トモダチロボットの制作 － ITGs －

内田　望愛　　小松原　ゆあ
髙見　来実　　藤井　香
堀　夕海

１．研究概要

M5Stack のプログラム制御に関する知識を身に付け、「話す」、「聞く」、「動く」、「表す」などの人間と同じような動作をする小型ロボット「トモダチロボット」の製作を行った（図１）。

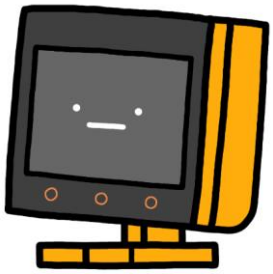

図１　トモダチロボット

２．研究の具体的内容

（１）製作計画

次のような年間製作計画を立てた（表１）。

表１　製作計画

４月	研究テーマの案出し・決定
５月	設計・Arduino IDE,M5Stackについて調査
６月	使用機材の調達
７月	M5Stackの身体となる部分を３Dプリンタで製作
８月	サーボモータのプログラム制御
９月	表情を画面に表示
１０月	話す・聞く機能をプログラム
１１月	岡工祭に向けて微調整
１２月	話す、聞く、動く、表すの４つの機能を並列処理
１月	発表会に向けて微調整

（２）使用機器・材料

＜ハードウェア＞

・M5Stack Core2 for AWS
　ESP32 IoT 開発キット（図２）
・ケース３点（3D プリンタを使用）
・マイクロサーボ（MG90S）×２個
・Grove 互換 4P（HY-2.0-4P）-2.54mm4P
　コネクタ変換ケーブル
・ユニバーサル基板 穴４×７
・両端ロングピンヘッダ３pin×２個
・ピンソケット４pin
・電解コンデンサ 6.3V 220μF
・セラミックコンデンサ 50V 0.1μF
・M1.4 のネジ×８本
・十字サーボホーン×２個

＜開発環境＞

・Arduino IDE

図２　M5Stack Core2 for AWS

（３）M5Stack Core2 for AWS について

今回私たちは、トモダチロボットを製作する上で「M5Stack Core2 for AWS」（図２）を使用した。これは５cm×５cm と非常にコンパクトなサイズの小型のマイコンモジュールである。小さな筐体に、バッテリや液晶ディスプレイ、タッチスクリーン、マイク、スピー

カなどの多くの機能が搭載されている。また、「ESP32」というマイコンが入っているためWi-Fi や Bluetooth などの、無線通信も可能である。

（４）ケースの製作

ケースの各パーツはインターネット上で公開されているものを参考に、次の手順で製作した。

1) 123D Design でシェル、ブラケット、足の各パーツを作る（図３）。

・ブラケット　サーボモータを固定する中身
・シェル　　外側のケース
・足　　足の部分

2) XYZware pro で 3D プリンタに各パーツのデータを送り、成型する。

3) サポートをはがし完成。

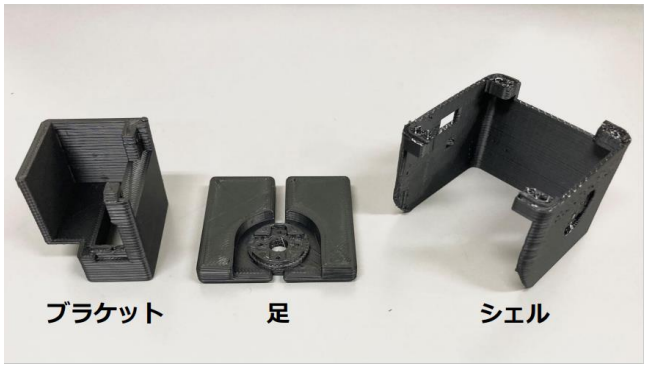

図３　ケースの各パーツ

（５）組み立てーその１ー

トモダチロボットのケースが完成した後、まずは、サーボモータを動かすための部品を次の手順で組み立てた。詳しくは「スタックチャンの作り方　M5Burner 版　分解なし【後半】」という動画を参考にした。

