Arduino

Generated by Doxygen 1.8.16

1 TDF_02-145 Arduino 1

1.1 Introduction . . . 1

1.2 Wiring . . . 1

1.2.1 Motor Wiring . . . 1

1.2.2 Encoder Wiring . . . 1

1.2.3 Servo Wiring . . . 2

1.2.3.1 Authors . . . 2

1.2.3.2 Version . . . 2

2 File Index 3

2.1 File List . . . 3

3 File Documentation 5

3.1 decider.ino File Reference . . . 5

3.1.1 Detailed Description . . . 5

3.1.2 Function Documentation . . . 5

3.1.2.1 decider() . . . 6

3.1.2.2 moveBackward() . . . 7

3.1.2.3 moveForward() . . . 7

3.1.2.4 turnLeft() . . . 8

3.1.2.5 turnRight() . . . 8

3.2 encoderFunctions.ino File Reference . . . 9

3.2.1 Detailed Description . . . 9

3.2.2 Function Documentation . . . 9

3.2.2.1 encoderLeft() . . . 9

3.2.2.2 encoderRight() . . . 10

3.3 headMovement.ino File Reference . . . 10

3.3.1 Detailed Description . . . 10

3.3.2 Function Documentation . . . 10

3.3.2.1 headMotion_predefined() . . . 10

3.3.2.2 movehead() . . . 11

3.4 move_Direction.ino File Reference . . . 12

3.4.1 Detailed Description . . . 12

3.4.2 Function Documentation . . . 12

3.4.2.1 motor() . . . 12

3.4.2.2 moveDirection() . . . 14

3.5 PID_Initial.ino File Reference . . . 15

3.5.1 Detailed Description . . . 15

3.5.2 Function Documentation . . . 15

3.5.2.1 PID_initial() . . . 15

3.6 printStatus.ino File Reference . . . 16

3.6.1 Detailed Description . . . 16

3.6.2 Function Documentation . . . 16

3.6.2.1 motionText() . . . 16
3.6.2.2 printStatus() . . . 17
3.7 README.md File Reference . . . 18
3.8 serialEvent.ino File Reference . . . 18
3.8.1 Detailed Description . . . 18
3.8.2 Function Documentation . . . 18
3.8.2.1 serialEvent() . . . 18
3.8.2.2 serialEvent2() . . . 19
3.9 stopMotion.ino File Reference . . . 19
3.9.1 Detailed Description . . . 19
3.9.2 Function Documentation . . . 19
3.9.2.1 stopMotion() . . . 20
3.10 TDF02-145_Arduino.ino File Reference . . . 20
3.10.1 Detailed Description . . . 21
3.10.2 Macro Definition Documentation . . . 21
3.10.2.1 CCW . . . 22
3.10.2.2 CW . . . 22
3.10.2.3 encoder_L_C1 . . . 22
3.10.2.4 encoder_R_C1 . . . 22
3.10.2.5 Front . . . 22
3.10.2.6 KD . . . 23
3.10.2.7 KI . . . 23
3.10.2.8 KP . . . 23
3.10.2.9 Left . . . 23
3.10.2.10 maxpitchlimit . . . 23
3.10.2.11 maxyawlimit . . . 24
3.10.2.12 minpitchlimit . . . 24
3.10.2.13 minyawlimit . . . 24
3.10.2.14 motor_ENA . . . 24
3.10.2.15 motor_ENB . . . 24
3.10.2.16 motor_IN1 . . . 25
3.10.2.17 motor_IN2 . . . 25
3.10.2.18 motor_IN3 . . . 25
3.10.2.19 motor_IN4 . . . 25
3.10.2.20 Rear . . . 25
3.10.2.21 Right . . . 26
3.10.2.22 robot_speed_max . . . 26
3.10.2.23 robot_speed_min . . . 26
3.10.2.24 servoPitchCenter . . . 26
3.10.2.25 servoPitchPin . . . 26
3.10.2.26 servoYawCenter . . . 27
3.10.2.27 servoYawPin . . . 27

3.10.2.28 Start 27
3.10.2.29 Stop 27
3.10.3 Function Documentation 27
3.10.3.1 loop() 28
3.10.3.2 setup() 28
3.10.4 Variable Documentation 28
3.10.4.1 countLeft 28
3.10.4.2 countRight 29
3.10.4.3 currentMotion 29
3.10.4.4 inputString 29
3.10.4.5 leftPID 29
3.10.4.6 pitchservo 30
3.10.4.7 pwmLeft 30
3.10.4.8 pwmRight 30
3.10.4.9 rightPID 30
3.10.4.10 stepsMotion 31
3.10.4.11 stepsOverall 31
3.10.4.12 yawservo 31

Index **33**

Chapter 1

TDF_02-145 Arduino

1.1 Introduction

This is the repository for arduino and hardware related matters for robot for Autism Spectrum Disorder therapy development funded through HEC TDF. The documentation folder contains all the code for Arduino division of HEC funded project TDF 02-145. There are multiple files contained in this folder. This code will run on an Arduino Mega.

1.2 Wiring

This section details the Arduino wiring.

