Putting it all Together Land


```
#include "src/BNO/BNO.h"
#include "src/GPS/gps.h"
#include "src/HeadingControl/HeadingControl.h"
#include "src/MotorControl/MotorControl.h"
#include "src/CircularArray/CircularArray.h"
#include <SD.h>

// Objects we will use                          ENA, ENB, IN2, IN1, IN3, IN4,
HBridgeControl myHBridge        = HBridgeControl(7,   2,  47,  49,  51,  53); // Motor Control
HeadingControl myHeadingControl = HeadingControl(); // Heading Controller
CircularArray  myArray          = CircularArray();  // Array to keep track of drift.
GPS            myGPS            = GPS();            // Our GPS Sensor
BNO            myBNO            = BNO();            // Our Acc., Gyro., Mag., sensor.


// Data Structures we will use
GPSData        myGPSData;
GPSPair        myGPSPair;

// Important Variables
double distanceBetween;       // Distance between current lat/lon and destination lat/lon
double targetHeading;         // The heading we need to travel at to reach destination
double currentHeading;        // The heading we are traveling at
double headingError;          // Difference in target and current heading
double gpsError;              // difference in myGPSData.Heading and currentHeading
double averageGPSError = 0;   // Average difference in myGPSData.Heading and currentHeading
int    bnoResetPin     = 6;   // Pin to reset the BNO chip
int    SDChipSelect    = 10;

// Acceptable Heading Error - 2%
const double ACC_HEAD_ERR = 0.02;

//// Path
long dLons[] =
{
  -819913429,
  -819916915,
  -819910571,
  -819909347,
  -819909936
};

long dLats[] =
{
  334676789,
  334674976,
  334672878,
  334675469,
  334677542
};

int waypointIndex   = 0;
int numWayPoints    = 5;
bool newWayPoint    = 1;
bool updatedAverage = 0;
double totalDistance;
long startTime;
bool GotFirstStartTime = 0;

```

These are all the variables and objects the Program will work with.

Putting it all Together


```
60 void setup()
61 {
62   Serial.begin(9600);
63   Serial.println("#GetObsessed!");
64   pinMode(LED_BUILTIN, OUTPUT);
65
66   // setupGPS()
67   // Connect the GPS to Serial1.
68   // This is RX0 and TX0 on the giga board.
69   Serial1.begin(38400);
70   if(!myGPS.setupGPS(Serial1))
71   {
72     // Freeze if setup fails
73     digitalWrite(LED_BUILTIN, HIGH);
74     Serial.println("GPS Failed to begin.");
75     while(1);
76   }
77
78   // setupBNO()
79   // Connect the BNO to Serial2
80   Serial2.begin(115200);
81   if(!myBNO.setupBNO(&Serial2, bnoResetPin))
82   {
83     // Freeze if BNO failed
84     digitalWrite(LED_BUILTIN, HIGH);
85     Serial.println("BNO Failed to start");
86     while(1);
87   }
88
89   //setupHBridge()
90   // Turn the motors off.
91   myHBridge.SetSpeed(0, 0);
92   // Set both the motors to forward
93   myHBridge.SetDirection(FORWARD, FORWARD);
94
95   // SetupSD()
96   // Set the SPI ChipSelect pin as an output
97   pinMode(SDChipSelect, OUTPUT);
98   if(!SD.begin(SDChipSelect))
99   {
100    // If it fails to begin, freeze.
101    // Check that the card is properly inserted.
102    digitalWrite(LED_BUILTIN, HIGH);
103    Serial.println("SD Card failed to begin.");
104    while(1);
105  }
106  startTime = millis();
107 }
```

