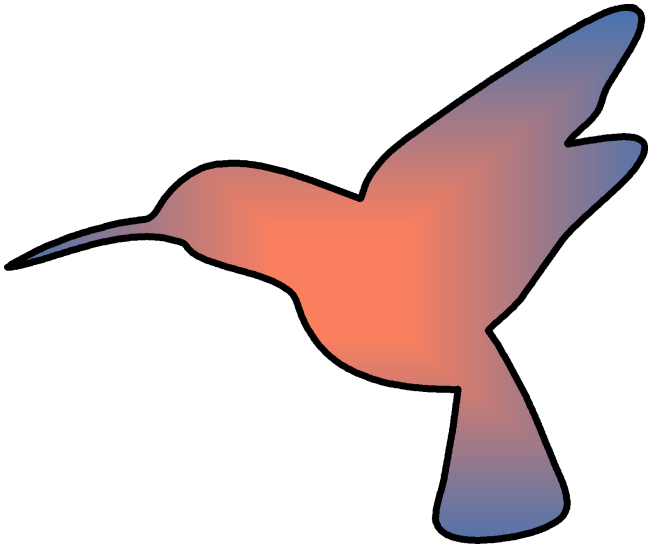
User Manual

Colibri

Inertial Motion Tracker



(Subject to technical modifications)

Augmented Vision Group of the

German Research Center for Artificial Intelligence GmbH

(http://www.dfki.de)

TRIVISIO

(http://www.trivisio.com)

Contents

1 Introduction **3**

2 Installation **4**
2.1 Windows . . . 4
2.1.1 Step 1 - Welcome Screen . . . 4
2.1.2 Step 2 - EULA . . . 5
2.1.3 Step 3 - Installation path . . . 5
2.1.4 Step 4 - Start menu folder . . . 6
2.1.5 Step 5 - Component installation . . . 6
2.2 Fedora 11 . . . 7

3 GUI **8**
3.1 Start the application on Windows operating systems . . . 8
3.2 Start the application on Linux . . . 8
3.3 The GUI . . . 9
3.4 Calibrating the magnetometers . . . 11
3.5 Boresighting . . . 12
3.6 Additional functions . . . 13
3.6.1 Disable drawing of textures . . . 13
3.6.2 Jitter Reduction . . . 13
3.6.3 Additional COM ports to scan . . . 14

4 SDK **15**
4.1 The test.c example . . . 15
4.2 The opengl example . . . 17
4.3 Simple Import mechanism for data profiling . . . 18

1 Introduction

Thank you for purchasing the Trivisio Colibri and using the SDK.

This document contains the user manual for the Colibri Tracker. The first chapter includes the installation guide. Within the second chapter the GUI with the including functionalities will be described. Chapter three represents the documentation of the SDK. The two basic applications are shown, whereas the `testc` example will be described.

2 Installation

This chapter provides the two installation guides for Microsoft Windows and Fedora 11.

2.1 Windows

If you don't have an installer yet, please download the installation file from the Trivisio GmbH website:

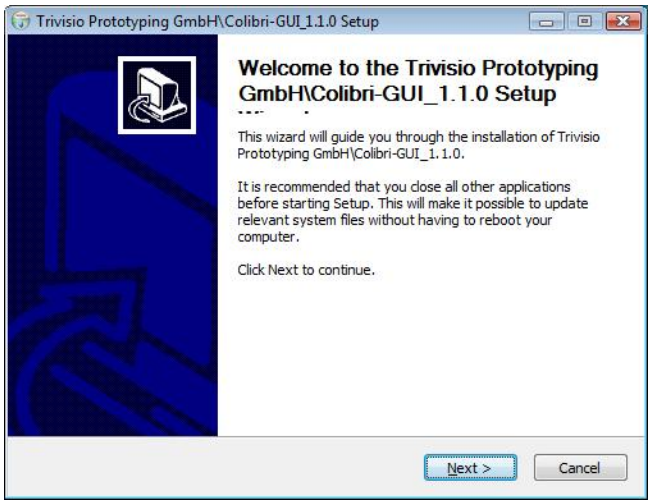
Go to the website http://www.trivisio.com/

Under *Support — Software* you will find the installer for the Colibri tracker.

After downloading the installer to a directory of your choice, execute the file. The following images will guide you through the installation.

2.1.1 Step 1 - Welcome Screen

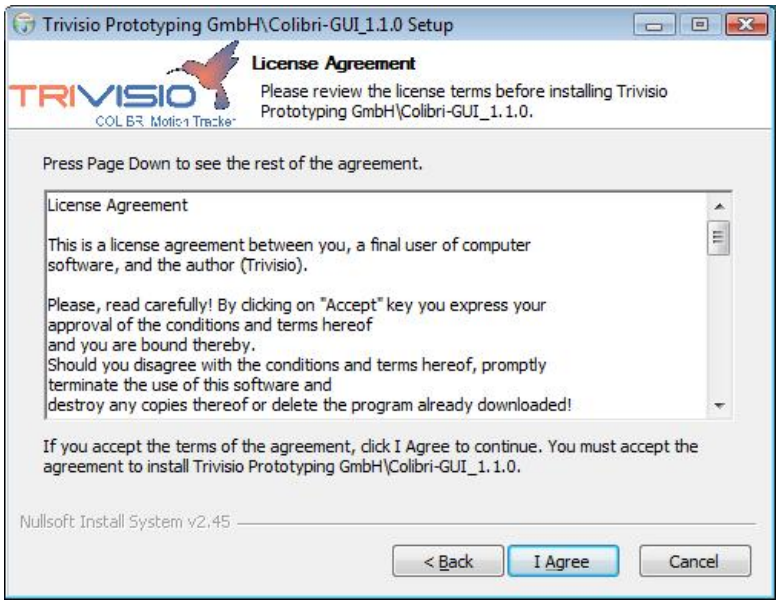
This is the welcome screen of the installer. Press next to start the installation or cancel to abort.