1.2.1 Motor Wiring

- ENA pin of L298N motor driver is connected to Arduino pin 8
- ENB pin of L298N motor driver is connected to Arduino pin 9
- IN1 pin of L298N motor driver is connected to Arduino pin 11
- IN2 pin of L298N motor driver is connected to Arduino pin 10
- IN3 pin of L298N motor driver is connected to Arduino pin 12
- IN4 pin of L298N motor driver is connected to Arduino pin 13

1.2.2 Encoder Wiring

- Encoder pin from left motor is connected to Arduino pin 3
- Encoder pin from right motor is connected to Arduino pin 2

1.2.3 Servo Wiring

- Servo for pitch movement (nodding of head - like when you indicate yes) of head is connected to Arduino pin 6
- Servo for yaw movement (shaking of head - like when you indicate no) of head is connected to Arduino pin 5

1.2.3.1 Authors

Taha Shaheen, Muhammad Hashir bin Khalid, Abdul Samad

1.2.3.2 Version

chotuX

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

decider.ino
: Decider of things . . . 5

encoderFunctions.ino
: Interrupt functions . . . 9

headMovement.ino
: Controls the head . . . 10

move_Direction.ino
: Responsible for motor control . . . 12

PID_Initial.ino
: Sets up the PID algorithm . . . 15

printStatus.ino
: A debugging option . . . 16

serialEvent.ino
: Handles serial communication . . . 18

stopMotion.ino
: Stops robot . . . 19

TDF02-145_Arduino.ino
: Sets everything up . . . 20

Chapter 3

File Documentation

3.1 decider.ino File Reference

Decider of things.

Functions

- void decider ()

  Handles all decision matters.
- void turnLeft (double steps)

  The robot turns left.
- void turnRight (double steps)

  The robot turns right.
- void moveForward (double steps)

  The robot begins to move forward.
- void moveBackward (double steps)

  The robot begins to move backwards.

3.1.1 Detailed Description

Decider of things.

Contains code pertaining to all decision flow for Chotu's locomotion and instruction handling.

3.1.2 Function Documentation

3.1.2.1 decider()

```
void decider ( )
```

Handles all decision matters.

- Updates stepsOverall which is the count of steps taken on average by the encoders from the time the robot was started until now
- Breaks the inputString into its constituent commands
- Interprets what the command means and who is to execute it
- If command pertains to locomotion or servo motion, Arduino handles its
- If command pertains to anything else, it is sent as is to the tablet
- When decider() is called, it nullifies any previous command.

Returns

void

Definition at line 20 of file decider.ino.

```
20              {
21
22    //stepsMotion stores the steps taken until now (also discards accidental movement readings) //
23    stepsMotion = stepsOverall;
24
25    //Holds the command character. For locomotion commands this is the direction to move in.//
26    char Direction;
27
28    //Holds the locomotion parameter which is in units pertaining to movement.//
29    int LocomotionParameter;
30
31    //Hold angles for the Yaw and Pitch motors.//
32    int YawAngle;
33    int PitchAngle;
34
35    //Holds the status of the PIR sensor?//
36    char PIRstatus;
37
38    //eg. "F"_0200_090_045 //
39    Direction = inputString.charAt(0);
40
41    //eg. F_"0200"_090_045 //
42    LocomotionParameter = (inputString.substring(2, 6)).toInt();
43
44    //eg. F_0200_"090_045 //
45    YawAngle = (inputString.substring(7, 10)).toInt();
46
47    //eg. F_0200_090_"045" //
48    PitchAngle = (inputString.substring(11, 14)).toInt();
49
50    // Stores the input String for a small check later //
51    String inputStringStored = inputString;
52
53    //Empties the inputString //
54    inputString = "";
55
56    //movehead(YawAngle, PitchAngle); //commented as no servos currently present in Chotu body
57
58    switch (Direction) {
59        case 'F':
60            moveForward(LocomotionParameter);
61            break;
62        case 'B':
63            moveBackward(LocomotionParameter);
64            break;
65        case 'R':
66            turnRight(LocomotionParameter);
67            break;
68        case 'L':
69            turnLeft(LocomotionParameter);
```

```
70              break;
71          case 'E':
72          case 'G':
73          case 'C':
74          //Letters that are not locomotion related. Other characters will stop the robot motion. This is a
        safety feature.//
75              if (inputStringStored == "#") /*checks that it isn't sending an empty command*/ {
76              } else
77                  Serial1.print(inputStringStored);
78              stopMotion();
79              //commented for now as no PIR sensor present in body
80              //    case 'P':
81              //      pirSensor();
82              //      PIRstatus = pirReturn ();
83              //      datatosend = "P_";
84              //      datatosend += PIRstatus;
85              //      datatosend += "_000_000#";
86              //      Serial1.println(datatosend);
87              //      break;
88          default:
89              stopMotion();
90      }
91  }
```

References inputString, moveBackward(), moveForward(), stepsMotion, stepsOverall, stopMotion(), turnLeft(), and turnRight().

Referenced by serialEvent(), and serialEvent2().

3.1.2.2 moveBackward()

```
void moveBackward (
            double steps )
```

The robot begins to move backwards.

Parameters

steps	Motion parameter

Returns

void

Definition at line 122 of file decider.ino.

```
122                                     {
123     moveDirection(Rear, steps);
124 }
```

Referenced by decider().

3.1.2.3 moveForward()

```
void moveForward (
            double steps )
```

The robot begins to move forward.

Parameters

steps	Motion parameter

Returns

void

Definition at line 114 of file decider.ino.

```
114                                 {
115     moveDirection(Front, steps);
116 }
```

Referenced by decider().

3.1.2.4 turnLeft()

```
void turnLeft (
              double steps )
```

The robot turns left.

Parameters

steps	Motion parameter

Returns

void

Definition at line 98 of file decider.ino.

```
98                                  {
99     moveDirection(Left, steps);
100 }
```

References Left, and moveDirection().

Referenced by decider().

3.1.2.5 turnRight()

```
void turnRight (
              double steps )
```

The robot turns right.

Parameters

steps	Motion parameter

Returns

void

Definition at line 106 of file decider.ino.

```
106                                 {
107     moveDirection(Right, steps);
108 }
```

Referenced by decider().

3.2 encoderFunctions.ino File Reference

Interrupt functions.