当初はミニブレッドボードを使用していたが、コンパクト化のためにユニバーサル基板を使用し、半田付けをした（図４）。

1) 穴が４×７個空いているユニバーサル基板に両端ロングピンヘッダ３pin を２個半田付けする。

2) ピンソケット４pin を半田付けし、Grove 互換-2.54mm 変換ケーブルを接続する。

3) 電解コンデンサとセラミックコンデンサのリード線を適当な長さに切断し、半田付けする。VCC-GND 間に並列になるようにする。

※電解コンデンサには極性があり、方向を間違えると壊れるので十分注意する。

4) メッキ線で配線する。

5) ピンヘッダ３pin にサーボモータのコネクタを接続する。

6) 1) ～5) が終わったら、テストプログラムを用いて、サーボモータの動作確認を行う。

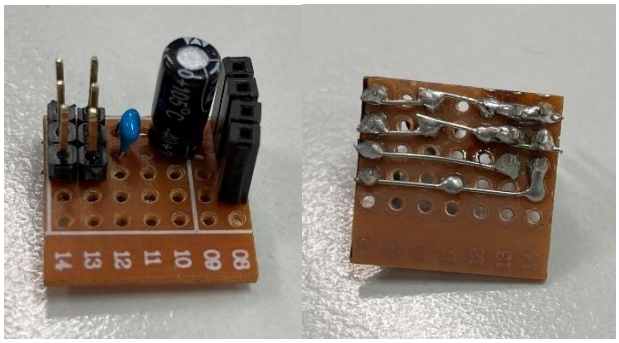

図４　半田付けされた基板

（６）組み立てーその２ー

続いて、ケースの組み立てを次の手順で行った。

1) M5Stack を裏返して、四隅のボルトを外してバッテリ ボトムを取り外す（図５）。

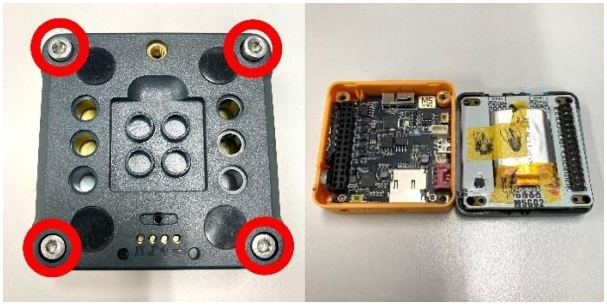

図５　バッテリ ボトムを取り外す

2) 四隅のネジを取り外して基板を取り外す（図６）。

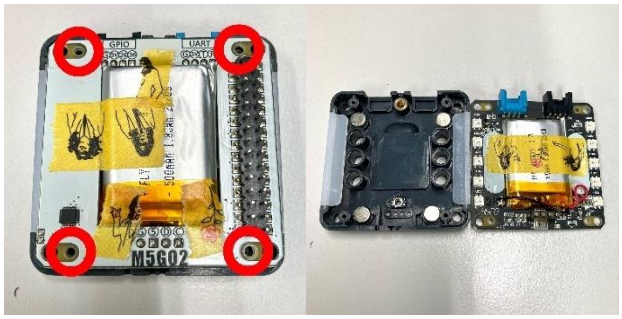

図６　基板を取り外す

3) バッテリをマスキングテープで固定する。
4) 取り外した基板をシェルに取り付け、Grove 互換-2.54mm 変換ケーブルを Port.C に接続する（図７）。

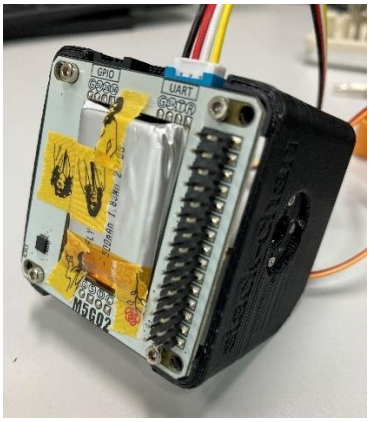

図７　取り付けられた基板

5) 十字サーボボーンを適当な長さにカットし、シェルに M1.4 のネジ４本で固定する。
6) シェルと M5Stack を固定する
7) 足に 5) と同様のカットした十字サーボボーンを M1.4 のネジ４本で固定する(図８)。

図８　足にサーボボーンを取り付ける

8) ブラケットに水平方向のサーボモータを取り付ける（図９）。

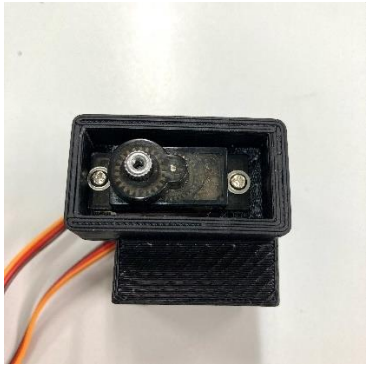

図９　水平方向のサーボモータ

9) 次に垂直方向のサーボモータを取り付ける（図 10)。

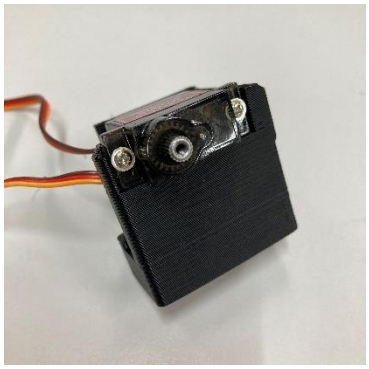

図 10　垂直方向のサーボモータ

10) ブラケットと足を組み合わせる。足の裏側からサーボ付属のネジでサーボボーンとサーボモータを固定する。
11) 先ほど 10) で組み合わせたものとシェルを組み合わせる。サーボ付属のネジでサーボボーンとサーボを固定する。

