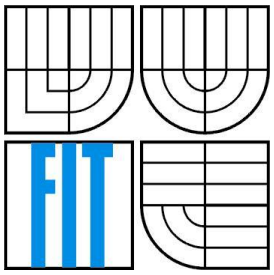

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

Brno University of Technology

SOFTWARE-RECORDING SYSTEM OF ROBOTIC DATA FOR SEVERAL SYNCHRONIZED SENSORS

AUTHOR Ing. Marek Šolony, Ing. Lukáš Maršík

1 Introduction

Mobile robots, especially the ones involved in difficult autonomous task employ multiple sensors to collect as much data about the environment as possible. To develop an application for the autonomous mobile robot, the testing data containing multiple scenarios have to be prerecorded so the final application can be evaluated. This software was created for the purpose of recording data from multiple sensors using global synchronization timestamp to achieve proper synchronization of data recorded on multiple machines.

Mobile robots may be equipped with multiple sensors connected to different processing units. Each of those units solve different tasks (object detection, localization, path planning, etc.) and simultaneously communicate with each other to use or fuse results for better accuracy and robustness. To develop such applications, much time is invested in building a dataset containing multiple prerecorded scenarios of robot performing desired tasks. This data has to be perfectly synchronized to achieve authentic reconstruction of events occuring on multiple sensors.

To achieve the synchronization we employ communication over RS232 serial port. All machines participating in recording are receiving same synchronization timestamps, so each frame of data receives a reference point in time in which its capture occured. Later while replaying the data it is easy to run simulated time and publish the data with corresponding timestamps.

The software is designed to work with following sensors: cameras, depth sensors (Kinect, etc.), LIDARs or radar signals, but the software architecture allows other extensions. Further, the system receives internal data from defined robotic platform and is able to store other information, for example odometry.

2 Description

Our software is composed of two packages, each package serves for different recording configuration. First one – KinRecord was used to record only Kinect and radar data without Robot Operating System (ROS) [1]. Second package – ROS_Recorder was created to work in ROS, utilizing its drivers and communication channels. This package also allows recording of laser data, spatial configuration of robot and also odometry. Detailed description of those packages can be found in following sections.

The synchronization is based on receiving of global timestamp and adding it to the data. This is achieved using one-way communication through RS232 serial port.

2.1 RS232 serial port communication

Our both applications assume, that the synchronization data are available on RS232 serial port. Except the synchronization timestamps, also the odometry data are included.

The comunication specifications are:

- Baud rate: 115200
- Parity: even
- Stop bits: one
- N of data bits: 8

The communication protocol:

- header:"TT"
- Message length: 1B
- pos X: 4B (mm)
- pos Y: 4B (mm)
- heading: 2B (0.01deg)
- wheel velocity: 2B (mm/s)
- steering angle: 2B (0.01deg)
- timestamp 4B (ms)
- CRC

2.2 Recording with ROS

This branch of SW was made using Robot Operating System (ROS). ROS is a software framework for robot software development, providing operating system-like functionality on a heterogenous computer cluster. ROS provides standard operating system services such as hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes, and package management. It is based on a graph architecture where processing takes place in nodes that may receive, post and multiplex sensor, control, state, planning, actuator and other messages. The library is geared toward a Unix-like system [1].

We focused on creating recording SW using ROS, because it is very convenient for developing applications for mobile robots. List of recorded data can be found further in this document.

Prerequsities:

- ROS electric: http://www.ros.org/wiki/electric/Installation/Ubuntu
- For recording, Kinect has to be plugged in, and ROS node for Kinect (openni_camera) has to be running: "rosrun openni_camera openni_camera"
- Optional, data published on /scan topic (mostly lidar data) can be recorded.

Installation:

To install this SW, place the directory with both but_com_receiver and but_kinect_recorder to your ROS workspace and execute following commands:

- rosmake --rosdep-install but_com_receiver
- rosmake --rosdep-install but_kinect_recorder

Usage:

This software consists of two ROS nodes. After successful build, the recording can be started followingly:

- Run BUT COM-port receiver: *./receiver <COM port number> [-v]*
  "-v" enables printing read input on screen
  This node is used to receive data from RS232 port and publishes them through ROS topics.
- Run BUT Kinect Recorder: *./recorder <bagfile name> <record laser - 1/0> <record every Nth frame>*
  This node is used to record data from external sensors created for recording data in E80, Italy. Before running this program, but_comm_receiver should be correctly running.

Recorded data:

/sync - synchronization time

/scan - laser scan from lidar

/tf - transformations for odometry

/camera/depth/image – depth image

/camera/rgb/image_color – RGB image

/camera/depth/camera_info

/camera/rgb/camera_info

Output:

Output is stored into single .BAG file specified as parameter of but_kinect_recorder. Data in this file are easily accessed through ROS commands or ROS C++ API.

2.3 Recording without ROS

This SW is located in KinRecord directory and can be compiled and run on Windows platform. The directory contains source codes - Visual Studio 2010 project. It is simple program that doesn‘t need ROS, and can record data from Kinect sensor and radar. The data, unlike recording with ROS, is stored into separate files.

Prerequsities:

- Windows operating system
- OpenNI [2] drivers: http://openni.org/Downloads.aspx , both OpenNI stable, and PrimeSense Sensor Module for OpenNI are required to be installed.
- Microsoft Kinect, plugged in and set to use OpenNI drivers.
- Synchronization data available on RS232 serial port.
- Radar output plugged in microphone input.

Installation:

After fulfilling prerequisites about OpenNI drivers, the project can be built in Visual Studio.

Usage:

Afer Kinect is plugged in, OpenNI drivers are set and synchronization data are published on RS232 port, executing KinMap.exe runs the recording program. With SPACE-BAR recording can be started and ended.

Recorded data:

- RGB video from Kinect RGB camera, 640x480
- Depth map - stored to binary file, each frame has 640x480 (each pixel has 16B)
- Data from radar to raw sound file
- LOG file with timestamps

Format of the LOG file:

The LOG file contains timestamps with pointers to the data files, showing how far were the data recorded at the current time:

time_stamp video_frame N_written_sound_buffers

- time_stamp – actual time stamp in ms
- video_frame – actual number of video frame (video frame and depth frame are always synchronized, so only one number is enough)
- N_written_sound_buffer – actual position in sound file (N * 4096 B)

Output:

Output data are saved into separate files in actual working directory. After recording another file, its suffix is increased by 1, so the files will not overwrite.

3 Results

We successfully created a recording system of robotic data for multiple synchronized sensors. It is capable of recording data from robotic sensors with synchronization, which is crucial for replaying the data. The source codes are available in this package, and one familiar with ROS can easily modify the application to record data from any type of sensor supported by ROS. This SW was tested for recording in E80[1] company in Italy, where the data was recorded from our sensors placed on Large Goods Vehicle (LGV, see Figure 1) on one PC, and data from built-in sensors on another PC. This data are used to test the applications developed for this company.

Figure 1: Mobile LGV equipped with multiple sensors.

[1] http://www.elettric80.com/

4 References

[1] Robot Operating System: http://en.wikipedia.org/wiki/ROS_(Robot_Operating_System)

[2] OpenNI: www.openni.org