Functions

- void encoderLeft ()
  Interrupt function.
- void encoderRight ()
  Interrupt function.

3.2.1 Detailed Description

Interrupt functions.

This file contains the functions that get called when the interrupt event occurs because of a rising signal from the encoders in the motors.

3.2.2 Function Documentation

3.2.2.1 encoderLeft()

```
void encoderLeft ( )
```

Interrupt function.

This function gets called when a rising pulse is received from the encoder on the left motor.

Returns

void

Definition at line 12 of file encoderFunctions.ino.

```
12                      {
13   countLeft++;
14
15   // Average of countLeft and countRight //
16   stepsOverall = (countLeft + countRight) / 2;
17 }
```

References countLeft, countRight, and stepsOverall.

Referenced by PID_initial().

3.2.2.2 encoderRight()

```
void encoderRight ( )
```

Interrupt function.

This function gets called when a rising pulse is received from the encoder on the right motor.

Definition at line 23 of file encoderFunctions.ino.

```
23                     {
24   countRight++;
25
26   // Average of countLeft and countRight //
27   stepsOverall = (countLeft + countRight) / 2;
28 }
```

References countLeft, countRight, and stepsOverall.

Referenced by PID_initial().

3.3 headMovement.ino File Reference

Controls the head.

Functions

- void movehead (int yaw, int pitch)

  Tasked with moving both servos.
- void headMotion_predefined (char Direction, int Speed, int Duration, int Empty)

  Predefined head movements (nodding and shaking for some time)

3.3.1 Detailed Description

Controls the head.

Writes angles to the servos in the neck.

3.3.2 Function Documentation

3.3.2.1 headMotion_predefined()

```
void headMotion_predefined (
             char Direction,
             int Speed,
             int Duration,
             int Empty )
```

Predefined head movements (nodding and shaking for some time)

Parameters

Direction	Nodding ('Y') or shaking ('N')
Speed	Speed with which to move the head
Duration	Time in seconds for which to move the head
Empty	Empty parameter. Exists to keep with the format of the inputString.

Returns

void

Definition at line 43 of file headMovement.ino.

```
43                                                                                {
44   unsigned long previousMillis;
45   const long interval = Duration * 1000;
46   unsigned long currentMillis = millis();
47   previousMillis = currentMillis;
48
49   // loop based on millis() passed //
50   while (abs(currentMillis - previousMillis) <= interval) {
51
52     currentMillis = millis();
53
54     switch (Direction) {
55
56       case 'Y':
57         // nodding //
58         yawservo.write(servoYawCenter, 255, true); // Takes head to center first //
59         pitchservo.write(servoPitchCenter + 25, Speed/*100 works best*/, true);
60         delay(500);
61         yawservo.write(servoYawCenter, 255, true);
62         pitchservo.write(servoPitchCenter - 25 /*This number can be changed*/, Speed, true);
63         delay(500);
64         break;
65
66       case 'N':
67         //head shaking
68         pitchservo.write(servoPitchCenter, 255, true);
69         yawservo.write(servoYawCenter + 15, Speed /*40 works best*/, true);
70         delay(500);
71         pitchservo.write(servoPitchCenter, 255, true);
72         yawservo.write(servoYawCenter - 15  /*This number can be changed*/, Speed, true);
73         delay(500);
74         break;
75     }
76
77     currentMillis = millis();
78   }
79
80   // Setting it back to center facing //
81   pitchservo.write(servoPitchCenter, 255, true); //pitch = yes, nodding motion
82   yawservo.write(servoYawCenter, 255, true); //yaw = no, shaking motion
83 }
```

References pitchservo, servoPitchCenter, servoYawCenter, and yawservo.

3.3.2.2 movehead()

```
void movehead (
              int yaw,
              int pitch )
```

Tasked with moving both servos.

Parameters

yaw	The yaw angle in degrees extracted in decider()
pitch	The pitch angle in degrees extracted in decider()

Returns

void

Definition at line 16 of file headMovement.ino.

```
16                                  {
17   if (yaw > maxyawlimit)
18     yaw = maxyawlimit;
19   else if (yaw < minyawlimit)
20     yaw = minyawlimit;
21
22   if (pitch > maxpitchlimit)
23     pitch = maxpitchlimit;
24   else if (pitch < minpitchlimit)
25     pitch = minpitchlimit;
26
27   // Writes the angle to the servo at a "100" speed //
28
29   pitchservo.write(pitch, 100/*, 255, true*/); //pitch = yes, nodding motion
30   yawservo.write(yaw, 100/*,255, true*/); //yaw = no, shaking motion
31 }
```

References maxpitchlimit, maxyawlimit, minpitchlimit, minyawlimit, pitchservo, and yawservo.

3.4 move_Direction.ino File Reference

Responsible for motor control.

Functions

- void moveDirection (int Direction, double steps)

  Handles the motor PWM and directions.
- void motor (int LeftRight, int pwm, int CWCCW)

  Controls both motors' angular direction and their speed.

3.4.1 Detailed Description

Responsible for motor control.

Contains code pertaining to movement and how much motion needs to happen.

3.4.2 Function Documentation

3.4.2.1 motor()

```
void motor (
            int LeftRight,
            int pwm,
            int CWCCW )
```

Controls both motors' angular direction and their speed.

Parameters

LeftRight	Defines which motor
pwm	PWM signal to control motor speed
CWCCW	CW=ClockWise, CCW = CounterclockWise. The direction of the motor when looking at it down the shaft from the wheel's end.

