User Manual

Editor:

Version 0.1

Status

Reviewed	Pontus Erlesand	2016-12-14
Approved		

PROJECT IDENTITY

2016/HT, TSKS05-POZYX

Linköping University, (ISY)

Group members

Name	Responsibility	Phone	Email
Rasmus Vilhelmsson(RV)	Project Leader (PL)	0700 423 400	rasvi146@student.liu.se
Pontus Erlesand(PE)	Head of documentation (DOC)	0707 964 955	poner538@student.liu.se
Markus Petersson(MP)	Head of sensor fusion	0703 823 630	marpe163@student.liu.se
Ching-Hsiang Yang(CY)	Head of hardware	0737 227 489	chiya469@student.liu.se
Susmita Saha(SS)	Head of testing	0720 451 029	sussa301@student.liu.se

Group e-mail: pozyx.cdio@gmail.com

Customer: ISY, Linköping University, 581 83 Linköping

Customer contact: Danyo Danev, 013-281335, danyo.danev@liu.se

Course leader: Danyo Danev, 013-281335, danyo.danev@liu.se

Supervisor: Trinh Van Chien, 013-282643, trinh.van.chien@liu.se

Contents

Document history

Version	**Date**	**Changes**	**Sign**	**Reviewed**
0.1	2016-12-14	First draft		PE

1 Introduction

This user manual is part of the course TSKS05, Communication Systems CDIO, in 2016.

As the result of the project, a system is designed using the Pozyx platform to enable tag positioning and tracking. The purpose of this document is to provide a user guide on how to use the developed system. This document gives a short description about each component of the system, as well as present the procedures of positioning and tracking using it.

2 Definition of terms

Term	Description
2D	Two dimensions
3D	Three dimensions
LOS	Line of sight
GUI	Graphical User Interface
UWB	Ultra Wide Band
USB	Universal Serial Bus
TOA	Time of Arrival

3 Hardware

In this section, we present the hardware components used in the system.

3.1 Hardware components

The components are shown in figure 1. The list of components follows:

1. A computer with MATLAB on Windows
2. Pozyx anchor
3. Pozyx tag
4. Arduino board
5. USB cable
6. Power adaptor for anchor

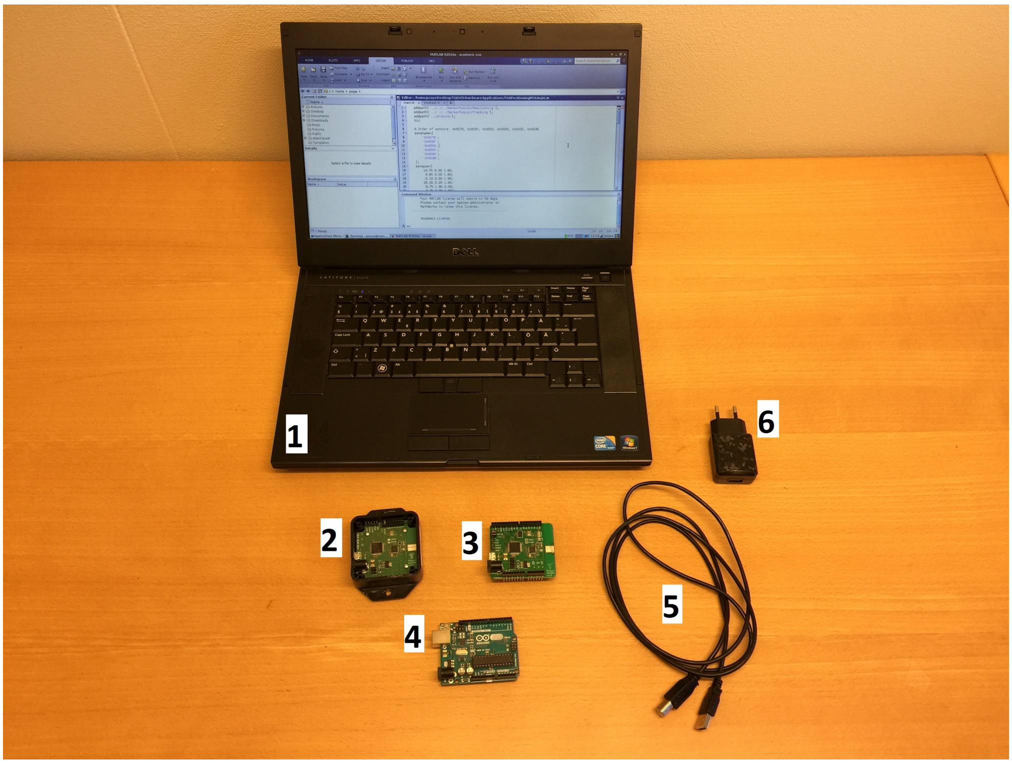

Figure 1: Hardware components of TP^3

3.2 Tested hardware versions

The system was tested using the following hardware:

