This doucment outlines the implementation of User Clibration of the magnetic compass. Specifically, it describes the mathematical computations in C/C++ source code.

This calibration will correct or compensate for the hard iron effects. Hard iron effects are magnetic field superimpositions on the earth's magnetic field that are fixed in magnitude and which do not depend on the orientation of the compass.

The compass calibration shall be initiated by a user command that put the compass into the calibrtion mode. Once in that mode the compass shall be rotated through 360 degrees abouth the Forward or Left direction followed by a 360 rotation about the Up-Down direction. That is, there are two full rotations required to complete the physical movement required. At the end of the rotations a command to end the calibration process shall be issed by the user to the compass.

It is important to keep the Forward (or Left) axis level diring the first rotation and to keep the Up-Down axis vertical during the secon calibration and until the ful calibrtion is complete. Failure to follow these process will result in less than optimal calibration and heading errors are likely.

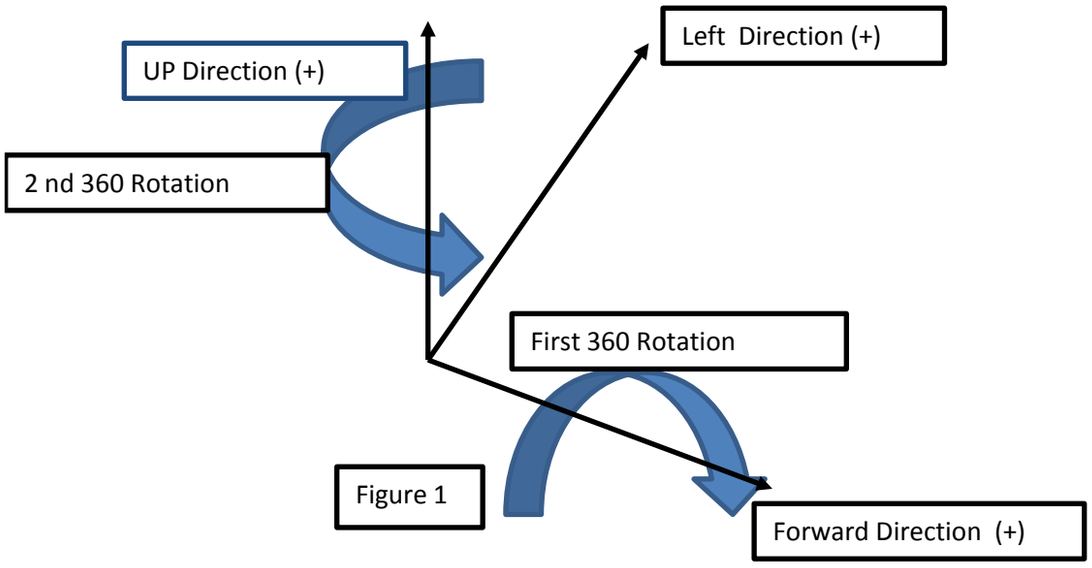

Figure 1

```
#define CalThreshold 0
int Xmax, Xmin, Ymax, Ymin, Zmax, Zmin;
void Initialize_Cal_Variables(int MagX, int MagY, int  MagZ)
void Calibrate(int MagX, int MagY, int  MagZ)
void Compute_and_Save(void)
void Hard_Iron_Correction( int Xoff, int Yoff, int Zoff )
void Initialize_Cal_Variables(int MagX, int MagY, int  MagZ)
{
     // set Max and Min values of the mag output to the current values
     Xmax=MagX;
     Xmin=MagX;
     Ymax=MagY;
     Ymin=MagY;
     Zmax=MagZ;
     Zmin=MagZ;


}

void Calibrate(int MagX, int MagY, int  MagZ)
{

//  this routine will capture the max and min values of the mag X, Y, and Z data while the
//  compass is being rotated 360 degrees through the level plane and the upright plane.
//  i.e. horizontal and vertical circles.
// This function should be invoked while making continuous measurements
//on the magnetometers
int  MagXreading,  MagYreading,  MagZreading;

     MagXreading=MagX;  // just for clarification...  can remove these lines
     MagYreading=MagY;
     MagZreading=MagZ;



     if (MagXreading > Xmax )   Xmax = MagXreading;
     if (MagXreading < Xmin )   Xmin = MagXreading;

     if(MagYreading > Ymax ) Ymax = MagYreading;
     if(MagYreading < Ymin ) Ymin = MagYreading;


     if(MagZreading > Zmax ) Zmax = MagZreading;
     if(MagZreading < Zmin ) Zmin = MagZreading;


}
```

```
void Compute_and_Save(void)
{

    if(abs(Xmax - Xmin) > CalThreshold )
    {

            Mag_UserCal_Offset_X = (Xmax + Xmin)/2;
    //      Save parameters in EE

    }

    if(abs(Ymax - Ymin) > CalThreshold )
    {

            Mag_UserCal_Offset_Y=  (Ymax + Ymin)/2;

            //Save parameters in EE

    }


    if(abs(Zmax - Zmin) > CalThreshold )
    {

            Mag_UserCal_Offset_Z =  (Zmax +Zmin)/2;

    //      Save parameters in EE
    }

}




void Hard_Iron_Correction( int Xoff, int Yoff, int Zoff )   // call this function for correction

{

    MagX -= Mag_UserCal_Offset_X;
    MagY -= Mag_UserCal_Offset_Y;
    MagZ -= Mag_UserCal_Offset_Z;

}
```