Returns

void

Definition at line 83 of file move_Direction.ino.

```
83                                                {
84   switch (LeftRight) {
85     case Right:
86       switch (CWCCW) {
87         case CW:
88           digitalWrite(motor_IN1, HIGH);
89           digitalWrite(motor_IN2, LOW);
90           break;
91         case CCW:
92           digitalWrite(motor_IN1, LOW);
93           digitalWrite(motor_IN2, HIGH);
94           break;
95       }
96
97       // Commented to disable PID //
98       // digitalWrite(motor_ENA,HIGH);
99       // analogWrite(motor_ENA, pwm);
100
101       // The following code replaces PID. Only works because both motors are mechanically similar. //
102       if (pwm > 0)
103         analogWrite(motor_ENA, 255);
104       else
105         analogWrite(motor_ENA, 0);
106       break;
107
108     case Left:
109       switch (CWCCW) {
110         case CW:
111           digitalWrite(motor_IN3, HIGH);
112           digitalWrite(motor_IN4, LOW);
113           break;
114         case CCW:
115           digitalWrite(motor_IN3, LOW);
116           digitalWrite(motor_IN4, HIGH);
117           break;
118       }
119
120       // Commented to disable PID //
121       // analogWrite(motor_ENB, pwm);
122
123       // The following code replaces PID. Only works because both motors are mechanically similar. //
124       if (pwm > 0)
125         analogWrite(motor_ENB, 255);
126       else
127         analogWrite(motor_ENB, 0);
128       break;
129   }
130 }
```

References CCW, CW, Left, motor_ENA, motor_ENB, motor_IN1, motor_IN2, motor_IN3, motor_IN4, and Right.

Referenced by moveDirection(), and stopMotion().

3.4.2.2 moveDirection()

```
void moveDirection (
            int Direction,
            double steps )
```

Handles the motor PWM and directions.

Based on what the direction is, it decides which motor needs to be going CW (clockwise) and which CCW (counter-clockwise).

Parameters

Direction	Front, Rear, Left, Right
steps	Motion Parameter

Returns

void

Definition at line 17 of file move_Direction.ino.

```
17                                                  {
18
19   // Uncomment for debugging //
20   //Serial.print(motionText(currentMotion));
21   //Serial.println(motionText(Direction));
22
23   // stepsOverall increases until the difference between stepsMotion (steps taken while moving until now)
       and it becomes greater than the steps the robot needs to take //
24   while (stepsOverall - stepsMotion < steps && !Serial2.available()) {
25
26     // Uncomment for debugging //
27     //Serial2.println("stepsOverall - stepsMotion < steps");
28     //Serial2.print(pwmRight); Serial2.print(" "); Serial2.print(pwmLeft); Serial2.print(" | ");
       Serial2.print(countRight); Serial2.print(" "); Serial2.print(countLeft); Serial2.print(" ");
       Serial2.print(stepsOverall); Serial2.println(" | ");
29
30     if (currentMotion == Direction) {
31       // Check to save from applying PID while changing direction. //
32
33       // runs PID and updates pwmLeft and pwmRight//
34       rightPID.run();
35       leftPID.run();
36
37       switch (Direction) {
38         // Applying updated PWM values //
39         case Front:
40           motor(Right, pwmRight, CCW);
41           motor(Left, pwmLeft, CW);
42           break;
43
44         case Rear:
45           motor(Right, pwmRight, CW);
46           motor(Left, pwmLeft, CCW);
47           break;
48
49         case Right:
50           motor(Right, pwmRight, CW);
51           motor(Left, pwmLeft, CW);
52           break;
53
54         case Left:
55           motor(Right, pwmRight, CCW);
56           motor(Left, pwmLeft, CCW);
57           break;
58       }
59     } else {
60
61       // changes direction //
62       currentMotion = Direction;
63       // saves steps so far to stepsMotion setting difference to zero //
64       stepsMotion = stepsOverall;
65
```

```
66        // Uncomment for debugging //
67        //Serial2.println("currentMotion = Direction; stepsMotion = stepsOverall;");
68      }
69    }
70    stopMotion();
71    //Serial2.println("stepsOverall - stepsMotion >= steps  stopMotion(); stepsMotion = stepsOverall;");
72  }
```

References CCW, currentMotion, CW, Front, Left, leftPID, motor(), pwmLeft, pwmRight, Rear, Right, rightPID, stepsMotion, stepsOverall, and stopMotion().

Referenced by turnLeft().

3.5 PID_Initial.ino File Reference

Sets up the PID algorithm.

Functions

- void PID_initial ()
  sets up PID

3.5.1 Detailed Description

Sets up the PID algorithm.

Contains one function. Initializes pins and interrupt events from encoder needed for the PID algorithm.

3.5.2 Function Documentation

3.5.2.1 PID_initial()

```
void PID_initial ( )
```

sets up PID

- Sets the Arduino pins connected to ENA, ENB, IN1, IN2, IN3 and IN4 to output.
- Sets the Arduino pins connected to encoders to input.
- Sets up interrupts on the encoder pins (these can only be set to specific pins on the Board). The interrupt is triggered on the RISING edge of the signal. Functions are called on this interrupt event.
- Sets the AutoPID objects leftPID and rightPID to have a time step in milliseconds for the PID calculation.

Returns

void

Definition at line 17 of file PID_Initial.ino.

```
17                  {
18     pinMode(motor_ENA, OUTPUT);
19     pinMode(motor_ENB, OUTPUT);
20     pinMode(motor_IN1, OUTPUT);
21     pinMode(motor_IN2, OUTPUT);
22     pinMode(motor_IN3, OUTPUT);
23     pinMode(motor_IN4, OUTPUT);
24
25     pinMode(encoder_L_C1, INPUT);
26     pinMode(encoder_R_C1, INPUT);
27
28     attachInterrupt(digitalPinToInterrupt(encoder_L_C1), encoderLeft, RISING);
29     attachInterrupt(digitalPinToInterrupt(encoder_R_C1), encoderRight, RISING);
30
31     leftPID.setTimeStep(4);
32     rightPID.setTimeStep(4);
33 }
```

References encoder_L_C1, encoder_R_C1, encoderLeft(), encoderRight(), leftPID, motor_ENA, motor_ENB, motor_IN1, motor_IN2, motor_IN3, motor_IN4, and rightPID.