2.1.2 Step 2 - EULA

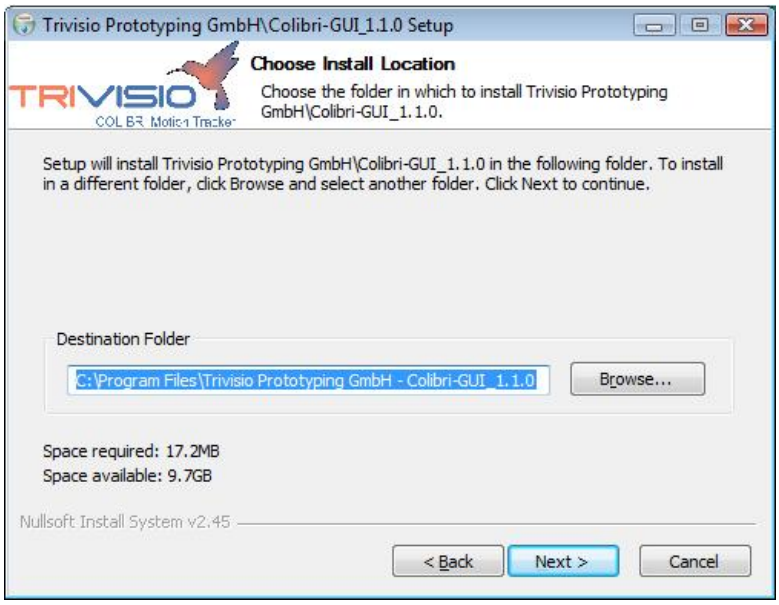
The next dialog is the End User License Agreement. Please read it carefully.

If you agree to the terms and conditions, please press the “I Agree” button to continue.



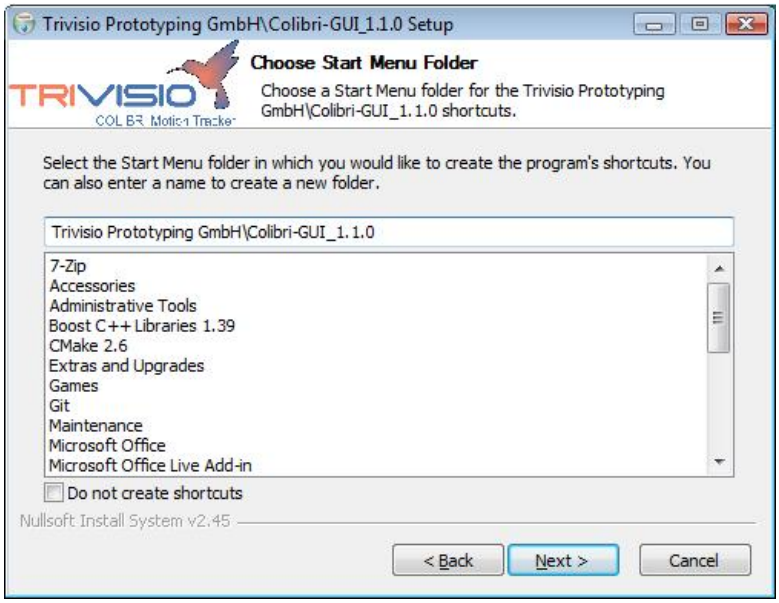
2.1.3 Step 3 - Installation path

In this dialog the user is able to choose or select the desired installation path.



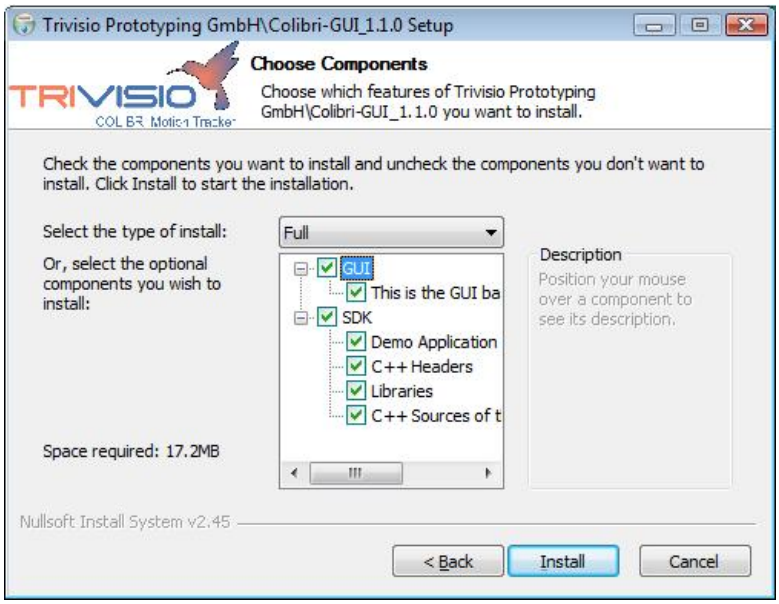
2.1.4 Step 4 - Start menu folder

If you want to have a different name for your start menu folder, please modify the path to your personal need.