（７）話す

当初の予定では、先に動きを制御するプログラムを組む予定であった。しかし、ケースの製作に並行して行うことができる、話すプログラムの作成を先に行った。

Github で公開されている「Aquestalk-M5Unified」を参考に音声出力やディスプレイを制御するプログラムを作成した。音声は AquesTalk というミドルウェアを使用し、ボ

タン操作や文字を入力することでトモダチロボットがしゃべるようプログラミングを行った（図 11）。

```
static void talk_task(void*) {
  int16_t wav[3][LEN_FRAME];
  int tri_index = 0;
  for(;;) {
    ulTaskNotifyTake(pdTRUE, portMAX_DELAY); // wait notify
    while(is_talking) {
      uint16_t len;
      if (CAqTkPicoF_SyntheFrame(wav[tri_index], &len)) { is_talking = false; break; }
      M5.Speaker.playRaw(wav[tri_index], len, 8000, false, 1, m5spk_virtual_channel, false);
      tri_index = tri_index < 2 ? tri_index + 1 : 0;

      // lipsync
      float f = abs(wav[tri_index][0]) / 12000.0;
      float open = min(1.0f, f);
      avatar.setMouthOpenRatio(open);
    }
    is_moving = true;
    xTaskNotifyGive(task_handle2);
  }
}

// 音声再生の終了を待機する
static void waitAquesTalk(void) {
  while(is_talking) { vTaskDelay(1); }
}

// 音声再生を停止する
static void stopAquesTalk(void) {
  if(is_talking) { is_talking = false; is_moving = false; vTaskDelay(1); }
}

// 音声再生を開始する（再生中の場合は中断して新たな音声再生を開始する）
static void playAquesTalk(const char *koe) {
  stopAquesTalk();

  int iret = CAqTkPicoF_SetKoe((const uint8_t*)koe, 100, 0xFFu);
  if(iret) {M5.Display.println("ERR:CAqTkPicoF_SetKoe");}

  is_talking = true;
  xTaskNotifyGive(task_handle);
}
```

図 11　話すプログラム

入力した文字データは、シリアル通信によって M5Stack で取得した。std::string 型で受信をして、c_str 関数を使い、const char*型に変換を行った（図 12）。

```
std::string stext;
stext = Serial.readStringUntil('¥n');
Serial.print("Input:");
Serial.println(stext);

playAquesTalk(text.c_str());
```

図 12　文字受信プログラム

（８）動く

次に、Github で公開されている「stackchan-tester-main」を参考にサーボモータを制御するプログラムを作成した。サーボモータは MG90S を使用することで、ボディを 180 度回転することができる。random 関数を使い、上下左右、ランダムに動くようプログラミングを行った（図 13）。

文化祭では、ボタン操作でしか動くことができなかったが、（９）に記述している並列処理を行うことで常にランダムに動くことができるよう改良を加えた。

```
static void moveXY(int x, int y, uint32_t millis_for_move = 0) {
  if (millis_for_move == 0) {
    servo_x.easeTo(x + servo_offset_x);
    servo_y.easeTo(y + servo_offset_y);
  } else {
    servo_x.easeToD(x + servo_offset_x, millis_for_move);
    servo_y.easeToD(y + servo_offset_y, millis_for_move);
  }
  // サーボが停止するまでウェイトします。
  synchronizeAllServosStartAndWaitForAllServosToStop();
}

static void moveRandom(void*) {
  for(;;) {
    ulTaskNotifyTake(pdTRUE, portMAX_DELAY); // wait notify
    while(is_moving) {
    // ランダムモード
      int x = random(45, 135);  // 45～135°　でランダム
      int y = random(50, 90);   // 50～90°　でランダム
       int delay_time = random(10);
      moveXY(x, y, 1000 + 100 * delay_time);
      delay(10 + 50 * delay_time);
    }
  }
}
```