Referenced by setup().

3.6 printStatus.ino File Reference

A debugging option.

Functions

- void printStatus ()

  Prints status on all serial communication channels.
- String motionText (int motionInteger)

  Converts the motion integer to text.

3.6.1 Detailed Description

A debugging option.

Runs in loop() and displays the movement status of the robot on all serial communication channels.

3.6.2 Function Documentation

3.6.2.1 motionText()

```
String motionText (
            int motionInteger )
```

Converts the motion integer to text.

Parameters

motionInteger	Front, Rear, Left, Right - integers from preprocessor directives

Returns

String

Definition at line 43 of file printStatus.ino.

```
43                                      {
44   switch (motionInteger) {
45     case Stop:
46       return "Stop";
47       break;
48     case Front:
49       return "Front";
50       break;
51     case Rear:
52       return "Rear";
53       break;
54     case Left:
55       return "Left";
56       break;
57     case Right:
58       return "Right";
59       break;
60   }
61 }
```

References Front, Left, Rear, Right, and Stop.

Referenced by printStatus().

3.6.2.2 printStatus()

```
void printStatus ( )
```

Prints status on all serial communication channels.

Prints

- countLeft
- countRight
- stepsOverall
- stepsMotion
- pwmLeft
- pwmRight

Definition at line 20 of file printStatus.ino.

```
20                      {
21     Serial2.print(motionText(currentMotion));
22     Serial2.print("; Count; L:");
23     Serial2.print(countLeft);
24     Serial2.print(", R:");
25     Serial2.print(countRight);
26     Serial2.print(", Steps; Overall:");
27     Serial2.print(stepsOverall);
28     Serial2.print(", Steps; Motion:");
29     Serial2.print(stepsMotion);
30     Serial2.print(", PWM; L:");
31     Serial2.print(pwmLeft);
32     Serial2.print(", R:");
33     Serial2.print(pwmRight);
34     Serial2.println();
35 }
```

References countLeft, countRight, currentMotion, motionText(), pwmLeft, pwmRight, stepsMotion, and stepsOverall.

3.7 README.md File Reference

3.8 serialEvent.ino File Reference

Handles serial communication.

Functions

- void serialEvent2 ()

  Gets called when there is an event on serial channel 2.
- void serialEvent ()

  Gets called when there is an event on serial channel 0.

3.8.1 Detailed Description

Handles serial communication.

Handles 3 serial communication trigger events. They get called whenever there is activity on any channel.

3.8.2 Function Documentation

3.8.2.1 serialEvent()

```
void serialEvent ( )
```

Gets called when there is an event on serial channel 0.

- runs until there is a communication line established
- stores incoming data into a character inChar
- appends inChar to inputString
- if the terminator '#' is received, the command is considered complete and decider() is called
- a confirmation of reception is given by sending the received command back to the sender

Returns

void

Definition at line 41 of file serialEvent.ino.

```
41                     {
42   while (Serial.available()) {
43     char inChar = (char)Serial.read();
44     inputString += inChar;
45     if (inChar == '#') {
46       //Serial2.println(inputString);
47       Serial.println(inputString);
48       decider();
49     }
50   }
51 }
```

References decider(), and inputString.

3.8.2.2 serialEvent2()

```
void serialEvent2 ( )
```

Gets called when there is an event on serial channel 2.

- runs until there is a communication line established
- stores incoming data into a character inChar2
- appends inChar2 to inputString
- if the terminator '#' is received, the command is considered complete and decider() is called
- a confirmation of reception is given by sending the received command back to the sender

Returns

void

Definition at line 19 of file serialEvent.ino.

```
19                      {
20   while (Serial2.available()) {
21     char inChar2 = (char)Serial2.read();
22     inputString += inChar2;
23     if (inChar2 == '#') {
24       //Serial2.println(inputString);
25       Serial2.println(inputString);
26       decider();
27     }
28   }
29 }
```

References decider(), and inputString.

3.9 stopMotion.ino File Reference

Stops robot.

Functions

- void stopMotion ()

  Commands both motors to stop spinning.

3.9.1 Detailed Description

Stops robot.

Stops all motion of the robot. Overwrites everything.

3.9.2 Function Documentation

3.9.2.1 stopMotion()

```
void stopMotion ( )
```

Commands both motors to stop spinning.

Returns

void

Definition at line 12 of file stopMotion.ino.

```
12                  {
13   motor(Left, Stop, Stop);
14   motor(Right, Stop, Stop);
15   currentMotion = Stop;
16
17   // Clear the enocder counters. //
18   countLeft = 0;
19   countRight = 0;
20 }
```

References countLeft, countRight, currentMotion, Left, motor(), Right, and Stop.

Referenced by decider(), and moveDirection().

