角度を割り出し、角度から MIRS 本体と壁との距離を算出し、角度が $\pi/4$[rad]以下で壁に侵入する場合と角度から割り出した距離が 50[cm]以下の場合回避行動に移るようにプログラムした。

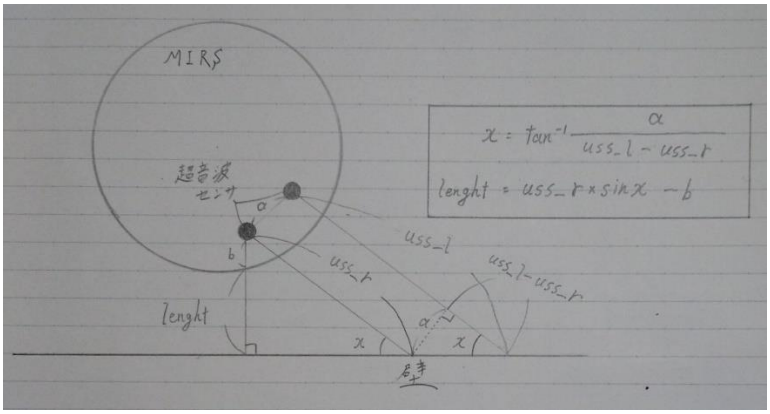

Figure 4　回避行動の考え方

しかし、これだけでは完全に回避しきれなかった。その原因は、超音波センサが壁に対して斜めに距離を読み取ると正しく読み取れていなかったということが大きな原因であった。この正しく読み取られていない読み取った値には次のようなものがあった。

a) コースの最大値である 280[cm]以上を読み取る
b) 左右の超音波センサの読み取り値の差が大きくずれる
c) 読み取れず-1 を返す

この 3 つのことに対して回避行動をプログラムに組み込んだところ壁に衝突することはなくなった。しかし、a)に対してはただの直進動作でも頻繁にみられることであり、これを組み込むと必要のない回避動作が多くなりすぎてしまった。そのため、a)の回避動作は行わないこととしたところ壁に衝突しない動作が完成した。

追加課題を実現するための動作を以下にまとめる。

1) 壁に侵入する角度を算出し $\pi/4$ より小さい角度で侵入したとき回避行動をする
2) 角度から壁との距離を算出し 50[cm]以下なら回避行動をする
3) 左右の超音波センサの読み取り値の差が 150[cm]以上だったら回避行動する
4) 超音波センサの読み取り値が-1 だったら回避行動をする

これらを実行した結果、追加課題の動作を行うようにできた。このプログラムの詳細は avoidance_2.c を参照する。

6. 特記事項

B 日程の作業を実施した上での変更点や注意点をまとめたものを Table 7 に示す。

Table 7　特記事項

項目名	特記事項	記入者
機体組み立て	標準機組み立て手順書をみて標準機を組み立てた。また、バンパーを 3D プリンタで印刷し設置する穴を開けた。	位田直弥 菊地愛結
RaspberryPi 単体での動作試験	WiFi(11A)に接続ができなかった。モバイルホットスポットにすることによって解決した。 test_request のファイルのサイズが 0 になっていた。 make clean をして make all することでプログラムが現れ解決した。	村山慶将
超音波センサ	左の超音波センサが認識されなかった。 超音波センサが故障していたので交換した。	村山慶将
タッチセンサ	タッチセンサは IO7 のピンを使用していた。そのため Raspberry Pi のプログラム io.c の pin_sw_f = 7 とすることでタッチセンサを認識する。	榊原里樹
数字認識	test_number を実行した際「7」だけ認識されなかった。 number_search.cpp というプログラムで認識を定義している。 数字の認識は 7segLED をイメージすると、左上が f1,右上が f2,左下が f3,右下が f4,上が f5,真ん中が f6,下が f7 に対応していた。 7 に対応する部分の f1 を f1 == 1 から f1 == 0 に変えると「7」を認識した	榊原里樹 村山慶将
直進制御	直進制御するとき速度を遅く設定しすぎるとうまく動作しない。 speed = 5[cm/s]としたときタイヤが前後に動き続けた。	榊原里樹
Arduino のプログラム	Arduino のプログラムを Raspberry Pi にダウンロードした /pi/Document/　に直進性能試験の 30[cm/s]ようと 60[cm/s]用で分けて入れてある	榊原里樹
統合試験	それぞれの試験に対して以下のようなプログラムを作成した 直進走行試験：straight.c 数字認識性能試験：number.c 障害物回避試験：avoidance.c	榊原里樹
追加課題	障害物回避試験では角度の関係で壁に衝突しまうため、追加課題では壁に衝突せず範囲内を動き回るようにする。 このプログラムは avoidance_2.c という名前で作成した。	榊原里樹
Makefile	統合試験と追加課題のプログラムのために Makefile を変更した。 変更内容は Table.3 を参照。	榊原里樹