These are all the Setup commands we want to do to ensure everything is functioning properly.

```
109 void loop()
110 {
111   // Set the "destination" to point 2 in the GPS pair.
112   myGPSPair.lat2_L = dLats[waypointIndex];
113   myGPSPair.lon2_L = dLons[waypointIndex];
114
115   // Update Our GPS data
116   myGPS.GetData(myGPSData);
117
118   // Set the "current" position to point 1 in the GPS pair.
119   myGPSPair.lat1_L = myGPSData.lattitude;
120   myGPSPair.lon1_L = myGPSData.longitude;
121
122   // Get the information we need to control the robot
123   distanceBetween = myHeadingControl.GetDistanceBetween(myGPSPair);
124   targetHeading   = myHeadingControl.GetTargetHeading(myGPSPair);
125   currentHeading  = myBNO.GetHeading() - averageGPSError;
126   headingError    = myHeadingControl.GetHeadingError(currentHeading, targetHeading);
127   gpsError        = currentHeading - (myGPSData.heading / 100000.0);  // could flip
128
129   // Log the data we are about to use to make decisions
130   logToSd();
131   printVars();
133   // If there are more than 4 sats. in view.
134   if (myGPSData.SIV > 4)
135   {
136
137     // Start the Timer for the first time
138     if(GotFirstStartTime == 0)
139     {
140       startTime          = millis();
141       GotFirstStartTime = 1;
142     }
143
144     // Keep track of the error
145     myArray.Update(gpsError);
146
147     // If we haven't been to this waypoint before.
148 //    if (newWayPoint)
149 //    {
150 //      // Get the distance we are away from it
151 //      totalDistance = distanceBetween;
152 //      // Tell the program we have seen this point before.
153 //      newWayPoint = 0;
154 //    }
155
156     // If our heading is too far off from the acceptable error
157     if (abs(headingError) > ACC_HEAD_ERR)
158     {
159       myHBridge.SetSpeed(0, 0); // Stop to think
160       RotateInPlaceLand();      // Fix Heading
161     }
163     // If we are within 3.0 m of our destination point.
164     if(distanceBetween <= 3.0)
165     {
166       // Turn Motors Off and go to the next waypoint
167       myHBridge.SetSpeed(0, 0);
168       waypointIndex++;
169
170       // Tell the program we are going to a new way point
171       // So we need to get a new total distance
172       newWayPoint = 1;
173
174       // Get the average for this leg and use it for the next one. Reset the average after.
175       averageGPSError = myArray.GetAverage() + averageGPSError;
176       myArray.ResetAverage();
177       startTime = millis();
```

page 34 book 2 for update

The heart of the algorithm...

Putting it all Together

Land

Dec 2023
Wesley Corbett

```
178
179       // Tell the program we haven't updated the average for this leg yet.
180       // updatedAverage = 0;
181
182       if (waypointIndex >= numWayPoints)
183       {
184         // Done with path. Freeze.
185         while(1);
186       }
187     }
188 //    else if ((distanceBetween <= totalDistance/2 and updatedAverage==0) and newWayPoint == 0)
189 //    { // If we are halfway to a new point and we haven't updated our average yet
190 //      // Then update the average
191 //      averageGPSError = myArray.GetAverage() + averageGPSError;
192 //      // Reset the average for the error
193 //      myArray.ResetAverage();
194 //      // Tell the program we have updated the average for this leg
195 //      updatedAverage  = 1;
196 //    }
197     else if ((millis() - startTime) > 10000) // If we haven't updated GPS average for 10 seconds.
198     {
199       averageGPSError = myArray.GetAverage() + averageGPSError;
200       myArray.ResetAverage();
201       startTime = millis();
202     }
203     else
204     {
205       // Our heading is not too far off and we are not close enough to the point
206       // Go Forwards
207       myHBridge.SetSpeed(255, 255);
208     }
209   }
210   else
211   {
212     // 4 Sats are not in view. Standby...
213     myHBridge.SetSpeed(0, 0);
214   }
215 }
```

```
248 void RotateInPlaceLand()
249 {
250   // While our heading is too far off from the target
251   while(abs(headingError) > ACC_HEAD_ERR)
252   {
253     if (headingError > 0)
254     {
255       // Rotate Right
256       // Set the left motor Forward and the right motor to reverse.
257       myHBridge.SetDirection(FORWARD, REVERSE);
258       myHBridge.SetSpeed(255, 255);
259     }
260     else
261     {
262       // Rotate Left
263       // Set the left motor Reverse and the right motor Forward.
264       myHBridge.SetDirection(REVERSE, FORWARD);
265       myHBridge.SetSpeed(255, 255);
266     }
267
268     currentHeading  = myBNO.GetHeading() - averageGPSError;
269     headingError    = myHeadingControl.GetHeadingError(currentHeading, targetHeading);
270   }
271   // Set the motors back to straight
272   myHBridge.SetDirection(FORWARD, FORWARD);
273 }
```

* → Page 39 book 2.