2.1.5 Step 5 - Component installation

The next dialog can be used to select different installation types. The default installation type is **FULL**.



The different options are

- Full
- Developer
- GUI

- Custom

whereas, a **Full** installation includes the **Developer** and **GUI** installation.

After this dialog the installation is executed and you are able to use the applications and develop your own software based on the SDK.

2.2 Fedora 11

If you don't have an installer yet, please download the installation file from the Trivisio GmbH website:

Go to the website http://www.trivisio.com/

Under *Support* — *Software* you will find the installer for the Colibri tracker.

After successfully downloading the software:

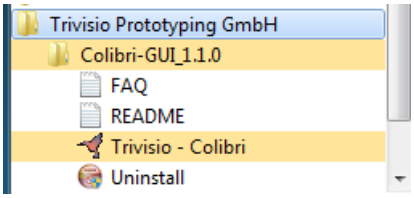
1. Execute the `FILENAME.sh` file.
2. Accept the license.
3. Follow the instructions for the target path installation.

3 GUI

This section provides an overview about the graphical user interface, which can be used to view the connected sensors and their values.

3.1 Start the application on Windows operating systems

You can start the application either by clicking the Trivisio - Colibri icon on your desktop or explore your start menu folder for your installation folder, default is: Trivisio Prototyping GmbH / Colibri_GUI_X.X.X (see image) and execute the Trivisio - Colibri.

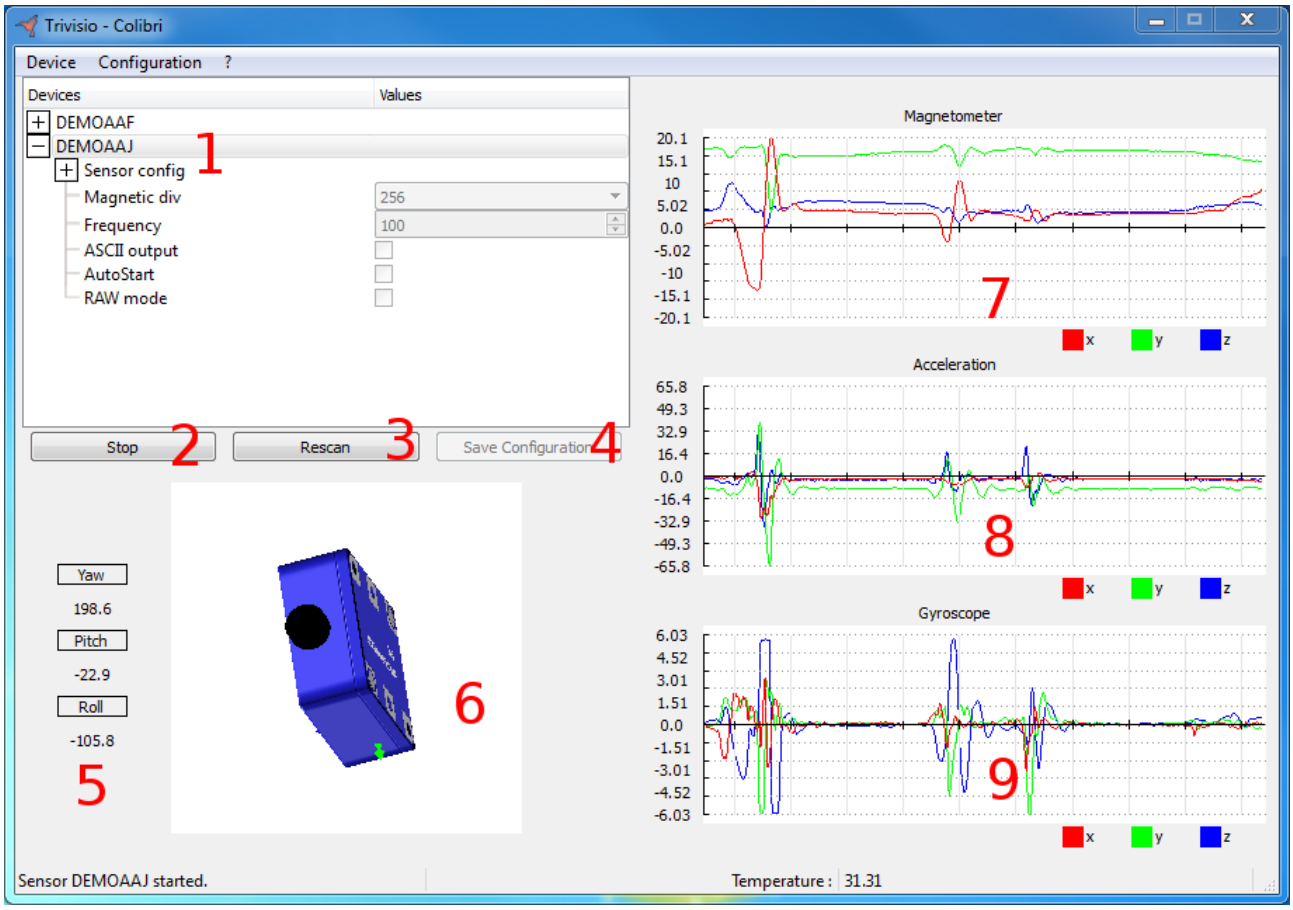


3.2 Start the application on Linux

1. Open a shell.
2. Go to your installation directory.
3. Execute the application by calling `bin/TrivisioGUI`.

3.3 The GUI

This is a snapshot of the gui application.

