

Sensory Robotics - optoforce

Bump-detection with a 3D force sensor

Goal:

To detect terrain bumps with the Optoforce 3D force sensor.

Short description of the exercise:

Software environment: MATLAB

Tools to use during this lab:

- Optoforce 3D force sensor;
- insulator tape: to create bumps on the table;
- optional: lego-kit, if you would like to realize a construction to provide smoother sliding to the sensor.

Please, before the measurement read carefully the description and datasheet of the Optoforce sensor:

- `optoforce__general_description.pdf`: the general description about the operational principle of the sensor,
- `optoforce__datasheet.pdf`: the datasheet of the sensor.

During the measurement, we will pull the sensor upside down on a smooth surface (table). The surface should contain 3-5 stripes (bumps) perpendicular to the movement of the sensor. The goal is to detect these bumps.

Description of the measurement:

As a preparation: please stick 3-5 insulation stripes to the table, and please also cover the silicon-semisphere of the sensor.

Optionally you can create a wheeled construction from the lego-kit, in order to realize a smoother movement during the pulling process of the sensor.

The sensor is available to us through serial port, so please check under Control Panel / System / Device Manager which portnumber is allocated by the system to the sensor: also update the sample code according to this value.

About the measurement data:

The Data Acquisition Board (DAQ) of the sensor sends continuously the measurement data as 14 bytes long packets. One packet contains the followings:

```
55 67 [config] [s1H] [s1L] [s2H] [s2L]  
      [s3H] [s3L] [s4H] [s4L] [tempH] [tempL] [checksum]
```

The computation of the checksum (on 8 bits):

```
config+s1+s2+s3+s4+temp
```

The computation of the individual euclidean coordinates from the 4 raw sensorial information:

$$\mathbf{x} = \mathbf{s1} - \mathbf{s3}$$

$$\mathbf{y} = \mathbf{s2} - \mathbf{s4}$$

$$\mathbf{z} = (\mathbf{s1} + \mathbf{s2} + \mathbf{s3} + \mathbf{s4}) / 4$$

(The actual raw value of a specific sensor builds up from two 8-bit numbers of course, as an example: $\mathbf{s1} = 256*\mathbf{s1H} + \mathbf{s1L}$.)

Detailed steps to do:

1. Please check, which port is allocated to the sensor by the operating system, update the sample code (`optoforce_test.m`).
2. Please try to understand the sample code, then run it (even multiple times) and try to observe the sensitivity and time-responsivity of the sensor.
3. Please modify the sample code (or create a brand new one) according to these steps:
 - a. first you should capture the output signal of the sensor for a few seconds,
 - b. then please save the recorded data to a `.mat` file,
 - c. then please load the saved archive,
 - d. and finally please somehow detect the peaks of the signal, leading to the detection of the bumps during the earlier movement. (This part can be solved even with a simple thresholding with a fixed constant value, but please try to create a more sophisticated solution, like an adaptive approach. You can consider as a known fact the number of bumps during the track of the sensor. Please do not use the MATLAB's `findpeaks` function and other high level built-in solutions!)

Figure 1 is an exemplary image showing the resultant of the force vector and the detected bumps.

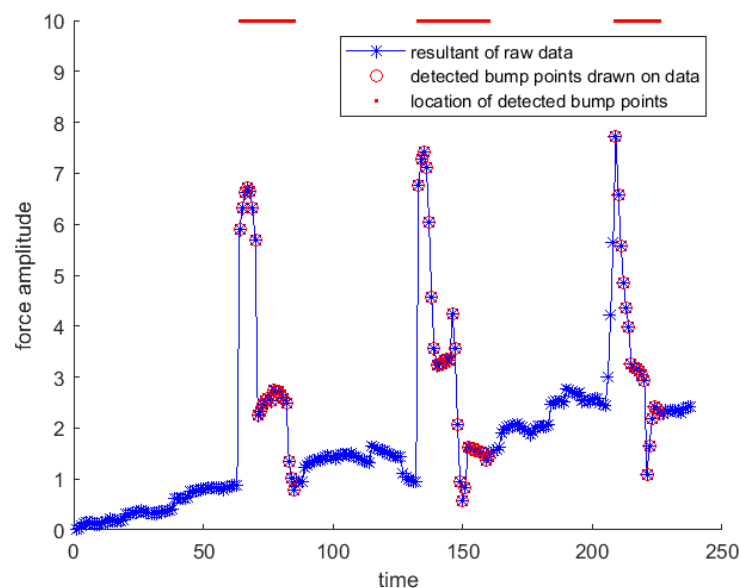


Figure 1: a sample result of a measurement over 3 bumps

Available source-codes:

Already given:

- `optoforce_test.m`: sample code about the handling of the sensor;
- `close_serials.m`: the only aim of this script is at to be able to close serial objects accidentally left without reference.

Please prepare:

- a script, which saves the measurement, then loads the archive file, plots the signals; does the bump detection you are planning to do.

What and when to upload:

What:

- all of your modified/new source codes,
- all of your `.mat` files,
- your report.

Please also make some comments/description about your detecting algorithm in the report.

Deadline: indicated in moodle.

Thank you.

Miklós