- Dell Latitude E6510
- Arduino UNO R3
- Pozyx-shield v1.3
- Pozyx-anchor v1.3

3.3 Setup of the Hardware

The components of the system has been presented in section 3.1. The following steps are needed to set up the hardware:

- Deploy the Pozyx anchors throughout the area in which tracking/positioning is to be performed. Keep in mind that at least four anchors in LOS are required to enable positioning/tracking in 3D and three anchors for 2D.
- Longer distances between anchors give better resolution in the direction of separation. For example, having anchors at largely different heights will increase the precision of the altitude measurements.
- Connect power to the anchors and connect the tag to the computer.

3.4 Working Principle of the Hardware

In the Pozyx platform, Pozyx anchors and tag communicates through UWB. The maximum range of UWB is 100 meters with LOS, but shortened in indoor positioning.

The Pozyx shield is fitted onto the Arduino board and connected to the computer with a USB cable. The Arduino board works as an interface between the Pozyx shield and the computer. The tag sends the UWB signals and gets response from Pozyx anchors. The tag uses the signal to estimate the range to each anchor. This data is sent to the Arduino board over I^2C. The Arduino board transmits the data over serial port to a personal computer for processing. The MATLAB application estimates the tag's position and displays the results in a GUI.

4 Software

The software components required for the system are listed below:

- Windows operating system (for GUI)
- MATLAB
- Arduino IDE[1]
- Pozyx Arduino Library[2]

[1] `https://www.arduino.cc/`
[2] `https://github.com/pozyxLabs/Pozyx-Arduino-library`

4.1 Version requirements

MATLAB version 2016 or later is required. All tests have been done on MATLAB 2016b. TOA positioning requires the MATLAB Optimization toolbox to function.

The Arduino IDE is needed to program the Arduino board, and also provides the required drivers for serial port communication. Arduino IDE version 1.6.11 with Pozyx Arduino library version 0.9 was used to compile and upload the programs that control the Pozyx tag.

4.2 Repository structure

The software components of the system can be accessed from the project's git repository[3]. The repository is organized into several modules, separated into directories. The main modules are:

GUI The MATLAB GUI frontend.

Hardware Interface to the Arduino board and Pozyx tag, contains also several test applications.

SensorFusion MATLAB code for data processing, such as positioning and tracking.

4.3 Hardware-software interface

The final product includes a MATLAB GUI frontend and Arduino programs that control what data to be transmitted via serial port (over USB). Care should be taken to ensure that the corresponding Arduino program is uploaded before using the MATLAB GUI.

The `Arduino` class (`/Hardware/Applications/Arduino`) provides an interface for receiving data from Arduino into MATLAB for processing, and is also used in the GUI.

4.4 Setting up and running the system

To use the system, the correct anchor positions, as well as their 4-digit hexadecimal identifiers, need to be configured in the MATLAB scripts. Additionaly, if utilizing the built-in positioning in the Pozyx platform, the anchor positions also need to be presented in the Arduino program.

Use the Arduino IDE to upload code to the Arduino board. Which program should be uploaded depends on the positioning method. If utilizing the built-in function of Pozyx positioning, the companion Arduino program in `GenerateTestData` can be used. If doing TOA positioning with RSS estimation, use the program in `TOAPositioningRSS`.

After making sure that the correct program is uploaded using Arduino IDE, open the desired GUI frontend located in the `GUI` directory.

The file `gui_1_TOA.m` is the preferred program for this application, hence this document and the Technical Documentation are rendered to suit this program. The second choice, `gui_1.m`, utilizes Pozyx built-in TOA-algorithms where the estimations are of a bit less quality, especially in corridors, but is also fully functional.

[3] `https://github.com/marpe163/TSKS05`

Before the GUI is initialized, the program opens two input dialog boxes, one after the other (see figure 2) where the user has to fill in some information about the anchors and the tags. This includes the serial port used, the number of anchors used, if a Kalman filter or an extended Kalman filter is used, the type of filter used for smoothing, cutoff frequency for smoothing and moving average order. This information can be predefined to have default values. If the user wants to use the default values he/she only has to press the OK-button in the dialog box. The filter for smoothing can be changed at any time after the program has been initialized but not when the system is running and estimating the positions and trajectories. However, the cutoff frequency and moving average order can not be changed once the program is initialized.