Putting it All Together water

```
#include "src/BNO/BNO.h"
#include "src/GPS/gps.h"
#include "src/HeadingControl/HeadingControl.h"
#include "src/MotorControl/MotorControl.h"
#include "src/CircularArray/CircularArray.h"
#include "src/Controller/Controller.h"
#include <SD.h>

// Objects we will use
ESCControl      myMotors          = ESCControl()    // Motor Control
HeadingControl myHeadingControl = HeadingControl(); // Heading Controller
CircularArray  myArray           = CircularArray();  // Array to keep track of drift.
GPS             myGPS             = GPS();            // Our GPS Sensor
BNO             myBNO             = BNO();            // Our Acc., Gyro., Mag., sensor.
Controller      myController      = Controller();


// Data Structures we will use
GPSData         myGPSData;
GPSPair         myGPSPair;

// Important Variables
double distanceBetween;       // Distance between current lat/lon and destination lat/lon
double targetHeading;         // The heading we need to travel at to reach destination
double currentHeading;        // The heading we are traveling at
double headingError;          // Difference in target and current heading
double gpsError;              // difference in myGPSData.Heading and currentHeading
double averageGPSError = 0;   // Average difference in myGPSData.Heading and currentHeading
int    bnoResetPin     = 6;   // Pin to reset the BNO chip
int    SDChipSelect    = 10;
int    leftServoPin    = 2;
int    rightServoPin   = 7;
int    currentLeftSpeed  = 1500;
int    currentRightSpeed = 1500;

// Acceptable Heading Error - 2%
const double ACC_HEAD_ERR = 0.02;
//

```

new

// Omitting Path (dLats[], dLons[])

```
int waypointIndex   = 0;
int numWayPoints    = 99;
bool newWayPoint    = 1;
bool updatedAverage = 0;
double totalDistance;
long startTime;
bool GotFirstStartTime = 0;
```

All the objects and variables the program will work with.

Putting it All Together

Dec 2023

```
227 void setup()
228 {
229   //Serial.begin(9600);
230 //  Serial.println("$GetObsessed!");
231   pinMode(LED_BUILTIN, OUTPUT);
232
233   // setupGPS()
234   // Connect the GPS to Serial1.
235   // This is RX0 and TX0 on the giga board.
236   Serial1.begin(38400);
237   if(!myGPS.setupGPS(Serial1))
238   {
239     // Freeze if setup fails
240     digitalWrite(LED_BUILTIN, HIGH);
241     //Serial.println("GPS Failed to begin.");
242     while(1);
243   }
244
245   // setupBNO()
246   // Connect the BNO to Serial2
247   Serial2.begin(115200);
248   if(!myBNO.setupBNO(&Serial2, bnoResetPin))
249   {
250     // Freeze if BNO failed
251     digitalWrite(LED_BUILTIN, HIGH);
252     //Serial.println("BNO Failed to start");
253     while(1);
254   }
255
256   // setUPServos()
257   // Set up the motors and turn them off
258   myMotors.setup(leftServoPin, rightServoPin);
259
260   // SetupSD()
261   // Set the SPI ChipSelect pin as an output
262   pinMode(SDChipSelect, OUTPUT);
263   if(!SD.begin(SDChipSelect))
264   {
265    // If it fails to begin, freeze.
266    // Check that the card is properly inserted.
267    digitalWrite(LED_BUILTIN, HIGH);
268    //Serial.println("SD Card failed to begin.");
269    while(1);
270   }
271
272   File outputfile = SD.open("GPSData.txt", FILE_WRITE);
273   outputfile.println("==================");
274   outputfile.close();
275
276   myController.begin();
277   startTime = millis();
278 }
```

```
280 void loop()
281 {
282   myController.ReadController();
283   //Serial.println(myController.ProgramMode);
284
285    File outputfile = SD.open("GPSData.txt", FILE_WRITE);
286   outputfile.println("Inside main loop");
287   outputfile.close();
288   if (myController.ProgramMode == 0)
289   {
290     // ============================
291      outputfile = SD.open("GPSData.txt", FILE_WRITE);
292     outputfile.println("Inside auto Loop");
293     outputfile.close();
294     // ============================
295
296     // Set the "destination" to point 2 in the GPS pair.
297     myGPSPair.lat2_L = dLats[waypointIndex];
298     myGPSPair.lon2_L = dLons[waypointIndex];
299
300     // Update Our GPS data
301     myGPS.GetData(myGPSData);
302
303     // Set the "current" position to point 1 in the GPS pair.
304     myGPSPair.lat1_L = myGPSData.lattitude;
305     myGPSPair.lon1_L = myGPSData.longitude;
307     // Get the information we need to control the robot
308     distanceBetween = myHeadingControl.GetDistanceBetween(myGPSPair);
309     targetHeading   = myHeadingControl.GetTargetHeading(myGPSPair);
310     currentHeading  = myBNO.GetHeading() - averageGPSError;             // would need to flip here too.
311     headingError    = myHeadingControl.GetHeadingError(currentHeading, targetHeading);
312     gpsError        = currentHeading - (myGPSData.heading / 100000.0);  // could flip
313
314     // ============================
315      outputfile = SD.open("GPSData.txt", FILE_WRITE);
316     outputfile.println("Before Log");
317     outputfile.close();
318     // ============================
319
320     // Log the data we are about to use to make decisions
321     logToSd();
322     //printVars();
323
324     // If there are more than 4 sats. in view.
325     if (myGPSData.SIV > 4)
326     {
327
328     // ============================
329      outputfile = SD.open("GPSData.txt", FILE_WRITE);
330     outputfile.println("Inside SIV");
331     outputfile.close();
332     // ============================
333
334       // Start the Timer for the first time
335       if(GotFirstStartTime == 0)
336       {
337         startTime          = millis();
338         GotFirstStartTime = 1;
339       }
340
341       // ============================
342       outputfile = SD.open("GPSData.txt", FILE_WRITE);
343       outputfile.println("Before Update the Array");
344       outputfile.close();
345       // ============================
346
347       // Keep track of the error
348       myArray.Update(gpsError);
349
350       outputfile = SD.open("GPSData.txt", FILE_WRITE);
351       outputfile.println("After Update the Array");
352       outputfile.close();
```