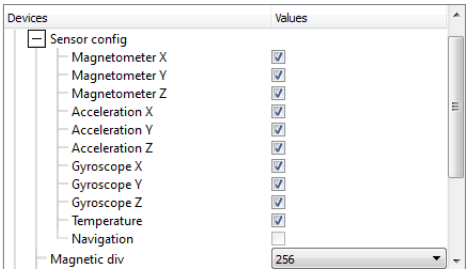


Based on this image, the functionalities will be described briefly in the following enumeration:

1. shows the list of the found sensors. The information of each sensor can be seen and changed by expanding the tree within the view by pressing the "plus" symbol.

   If the sensor is not running, the parameters can be changed, otherwise not. The user can enable the magnetometers, the accelerometers and the gyroscopes in "Sensor Config" (see image on the bottom). In addition, the Magnetic deviation, the frequency, the ASCII output, the autostart and the RAW mode can be enabled or disabled. Pay attention: the graphical representation is only suitable for disabled ASCII and disabled RAW mode. The modes and the according enabled widgets are shown in the following table.

RAW	ASCII	Enabled displays
off	off	Sensor and graphs
on	off	Only graphs
off	on	No output
on	on	No output



2. is the start and stop button for the sensors. The desired sensor must be selected and can be started or stopped by this button.

3. is the rescan button. When you connect a sensor during run-time, you can rescan for new sensors by hitting this button.

4. can be pressed, if you changed the configuration of the sensor and want to save it to the sensor.

5. are the computed roll, pitch and yaw values of the sensor.

6. is the OpenGL®, representation of the sensor orientation. The OpenGL®, textures can be enabled or disabled by the menubar item “Configuration”.

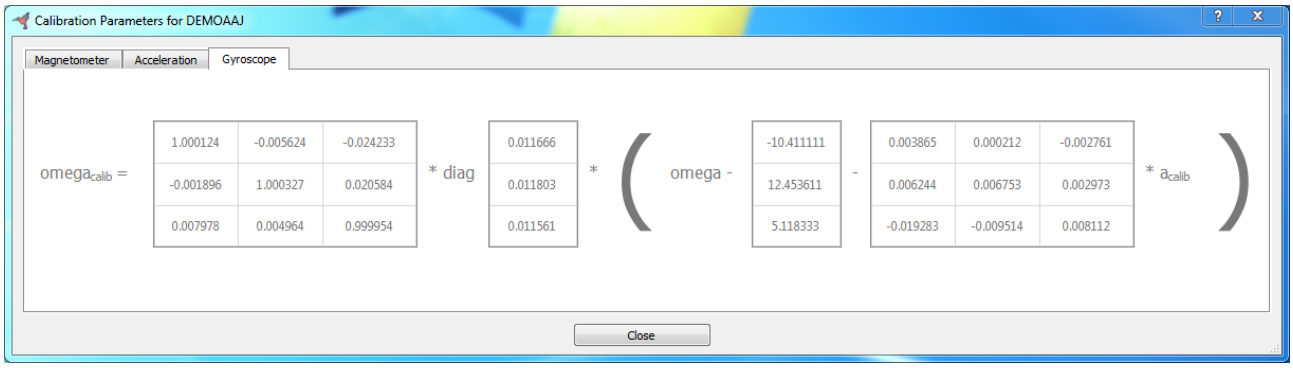
7. is the graphical view of the magnetometer sensor values.

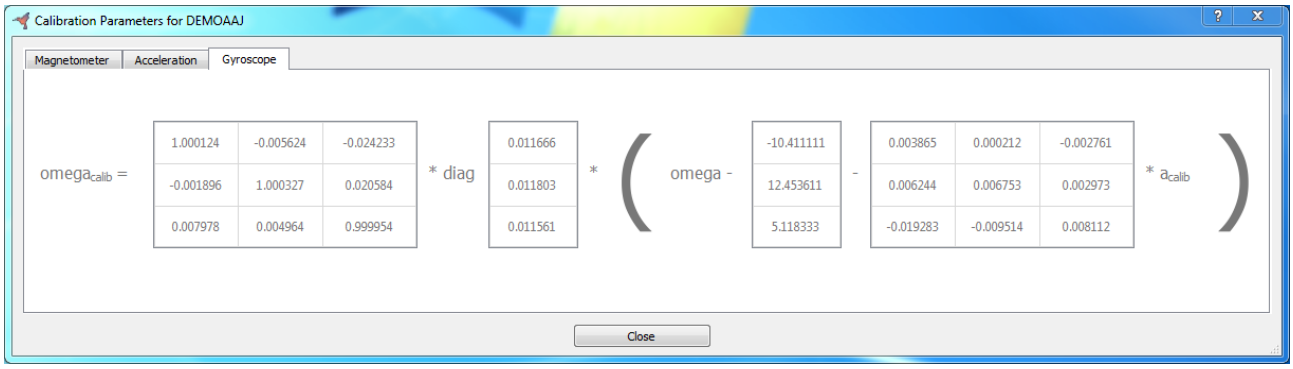
8. is the graphical view of the accelerometer sensor values.