3.10 TDF02-145_Arduino.ino File Reference

Sets everything up.

```
#include <AutoPID.h>
#include <VarSpeedServo.h>
```

Macros

- #define robot_speed_min 200
- #define robot_speed_max 255
- #define KP 0.012
- #define KI 0.03
- #define KD 0.0001
- #define motor_ENA 8
- #define motor_ENB 9
- #define motor_IN1 11
- #define motor_IN2 10
- #define motor_IN3 12
- #define motor_IN4 13
- #define encoder_L_C1 3
- #define encoder_R_C1 2
- #define Front 1
- #define Rear 2
- #define Right 3
- #define Left 4
- #define Stop 0
- #define Start 1
- #define CW 2
- #define CCW 3
- #define servoPitchPin 6
- #define servoYawPin 5
- #define servoPitchCenter 70
- #define servoYawCenter 80
- #define maxpitchlimit 180
- #define minpitchlimit 55
- #define maxyawlimit 180
- #define minyawlimit 00

Functions

- void setup ()

  Runs only once when the robot starts up.
- void loop ()

  Loops constantly.

Variables

- double countLeft = 0

  Integer to keep count of encoder signals from the left motor.
- double countRight = 0

  Integer to keep count of encoder signals from the right motor.
- double stepsOverall = 0

  Integer to keep an average count of encoder signals from the both motors.
- double stepsMotion = 0

  Don't remember.
- double pwmLeft

  PWM output for left motor.
- double pwmRight

  PWM output for right motor.
- int currentMotion = 0

  Integer to contain the present state of locomotion.
- AutoPID leftPID = AutoPID(&countLeft, &countRight, &pwmLeft, robot_speed_min, robot_speed_max, KP, KI, KD)

  Constructor creates an instance of AutoPID named leftPID.
- AutoPID rightPID = AutoPID(&countRight, &countLeft, &pwmRight, robot_speed_min, robot_speed_max, KP, KI, KD)

  Constructor creates an instance of AutoPID named rightPID.
- String inputString = ""

  Carries command instruction string.
- VarSpeedServo pitchservo

  Pitch servo motor defined as a VarSpeedServo object.
- VarSpeedServo yawservo

  Yaw servo motor defined as a VarSpeedServo object.

3.10.1 Detailed Description

Sets everything up.

Main file. Contains setup() and loop(). Has all library inclusions, macros, and variables.

3.10.2 Macro Definition Documentation

3.10.2.1 CCW

```
#define CCW 3
```

Counterclockwise. An integer used in the decision flow for locomotion.

Definition at line 105 of file TDF02-145_Arduino.ino.

3.10.2.2 CW

```
#define CW 2
```

Clockwise. An integer used in the decision flow for locomotion.

Definition at line 104 of file TDF02-145_Arduino.ino.

3.10.2.3 encoder_L_C1

```
#define encoder_L_C1 3
```

Encoder pin from left motor is connected to Arduino pin 3

Definition at line 70 of file TDF02-145_Arduino.ino.

3.10.2.4 encoder_R_C1

```
#define encoder_R_C1 2
```

Encoder pin from right motor is connected to Arduino pin 2

Definition at line 71 of file TDF02-145_Arduino.ino.

3.10.2.5 Front

```
#define Front 1
```

An integer used in the decision flow for locomotion

Definition at line 98 of file TDF02-145_Arduino.ino.

3.10.2.6 KD

```
#define KD 0.0001
```

Derivative constant for PID

Definition at line 35 of file TDF02-145_Arduino.ino.

3.10.2.7 KI

```
#define KI 0.03
```

Integral constant for PID

Definition at line 34 of file TDF02-145_Arduino.ino.

3.10.2.8 KP

```
#define KP 0.012
```

Proportional constant for PID

-

Definition at line 33 of file TDF02-145_Arduino.ino.

3.10.2.9 Left

```
#define Left 4
```

An integer used in the decision flow for locomotion

Definition at line 101 of file TDF02-145_Arduino.ino.

3.10.2.10 maxpitchlimit

```
#define maxpitchlimit 180
```

Maximum angle allowed for servo

Definition at line 195 of file TDF02-145_Arduino.ino.

3.10.2.11 maxyawlimit

```
#define maxyawlimit 180
```

Maximum angle allowed for servo

Definition at line 197 of file TDF02-145_Arduino.ino.

3.10.2.12 minpitchlimit

```
#define minpitchlimit 55
```

Minimum angle allowed for servo

Definition at line 196 of file TDF02-145_Arduino.ino.

3.10.2.13 minyawlimit

```
#define minyawlimit 00
```

Minimum angle allowed for servo

Definition at line 198 of file TDF02-145_Arduino.ino.

3.10.2.14 motor_ENA

```
#define motor_ENA 8
```

ENA pin of L298N motor driver is connected to Arduino pin 8

Definition at line 56 of file TDF02-145_Arduino.ino.

3.10.2.15 motor_ENB

```
#define motor_ENB 9
```

ENB of L298N motor driver is connected to Arduino pin 9

Definition at line 57 of file TDF02-145_Arduino.ino.

3.10.2.16 motor_IN1

```
#define motor_IN1 11
```

IN1 of L298N motor driver is connected to Arduino pin 11

Definition at line 58 of file TDF02-145_Arduino.ino.

3.10.2.17 motor_IN2

```
#define motor_IN2 10
```

IN2 of L298N motor driver is connected to Arduino pin 10

Definition at line 59 of file TDF02-145_Arduino.ino.

3.10.2.18 motor_IN3

```
#define motor_IN3 12
```

IN3 of L298N motor driver is connected to Arduino pin 12

Definition at line 60 of file TDF02-145_Arduino.ino.

3.10.2.19 motor_IN4

```
#define motor_IN4 13
```

IN4 of L298N motor driver is connected to Arduino pin 13

Definition at line 61 of file TDF02-145_Arduino.ino.

3.10.2.20 Rear

```
#define Rear 2
```

An integer used in the decision flow for locomotion

Definition at line 99 of file TDF02-145_Arduino.ino.

3.10.2.21 Right

```
#define Right 3
```

An integer used in the decision flow for locomotion

Definition at line 100 of file TDF02-145_Arduino.ino.