Putting it All Together

Dec 2023
Wesley Castle

```
353
354       // If we haven't been to this waypoint before.
355       //    if (newWayPoint)
356       //    {
357       //      // Get the distance we are away from it
358       //      totalDistance = distanceBetween;
359       //      // Tell the program we have seen this point before.
360       //      newWayPoint = 0;
361       //    }
362
363       // If our heading is too far off from the acceptable error
364       if (abs(headingError) > ACC_HEAD_ERR)
365       {
366         updateMotors(1500, 1500); // Stop to think
367         RotateInPlaceBoat();      // Fix Heading
368       }
369
370       // If we are within 3.0 m of our destination point.
371       if(distanceBetween <= 2.0)
372       {
373         // Turn Motors Off and go to the next waypoint
374         updateMotors(1500, 1500);
375         waypointIndex++;
376
377         // Tell the program we are going to a new way point
378         // So we need to get a new total distance
379         newWayPoint = 1;
380
381         // Get the average for this leg and use it for the next one. Reset the average after.
382         averageGPSError = myArray.GetAverage() + averageGPSError;
383         myArray.ResetAverage();
384         startTime = millis();
385
386         // Tell the program we haven't updated the average for this leg yet.
387         //      updatedAverage = 0;
388
389         if (waypointIndex >= numWayPoints)
390         {
391           // Done with path.
392           newWayPoint = 1;
393           waypointIndex = 0;
394           myController.ProgramMode = 1;
395           myArray.ResetAverage();
396           GotFirstStartTime = 0;
397           averageGPSError = 0;
398         }
399       }
400         //    else if ((distanceBetween <= totalDistance/2 and updatedAverage==0) and newWayPoint == 0)
401         //    { // If we are halfway to a new point and we haven't updated our average yet
402         //      // Then update the average
403         //      averageGPSError = myArray.GetAverage() + averageGPSError;
404         //      // Reset the average for the error
405         //      myArray.ResetAverage();
406         //      // Tell the program we have updated the average for this leg
407         //      updatedAverage  = 1;
408         //    }
409       else if ((millis() - startTime) > 10000) // If we haven't updated GPS average for 10 seconds.
410       {
411         averageGPSError = myArray.GetAverage() + averageGPSError;
412         myArray.ResetAverage();
413         startTime = millis();
414       }
415       else
416       {
417         // Our heading is not too far off and we are not close enough to the point
418         // Go Forwards
419       // ============================
420        outputfile = SD.open("GPSData.txt", FILE_WRITE);
421       outputfile.println("Set Motors To Forward");
422       outputfile.close();
423       // ============================
424         updateMotors(1650, 1650);
425       }
426     }
```

```
    else
    {
      // 4 Sats are not in view. Standby...
      // ============================
       outputfile = SD.open("GPSData.txt", FILE_WRITE);
      outputfile.println("4 Not In View");
      outputfile.close();
    // ============================
      updateMotors(1500, 1500);
    }
  }
  else // Remote Control Mode
  {
    //Serial.print("LeftSpeed: ");
    //Serial.print(myController.leftMotorSpeedTarget);
    //Serial.print(" RightSpeed: ");
    //Serial.println(myController.rightMotorSpeedTarget);

    updateMotors(
      myController.leftMotorSpeedTarget,
      myController.rightMotorSpeedTarget);
  }
}
```

```
void RotateInPlaceBoat()
{
  // While our heading is too far off from the target
  while(abs(headingError) > ACC_HEAD_ERR)
  {
    // check the controller for updates
    myController.ReadController();
    if (myController.ProgramMode == 1)
    {
      // if we should be in remote control mode, leave this method.
      return;
    }

    if (headingError > 0)
    {
      // Rotate Right
      // Set the left motor Forward and the right motor to reverse.
      updateMotors(1700, 1280);
    }
    else
    {
      // Rotate Left
      // Set the left motor Reverse and the right motor Forward.
      updateMotors(1280, 1700);
    }

    currentHeading  = myBNO.GetHeading() - averageGPSError;
    headingError    = myHeadingControl.GetHeadingError(currentHeading, targetHeading);
  }
  // Set the motors back to straight
  updateMotors(1500, 1500);
}
```

