Developing an Indoor Localization and Mapping System for iRobot using Microsoft Kinect and Raspberry Pi

Thomas Lavastida[1], Shuai Zheng [3,2], Wuyi Yu [1,2], and Xin Li [1,2]
[1]School of Electrical Engineering and Computer Science, Louisiana State University
[2]Center for Computation and Technology, Louisiana State University
[3]Xi'an Jiaotong University, China

Abstract

Solving the Simultaneous Localization And Mapping (SLAM) problem allows a mobile robot to use only images collected by its equipped sensor to answer two fundamental questions: What does my environment look like and where am I located within that environment? Effective SLAM has many applications, such as using autonomous robots to discover/navigate unfamiliar environment or inspect complex hazardous system [1].

This project aims to develop a hardware platform to implement and evaluate a novel SLAM scheme developed in LSU Geometric and Visual Computing (GVC) group. This development uses an iRobot carrying a Microsoft Kinect to gather surrounding world information. RGB-Depth data acquired by the moving Kinect sensor are efficiently transferred through WIFI to a workstation, where SLAM is computed. Effective and adaptive robot control and data communication are implemented on a Raspberry Pi, an inexpensive embedded programmable board,.

Figure 1: (Left) Localization and mapping attempts to create a 3 dimensional reconstruction of an environment. (Top) The Microsoft Kinect, carried by an iRobot, is used to collect data to accomplish this.

Background

This project involves integrating several different components to create a functioning robotic system:

- Microsoft Kinect – A multi-sensor device containing an infrared depth sensor and RGB camera.
- Raspberry Pi – An inexpensive, credit-card sized computer that uses an ARM processor. Typically runs a GNU/Linux based operating system.
- iRobot Create 2 – Programmable mobile robot based on the iRobot Roomba 600. A serial port is available as a simple interface to allow control of itself. [2]
- TCP/IP Networking – Allows for reliable communication between two devices connected via the internet.
- Zlib – A simple to use, open-source data compression library [3].

Figure 2: (Left) The Raspberry Pi and (Right) the iRobot Create 2

Method

The function of the system can be divided into two main tasks: collecting the data from the Kinect and controlling the iRobot. In order to accomplish the task of sending RGB and depth data from the Raspberry Pi to another computer, two programs need to be written. One of the programs runs on the Raspberry Pi as a server and collects the frames of data from the Kinect and then transmits the data using TCP. The second program runs on the other computer and connects to the Raspberry Pi, then continually loops to receive the data from the Raspberry Pi. To control the motion of the iRobot, a python program was implemented to read commands from a text file line-by-line and send the command to the iRobot using serial port communication.

System Architecture

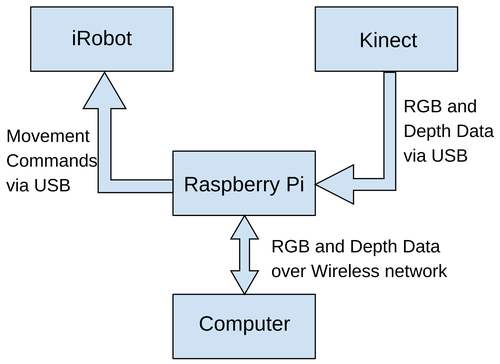


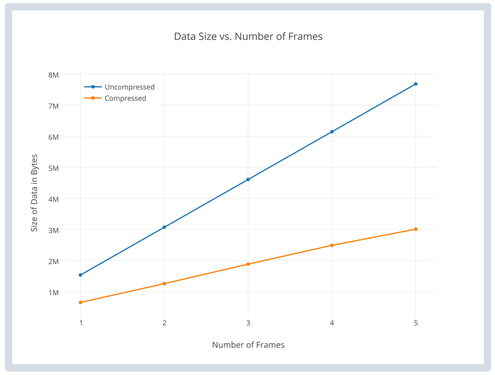
Figure 3: Diagram of system architecture. The Raspberry Pi acts as a hub and controller for the rest of the devices in the system.

```
Begin:
allocate buffer for N frames

Loop:
repeat N times:
    get RGB and Depth data
    and place data in buffer

if compression is on:
    compress data in buffer

send data in buffer

go back to Loop
```

Figure 4: Pseudo-code algorithm for sending the data from the Raspberry Pi. A similar but opposite routine is used to receive the data.