3.10.2.22 robot_speed_max

```
#define robot_speed_max 255
```

Maximum PWM for speed. It can range from 0 to 255.

Definition at line 21 of file TDF02-145_Arduino.ino.

3.10.2.23 robot_speed_min

```
#define robot_speed_min 200
```

Minimum PWM for speed. It can range from 0 to 255.

Definition at line 20 of file TDF02-145_Arduino.ino.

3.10.2.24 servoPitchCenter

```
#define servoPitchCenter 70
```

Central angle in degrees

Definition at line 193 of file TDF02-145_Arduino.ino.

3.10.2.25 servoPitchPin

```
#define servoPitchPin 6
```

Servo for pitch movement (nodding of head - like when you indicate yes) of head is connected to Arduino pin 6

Definition at line 174 of file TDF02-145_Arduino.ino.

3.10.2.26 servoYawCenter

```
#define servoYawCenter 80
```

Central angle in degrees

Definition at line 194 of file TDF02-145_Arduino.ino.

3.10.2.27 servoYawPin

```
#define servoYawPin 5
```

Servo for yaw movement (shaking of head - like when you indicate no) of head is connected to Arduino pin 5

Definition at line 175 of file TDF02-145_Arduino.ino.

3.10.2.28 Start

```
#define Start 1
```

An integer used in the decision flow for locomotion

Definition at line 103 of file TDF02-145_Arduino.ino.

3.10.2.29 Stop

```
#define Stop 0
```

An integer used in the decision flow for locomotion

Definition at line 102 of file TDF02-145_Arduino.ino.

3.10.3 Function Documentation

3.10.3.1 loop()

```
void loop ( )
```

Loops constantly.

Runs constantly. Nothing is put here because all instructions that Chotu receives are executed at once. There is no need for looping. In addition, when a new command is sent to Chotu, it is through the SerialEvent function, which works as an interrupt.

Loop can be used during diagnosis or tuning by printing out the locomotion or other statuses.

Definition at line 247 of file TDF02-145_Arduino.ino.

```
247             {
248     //printStatus();
249 }
```

3.10.3.2 setup()

```
void setup ( )
```

Runs only once when the robot starts up.

Sets up everything.

Sets up the PID algorithm. Attaches both VarSpeedServo objects to their respective servo motor signal pins. Writes initial angles to the servo motors. Begins serial communication for channles 0, 1, and 2.

Definition at line 227 of file TDF02-145_Arduino.ino.

```
227             {
228     PID_initial();
229
230     pitchservo.attach(servoPitchPin);
231     yawservo.attach(servoYawPin);
232
233     pitchservo.write(45, 100);
234     yawservo.write(45, 100);
235
236     Serial.begin(9600);
237     Serial1.begin(9600);
238     Serial2.begin(9600);
239 }
```

References PID_initial(), pitchservo, servoPitchPin, servoYawPin, and yawservo.

3.10.4 Variable Documentation

3.10.4.1 countLeft

```
double countLeft = 0
```

Integer to keep count of encoder signals from the left motor.

Definition at line 110 of file TDF02-145_Arduino.ino.

Referenced by encoderLeft(), encoderRight(), printStatus(), and stopMotion().

3.10.4.2 countRight

```
double countRight = 0
```

Integer to keep count of encoder signals from the right motor.

Definition at line 115 of file TDF02-145_Arduino.ino.

Referenced by encoderLeft(), encoderRight(), printStatus(), and stopMotion().

3.10.4.3 currentMotion

```
int currentMotion = 0
```

Integer to contain the present state of locomotion.

Definition at line 140 of file TDF02-145_Arduino.ino.

Referenced by moveDirection(), printStatus(), and stopMotion().

3.10.4.4 inputString

```
String inputString = ""
```

Carries command instruction string.

Starts out empty.

Definition at line 165 of file TDF02-145_Arduino.ino.

Referenced by decider(), serialEvent(), and serialEvent2().

3.10.4.5 leftPID

```
AutoPID leftPID = AutoPID(&countLeft, &countRight, &pwmLeft, robot_speed_min, robot_speed_max,
KP, KI, KD)
```

Constructor creates an instance of AutoPID named leftPID.

Applies PID algorithm to left motor.

Parameters

countLeft	Process Variable
countRight	Set Point
pwmLeft	Manipulated Value
robot_speed_min	The minimum value
robot_speed_max	The maximum value

Definition at line 150 of file TDF02-145_Arduino.ino.

Referenced by moveDirection(), and PID_initial().

3.10.4.6 pitchservo

```
VarSpeedServo pitchservo
```

Pitch servo motor defined as a VarSpeedServo object.

Definition at line 209 of file TDF02-145_Arduino.ino.

Referenced by headMotion_predefined(), movehead(), and setup().

3.10.4.7 pwmLeft

```
double pwmLeft
```

PWM output for left motor.

Definition at line 130 of file TDF02-145_Arduino.ino.

Referenced by moveDirection(), and printStatus().

3.10.4.8 pwmRight

```
double pwmRight
```

PWM output for right motor.

Definition at line 135 of file TDF02-145_Arduino.ino.

Referenced by moveDirection(), and printStatus().

3.10.4.9 rightPID

```
AutoPID rightPID = AutoPID(&countRight, &countLeft, &pwmRight, robot_speed_min, robot_speed_max,
KP, KI, KD)
```

Constructor creates an instance of AutoPID named rightPID.

Applies PID algorithm to right motor.

Parameters

countRight	Process Variable
countLeft	Set Point
pwmRight	Manipulated Value
robot_speed_min	The minimum value
robot_speed_max	The maximum value

Definition at line 160 of file TDF02-145_Arduino.ino.