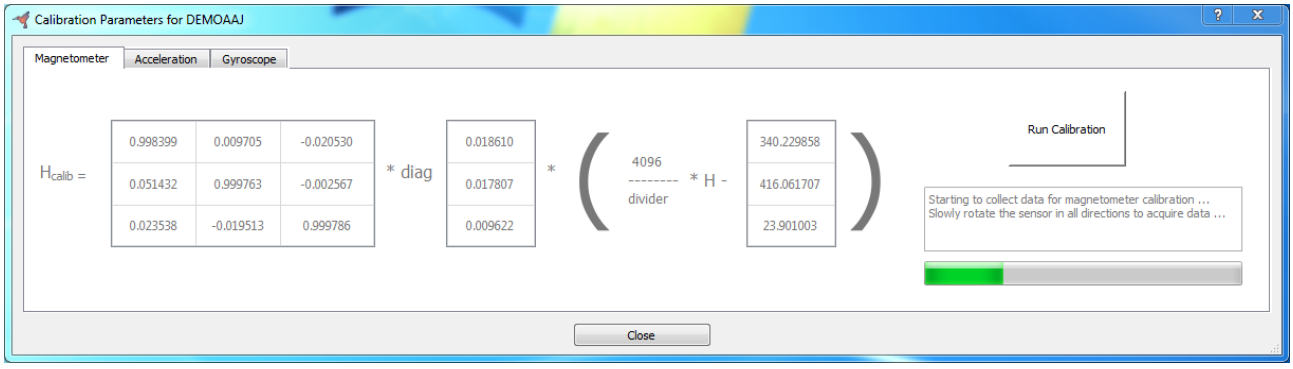
9. is the graphical view of the gyroscope sensor values.

3.4 Calibrating the magnetometers

If you want to view the parameters of the magnetometers, the accelerometers, or the gyroscopes, you can select a sensor and press on “Calibration Parameters” hidden in the “Device” menubar item. The dialog looks as follows:







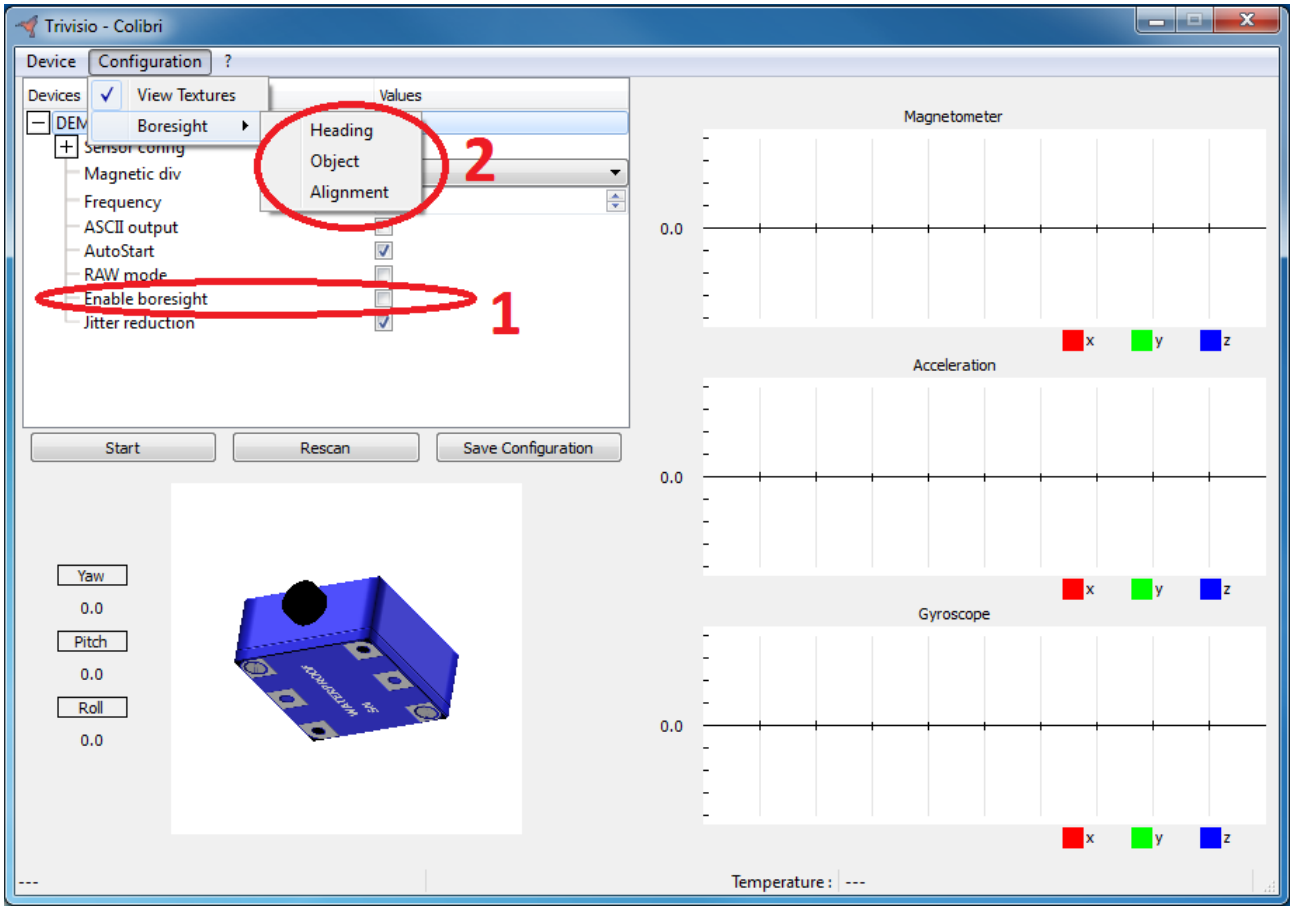
If you want to calibrate the magnetometers, press the “Run Calibration” button. Then you have to turn the sensor in smooth motions, so that as many different angles as possible can be measured. If the calibration was successful, the parameters can be stored to the sensor or dropped otherwise.