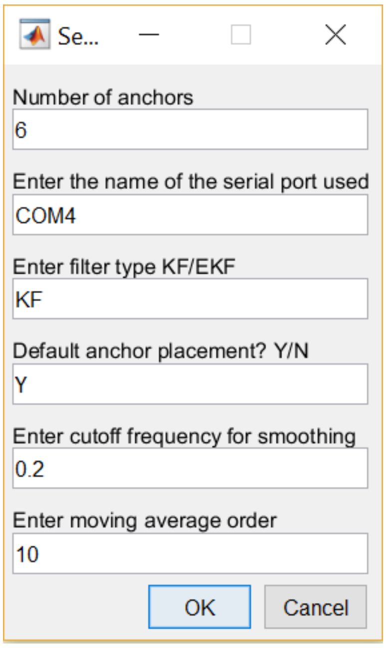

Figure 2: Input dialog box

When the GUI is successfully started, simply click the toggle-button `Start` to start the system. The blue squares on the map indicates the anchor positions, the green square is the origin, the red circle is the tag's estimated position and the red line is the estimated trajectory. The tracking and positioning can be stopped by clicking the toggle-button again which now says `Stop`. When the tracking and positioning is stopped, the user can choose to clear trajectory on the map and change the smoothing algorithm for the trajectory. Figure 3 illustrates the GUI when it is running the program. In addition to the tracking and positioning on the map, the GUI also provides three plots, two for velocities in x and y directions and one for altitude of the tag.

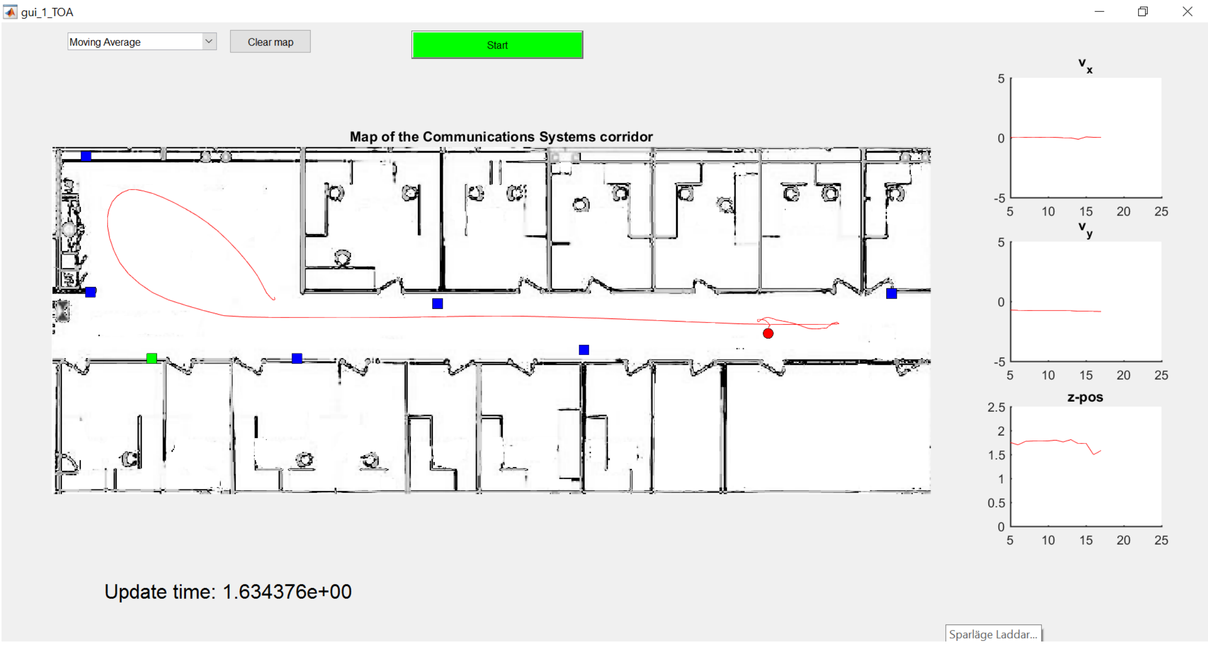

Figure 3: The systems GUI when the program is running

4.5 Additional examples

The MATLAB scripts in the directories under `Hardware/Applications` are useful in testing the hardware setup. These are meant to be run individually, and comes with the Arduino program needed. They also run on both Windows and Linux.

Additional information on testing the system can be found in `Hardware/README.md`.