Referenced by moveDirection(), and PID_initial().

3.10.4.10 stepsMotion

```
double stepsMotion = 0
```

Don't remember.

Definition at line 125 of file TDF02-145_Arduino.ino.

Referenced by decider(), moveDirection(), and printStatus().

3.10.4.11 stepsOverall

```
double stepsOverall = 0
```

Integer to keep an average count of encoder signals from the both motors.

Definition at line 120 of file TDF02-145_Arduino.ino.

Referenced by decider(), encoderLeft(), encoderRight(), moveDirection(), and printStatus().

3.10.4.12 yawservo

```
VarSpeedServo yawservo
```

Yaw servo motor defined as a VarSpeedServo object.

Definition at line 214 of file TDF02-145_Arduino.ino.

Referenced by headMotion_predefined(), movehead(), and setup().

Index

CCW
TDF02-145_Arduino.ino, 21
countLeft
TDF02-145_Arduino.ino, 28
countRight
TDF02-145_Arduino.ino, 28
currentMotion
TDF02-145_Arduino.ino, 29
CW
TDF02-145_Arduino.ino, 22

decider
decider.ino, 5
decider.ino, 5
decider, 5
moveBackward, 7
moveForward, 7
turnLeft, 8
turnRight, 8

encoder_L_C1
TDF02-145_Arduino.ino, 22
encoder_R_C1
TDF02-145_Arduino.ino, 22
encoderFunctions.ino, 9
encoderLeft, 9
encoderRight, 9
encoderLeft
encoderFunctions.ino, 9
encoderRight
encoderFunctions.ino, 9

Front
TDF02-145_Arduino.ino, 22

headMotion_predefined
headMovement.ino, 10
headMovement.ino, 10
headMotion_predefined, 10
movehead, 11

inputString
TDF02-145_Arduino.ino, 29

KD
TDF02-145_Arduino.ino, 22
KI
TDF02-145_Arduino.ino, 23
KP
TDF02-145_Arduino.ino, 23

Left
TDF02-145_Arduino.ino, 23
leftPID
TDF02-145_Arduino.ino, 29
loop
TDF02-145_Arduino.ino, 27

maxpitchlimit
TDF02-145_Arduino.ino, 23
maxyawlimit
TDF02-145_Arduino.ino, 23
minpitchlimit
TDF02-145_Arduino.ino, 24
minyawlimit
TDF02-145_Arduino.ino, 24
motionText
printStatus.ino, 16
motor
move_Direction.ino, 12
motor_ENA
TDF02-145_Arduino.ino, 24
motor_ENB
TDF02-145_Arduino.ino, 24
motor_IN1
TDF02-145_Arduino.ino, 24
motor_IN2
TDF02-145_Arduino.ino, 25
motor_IN3
TDF02-145_Arduino.ino, 25
motor_IN4
TDF02-145_Arduino.ino, 25
move_Direction.ino, 12
motor, 12
moveDirection, 13
moveBackward
decider.ino, 7
moveDirection
move_Direction.ino, 13
moveForward
decider.ino, 7
movehead
headMovement.ino, 11

PID_initial
PID_Initial.ino, 15
PID_Initial.ino, 15
PID_initial, 15
pitchservo
TDF02-145_Arduino.ino, 30
printStatus

printStatus.ino, 17
printStatus.ino, 16
motionText, 16
printStatus, 17
pwmLeft
TDF02-145_Arduino.ino, 30
pwmRight
TDF02-145_Arduino.ino, 30

README.md, 18
Rear
TDF02-145_Arduino.ino, 25
Right
TDF02-145_Arduino.ino, 25
rightPID
TDF02-145_Arduino.ino, 30
robot_speed_max
TDF02-145_Arduino.ino, 26
robot_speed_min
TDF02-145_Arduino.ino, 26

serialEvent
serialEvent.ino, 18
serialEvent.ino, 18
serialEvent, 18
serialEvent2, 18
serialEvent2
serialEvent.ino, 18
servoPitchCenter
TDF02-145_Arduino.ino, 26
servoPitchPin
TDF02-145_Arduino.ino, 26
servoYawCenter
TDF02-145_Arduino.ino, 26
servoYawPin
TDF02-145_Arduino.ino, 27
setup
TDF02-145_Arduino.ino, 28
Start
TDF02-145_Arduino.ino, 27
stepsMotion
TDF02-145_Arduino.ino, 31
stepsOverall
TDF02-145_Arduino.ino, 31
Stop
TDF02-145_Arduino.ino, 27
stopMotion
stopMotion.ino, 19
stopMotion.ino, 19
stopMotion, 19

TDF02-145_Arduino.ino, 20
CCW, 21
countLeft, 28
countRight, 28
currentMotion, 29
CW, 22
encoder_L_C1, 22
encoder_R_C1, 22
Front, 22
inputString, 29
KD, 22
KI, 23
KP, 23
Left, 23
leftPID, 29
loop, 27
maxpitchlimit, 23
maxyawlimit, 23
minpitchlimit, 24
minyawlimit, 24
motor_ENA, 24
motor_ENB, 24
motor_IN1, 24
motor_IN2, 25
motor_IN3, 25
motor_IN4, 25
pitchservo, 30
pwmLeft, 30
pwmRight, 30
Rear, 25
Right, 25
rightPID, 30
robot_speed_max, 26
robot_speed_min, 26
servoPitchCenter, 26
servoPitchPin, 26
servoYawCenter, 26
servoYawPin, 27
setup, 28
Start, 27
stepsMotion, 31
stepsOverall, 31
Stop, 27
yawservo, 31
turnLeft
decider.ino, 8
turnRight
decider.ino, 8

yawservo
TDF02-145_Arduino.ino, 31