3.5 Boresighting

Concerning boresighting, the user is able to enable boresighting in the tree view of the sensor (see **1** in the image below). In order to set different alignments, the menu items in the Configuration->Boresight menu allow the user to set

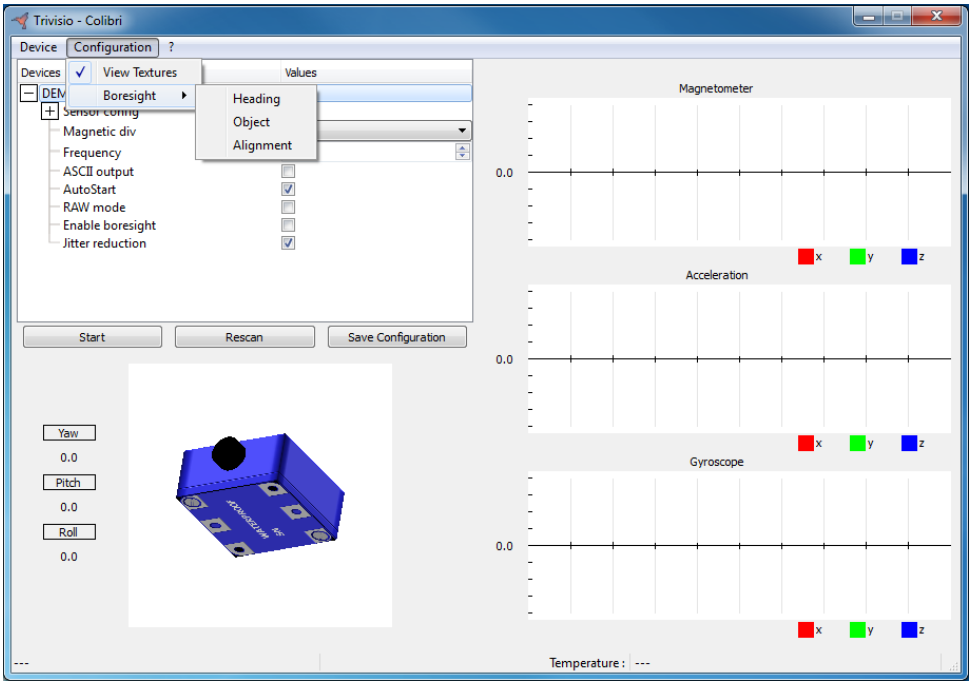
- object alignment (aligns the z-axis with gravity),
- heading alignment(assume yaw = 0),
- and complete alignment (current pose is matched with the nominal pose).

By selecting the different menu items (see **2** in the image below), the alignment will be set to the sensor.