図 13　動くプログラム

（９）並列処理

本来 M5Stack では、並列処理ができない。そのため、タスクを使うことで「話す」と「動く」の並列処理を可能にした（図 14）。

```
xTaskCreateUniversal(talk_task, "talk_task", 4096, nullptr, 1, &task_handle,
APP_CPU_NUM);
xTaskCreateUniversal(moveRandom, "moveRandom", 4096, nullptr, 1, &task_handle2,
APP_CPU_NUM);


xTaskNotifyGive(task_handle);
xTaskNotifyGive(task_handle2);


ulTaskNotifyTake(pdTRUE, portMAX_DELAY);
```

図 14　並列処理プログラム

（10）完成

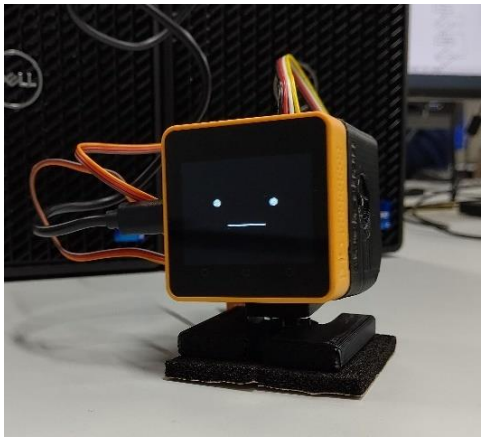

図 15　　完成（仮）

３．研究のまとめ

せっかくメンバーが５人いるにも関わらず役割分担がしっかりとできていなかったことが大きな反省点となった。そのため、最初の計画からかなりずれてしまい､「聞く」「表す」の機能を実装することができなかった。トモダチロボットのケースはかなりスムーズに成型をすることができた｡しかしプログラムは、インターネット上のプログラムや入門書等を参考にプログラミングを行ったが、想定と異なる処理を行ったり、エラー文が大量に出たりしたため、修正をすることが大変だった。

特に、サーボモータが動かず、半月ほどかけて原因を探り、やっと動いたときは感動した。また、トモダチロボットを稼働させているとき、突然サーボモータがうなりだしたのには大変恐怖を覚えた。

今回は、時間が足りず当初つける予定であった機能が実装できなかったため、今後は実装予定であった機能をつけて、トモダチロボットを改良していこうと考えている。

最後に、この課題研究を通して、改めてモノづくりの大変さを思い知った。課題研究で学んだいろいろな気付きを、今後の人生に生かしていけたらと思う。

４．参考文献

＜ハード＞

○サーボモータ

スタックチャン　タカオ版　組み立て方法【その１部品キット編】
https://raspberrypi.mongonta.com/how-to-make-stackchan-m5gobottom/

○ケース、組み立て

スタックチャン　タカオ版　組み立て方法【その２ケースセットの組み立てと完成まで】【分解あり】
https://raspberrypi.mongonta.com/how-to-make-stackchan-m5gobottom-2/

○ボディのデータ

https://github.com/meganetaaan/stack-chan/tree/main/case/contributed/mongonta_case_for_SG90_and_M5GoBottomBoard/case_for_SG90andM5GoBottomBoard

＜ソフト＞

○開発環境

初心者向けM5Stackの始め方(ArduinoIDE編)
https://raspberrypi.mongonta.com/howto-start-m5stack-arduinoide/

ボードマネージャ「ESP32」のインストール
https://www.abit.co.jp/iot/ak030/manual/2-arduino-ide/3-install-arduino-core-esp32/

○話す

M5StackでAquesTalkを使って音声出力する
https://yamaccu.github.io/tils/20220829-M5Stack-AquesTalk

プログラム
https://github.com/yamaccu/M5Stack-Sample/tree/main/Aquestalk-M5Unified

AquesTalkの使い方
http://blog-yama.a-quest.com/?eid=970195

AquesTalkダウンロード
https://www.a-quest.com/products/aquestalk.html

シリアル通信
https://qiita.com/hsgucci/items/eee5894e3651d0a8cb75

「const char*」と「std::string」の変換
https://www.paveway.info/entry/2018/12/25/cpp_std_string_char

○サーボモータ
スタックチャン　M5GoBottom 版のファームウェアについて
https://raspberrypi.mongonta.com/softwares-for-stackchan/
スタックチャンを ArduinoFramework でスムーズに動かす［PWM サーボ版］
https://raspberrypi.mongonta.com/how-to-move-with-sg90-arduinoframework/
プログラム
https://github.com/mongonta0716/stack-chan-tester/tree/main/src

○並列処理
https://lang-ship.com/blog/work/esp32-freertos-l02-taskcreate/