Results

After implementing the base of the system, we evaluate a possible scheme to improve the performance of the data transfer – data compression. The graphs below detail the results of testing a modified system implementing the compression routines provided by zlib.



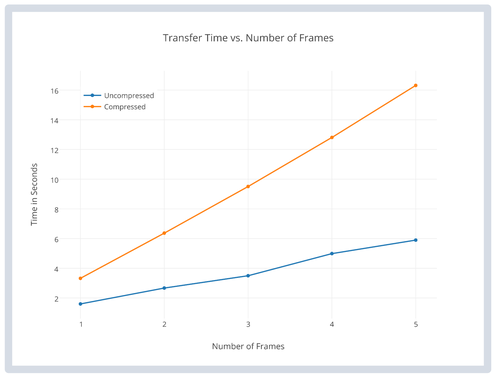


Figure 5: (Above) Front and rear views of the completed robot in operation. (Top Left) Results of data size against number of frames for the test of compressing Kinect data. (Left) Results of timing experiments for the test of compressing Kinect data.

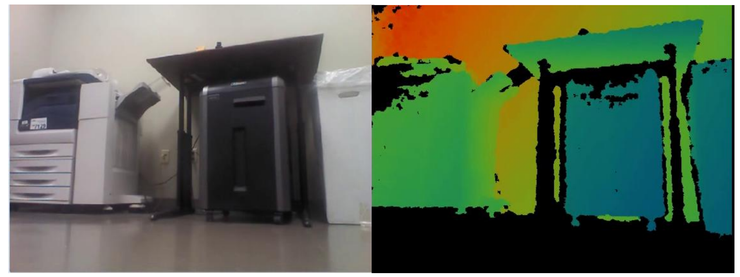


Figure 6: RGB and depth data collected by the Kinect. These are used to progressively reconstruct the environment as demonstrated in Figure 1

Discussion

Initially, the system successfully operated and was able to collect data from the Kinect, transfer data to another computer, and move through the test environment by controlling the iRobot. However, data transfer was relatively slow, transferring 1 set of frames in about 1.6 seconds. Ideally, data transfer should occur at 30 sets of frames per second, the rate the Kinect produces data. This ideal rate is probably not attainable due to limits in the computation power of the Raspberry Pi and limitations in network latency. While data compression was able to reduce the size of the data effectively, the Raspberry Pi did not perform well in being able to compress and transfer the data in less time than sending the data without compression.

Future Work

Several directions can be explored to improve the performance of the overall system that has been built during this summer:

- The data transfer program can be made more robust so that captured frames are placed in a sliding-window buffer where they are then transferred as they are made available depending on network conditions.
- Other compression schemes can be tested to see if a performance gain can be attained with the current hardware.
- Upgrading to Raspberry Pi 2 could bring a substantial performance gain.

Summary

Overall, implementing the basic data transfer and robot control system was a success. The results of this Summer provide a good starting point to make a more robust system to implement and test robot localization and mapping. Using compression to improve data transfer performance was not as successful as desired, but provided insights into the limitations of the hardware in the Raspberry Pi.

References

1. X. Li, W. Yu, X. Lin,and S. S. Iyengar, "On Optimizing Autonomous Pipeline Inspection," IEEE Transactions on Robotics, 28(1):223– 233, 2012.
2. "iRobot® Create® 2 Open Interface (OI) Specification based on the iRobot® Roomba® 600", www.irobot.com, 2015.
3. "zlib 1.2.8 Manual", www.zlib.net/manual.html, 2013.

Acknowledgements

This work is funded by National Science Foundation IIS-1320959, with additional support from NSF OCI-1263236 and the Center for Computation & Technology at Louisiana State University.