3.6 Additional functions

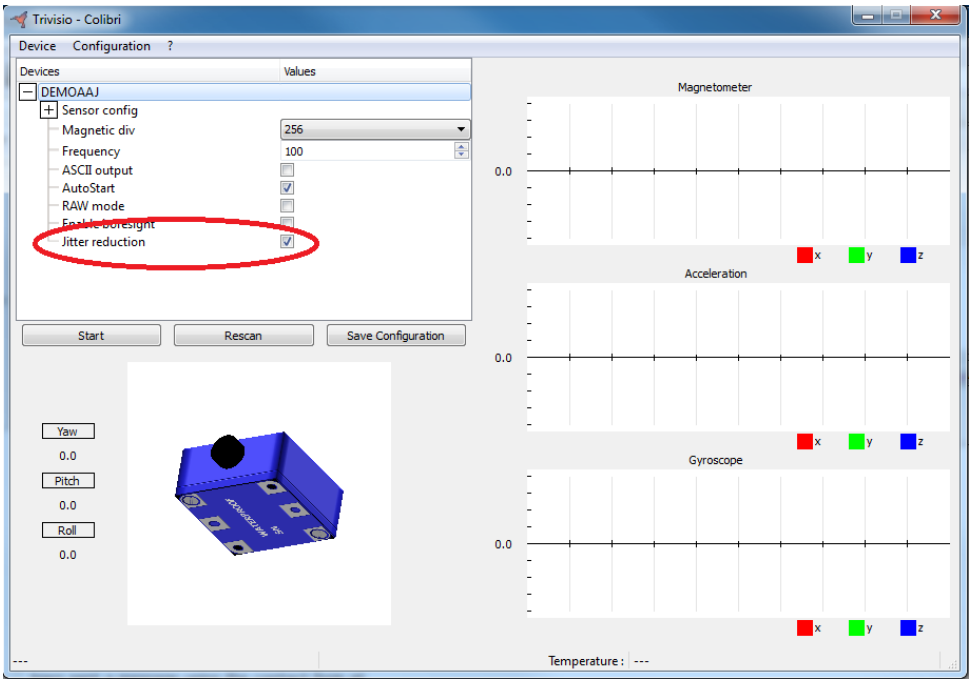
3.6.1 Disable drawing of textures



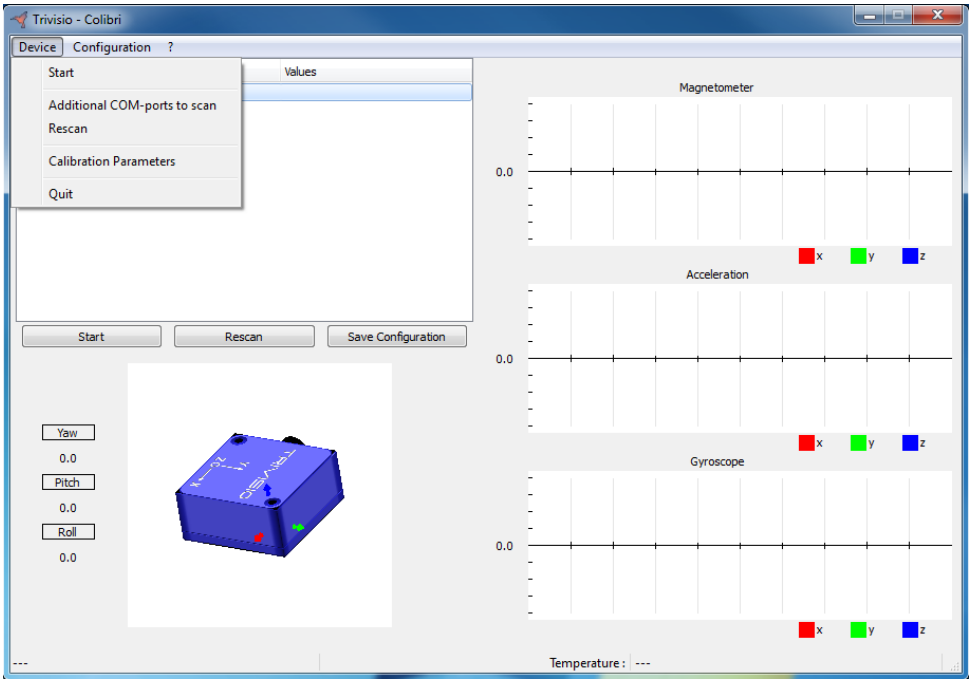
The menu bar contains the *Configuration* menu entry (see image above). You are able to enable or disable the textures of the sensor.

3.6.2 Jitter Reduction

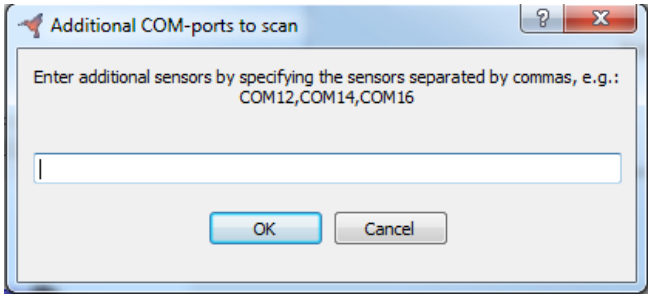
In order to enable or disable jitter reduction, there's a checkbox in the expanded tree view of the sensor. If you check it, it is enabled otherwise not.



3.6.3 Additional COM ports to scan



Considering your colibri tracker is connected via a virtual COM port or via RS232. For this purpose, you can manually add additional COM ports, which should be scanned. So, select the **Additonal COM ports to scan** from the menu and the following dialog will raise, where you can enter those ports. **Pay attention:** In order to scan for the new ports and have the devices in the list, you **must** press the **Rescan** button!



4 SDK

Within the SDK two example applications are provided. The first one is a console application named `testc` and the second one is a GLUT® application.

The two applications can be found in your installation directory in the `src` directory. For the Windows installation, predefined Visual Studio project files and a solution exist in your installation directory. The Fedora installation includes a predefined `Makefile`.

The needed header files are in your installation directory in the `include` folder. The used libraries are in the `lib` or `bin` directory.

Before using the SDK under Windows, you need to install `vcredist_x86.exe` to get the required Visual Studio libraries.

4.1 The test.c example

In the following section, we will have a deeper look at the first application.

The following headers are included in the testc application. The `TrivisioColibri.h` is the default header, which must be included.

```
1  #include "TrivisioColibri.h"
2
3  #include <stdio.h>
4  #ifdef WIN32
5  # include <windows.h>
6  #else
7  # include <unistd.h>
8  #endif
9
10 #define M_PI 3.14159265358979323846
```

The `main` function is the only function, where the Colibri sensors are scanned and the data are printed to the console. In addition the methods for getting and setting the configuration will be shown.

```
12 int main()
13 {
14   struct TrivisioSensor sensorList[10];
15   int sensorCount = colibriGetDeviceList(sensorList, 10);
```

The available devices can be triggered by calling the `colibriGetDeviceList`. The return value of the function is the number of available sensors. The two parameters of the functions are an array for the sensors and the size of this array.

```
17   void* imu = colibriCreate(100);
```

This function creates the colibri devices, whereas the parameter is the length of the buffer.

```
18   struct ColibriConfig conf;
19   char ID[8];
20
21   /* Diagonal matrices with diagonal element .68 yields approx 20Hz
22      bandwidth @ 100Hz */
23   float Ka[9] = { 0.68f,    0.00f,    0.00f,
24                   0.00f,    0.68f,    0.00f,
25                   0.00f,    0.00f,    0.68f };
26   float Kg[9] = { 0.68f,    0.00f,    0.00f,
27                   0.00f,    0.68f,    0.00f,
28                   0.00f,    0.00f,    0.68f };
```

```
29
30    struct TrivisioIMUData data;
31    double oldt = 0;
32    int i;
```

In this block an custom prefiltering parameters are generated and will later on be transmitted to the sensor. Using diagonal matrices with 0.68 as diagonal element yields independent filtering of each data channel with a bandwidth of approximately 20 Hz. Some variable need later are also defined.

```
34    printf("Number of Colibris found: %d\n", sensorCount);
35    if (sensorCount<0)
36      sensorCount = 10;
37    for (i=0; i<sensorCount; ++i)
38      printf("%s:\t %s (FW %d.%d)\n", sensorList[i].dev, sensorList[i].ID,
39                                      sensorList[i].FWver, sensorList[i].FWsubver);
40    printf("\n\n");
41
42    if (sensorCount<1) {
43      fprintf(stderr, "No Colibri sensors found\n");
44      return 0;
45    }
```