Putting it all Together


```
541 void updateMotors(int targetLeft, int targetRight)
542 {
543     while (currentLeftSpeed != targetLeft || currentRightSpeed != targetRight)
544     {
545         if (currentLeftSpeed != targetLeft)
546         {
547             if(currentLeftSpeed < targetLeft)
548             {
549                 currentLeftSpeed++;
550             }
551             else
552             {
553                 currentLeftSpeed--;
554             }
555         }
556
557         if (currentRightSpeed != targetRight)
558         {
559             if(currentRightSpeed < targetRight)
560             {
561                 currentRightSpeed++;
562             }
563             else
564             {
565                 currentRightSpeed--;
566             }
567         }
568         myMotors.SetSpeed(currentLeftSpeed, currentRightSpeed);
569     }
570 }
```

This method ensures that the motor speed doesn't change by more than 1 at a time. This helps reduce jutters.

Water

These methods are used in both programs but only listed once for space reasons.

```
278 void printVars()
276 {
277   Serial.print(myGPSData.lattitude);
278   Serial.print(", ");
279   Serial.print(myGPSData.longitude);
280   Serial.print(", ");
281   Serial.print(myGPSData.SIV);
282   Serial.print(", ");
283   Serial.print(myGPSData.heading);
284   Serial.print(", ");
285   Serial.print(myGPSData.speed);
286   Serial.print(", ");
287   Serial.print(distanceBetween);
288   Serial.print(", ");
289   Serial.print(targetHeading);
290   Serial.print(", ");
291   Serial.print(currentHeading);
292   Serial.print(", ");
293   Serial.print(headingError);
294   Serial.print(", ");
295   Serial.print(gpsError);
296   Serial.print(", ");
297   Serial.print(averageGPSError);
298   Serial.println();
299 }
```

```
217 void logToSd()
218 {
219   File outputfile = SD.open("GPSData.txt", FILE_WRITE);
220
221   outputfile.print(myGPSData.lattitude);
222   outputfile.print(", ");
223   outputfile.print(myGPSData.longitude);
224   outputfile.print(", ");
225   outputfile.print(myGPSData.SIV);
226   outputfile.print(", ");
227   outputfile.print(myGPSData.heading);
228   outputfile.print(", ");
229   outputfile.print(myGPSData.speed);
230   outputfile.print(", ");
231   outputfile.print(distanceBetween);
232   outputfile.print(", ");
233   outputfile.print(targetHeading);
234   outputfile.print(", ");
235   outputfile.print(currentHeading);
236   outputfile.print(", ");
237   outputfile.print(headingError);
238   outputfile.print(", ");
239   outputfile.print(gpsError);
240   outputfile.print(", ");
241   outputfile.print(averageGPSError);
242   outputfile.println();
243
244   // Close the file
245   outputfile.close();
246 }
```