This block prints the available sensors to the console. The printing shows

- the device (`sensorList[i].dev`), e.g. COM1,
- the id of the sensor (`sensorList[i].ID`),
- the firmware version of the sensor (`sensorList[i].FWver`),
- and the firmware sub verion number of the sensor (`sensorList[i].FWsubver`).

```
46    if (colibriOpen(imu, 0, 0) < 0) {
47      fprintf(stderr, "Error while trying to access Colibri\n");
48      return -1;
49    }
```

Try to open a sensor by calling `colibriOpen`. The parameters are the `imu`, which was created earlier, a predefined configuration of the sensor, and a device port. The opening of the sensor fails, if the function returns a negative value.

Retrieve the current configuration of the acquired sensor, and set raw and ascii mode, frequency, and sensor configuration to the desired values. The configuration is then written back to the sensor using `colibriSetConfig` to take effect.

```
51    colibriGetConfig(imu, &conf);
52    conf.raw = 0;
53    conf.freq = 100;
54    conf.sensor = 1023;
55    conf.ascii = 0;
56    colibriSetConfig(imu, &conf);
```

Next preprocessing of the accelerometer and gyroscope data is activated, as well as jitter reduction.

```
58    colibriSetKa(imu, Ka);
59    colibriSetKaStatus(imu, 1);
60    colibriSetKg(imu, Kg);
61    colibriSetKgStatus(imu, 1);
62    colibriSetJitterStatus(imu, 1);
```

And the sensor settings are printed out.

```
64    printf("Colibri IMU\n");
65    colibriGetID(imu, ID);
66    printf("Device ID:         %s\n", ID);
67    printf("Sensor config:     %d\n", conf.sensor);
68    printf("Magnetic div:      %d\n", (unsigned)conf.magDiv);
69    printf("Frequency:         %d\n", conf.freq);
70    printf("ASCII output:      %d\n", conf.ascii);
71    printf("autoStart:         %d\n", conf.autoStart);
72    printf("RAW mode:          %d\n", conf.raw);
73    printf("Jitter reduction: %d\n", colibriGetJitterStatus(imu));
```

Start the colibri by calling the function `colibriStart`.

```
76    colibriStart(imu);
77    for (;;) {
78      colibriGetData(imu, &data);
79      if (data.t > oldt) {
80        float eul[3];
81        printf("Time: %6.2f\t", data.t*1e-4);
82        printf("Temp: %6.2f\t", data.temp);
83        printf("Acc: %6.2f, %6.2f, %6.2f\t", data.acc_x, data.acc_y, data.acc_z);
84        printf("Gyr: %6.2f, %6.2f, %6.2f\t", data.gyr_x, data.gyr_y, data.gyr_z);
85        printf("Mag: %6.2f, %6.2f, %6.2f\t", data.mag_x, data.mag_y, data.mag_z);
86        printf("Quat: %6.2f, %6.2f, %6.2f, %6.2f\t",
87               data.q_w, data.q_x, data.q_y, data.q_z);
88        colibriEulerOri(&data, eul);
89        printf("Euler: %10.4f, %10.4f, %10.4f\n",
90               180/M_PI*eul[0], 180/M_PI*eul[1], 180/M_PI*eul[2]);
91        oldt = data.t;
92      }
93 #ifdef WIN32
94      Sleep(2);
95 #else
96      usleep(2000);
97 #endif
98    }
```

The data will be read out of the sensor by the function `colibriGetData`. The euler orientation can be read out by calling `colibriEulerOri`.

```
100   colibriStop(imu);
101   colibriClose(imu);
```

The sensor is stopped by calling `colibriStop` and closed by `colibriClose`.

4.2 The opengl example

As the `testc.c` was already explained in detail, only new functions or shortcuts will be shown here.

Key shortcuts:

'n' toggles visualisation of euler angles as numbers

'c' toggles visualisation of cube

'q' quits the program

Enabling jitter reduction is done by calling `colibriSetJitterStatus(imu, true)`.
Three different types of bore-sighting are provided:

'h' yaw adjustment (assume yaw = 0 when 'h' is pressed)

'o' align the up axis of the sensor with gravity, keeping the yaw as it is

'a' complete alignment (current pose is matched with the nominal pose) (h+o)

'r' reset alignment (undo any previous adjustment)

Boresighting is activated with colibriSetBoresight (imu, 1) and deactivated with colibriSetBoresight (imu, 0). To boresight, use colibriBoresight (imu, type), where type should be one of: COLIBRI_HEADING_RESET (for heading rest), COLIBRI_OBJECT_RESET (for object reset), and COLIBRI_ALIGNMENT_RESET.

4.3 Simple Import mechanism for data profiling

In order to log the output to a file, use `testc.exe` (Win32) or `testc` (Fedora) to log data as follows:

Windows: `testc.exe > YOUR_FILE_NAME.txt`

Fedora: `testc > YOUR_FILE_NAME.txt`

The file can be imported with MS Excel or OpenOffice. For the latter, the delimiters are Tab, Comma and Space.

The settings for the data format are (Advanced Text Import Settings):

- Decimal separator: ','
- Thousands separator: ' '
- Uncheck "Trailing minus for negative numbers"