WiP Abstract: Framework for Surveillance of Vulnerable People using Depth Camera

Hee Jung Yoon\textsuperscript{1}, Ho-Kyeong Ra\textsuperscript{1}, Jin-Hee Lee\textsuperscript{2}, JeongGil Ko\textsuperscript{3}, and Sang Hyuk Son\textsuperscript{1}
\textsuperscript{1}Department of Information & Communication Engineering, DGIST, Republic of Korea
\textsuperscript{2}CPS Global Center, DGIST, Republic of Korea
\textsuperscript{3}Department of Software and Computer Engineering, Ajou University, Republic of Korea
\{\textsuperscript{1}heejung8, \textsuperscript{1}hk, \textsuperscript{2}jhlee07, \textsuperscript{1}son\}@dgist.ac.kr, \textsuperscript{3}jgko@ajou.ac.kr

I. INTRODUCTION

With the advancement of technology in various domains, many efforts have been made to design state-of-the-art classification engines using depth cameras. Being inspired by its potential of providing information at the skeleton level using a non-invasive infrared camera, many studies have been done to aid vulnerable people such as children, elderly, and people that are physically or mentally ill. However, most of these studies focus on the algorithms and processing of a single camera, and do not consider issues that are found in practical deployments. We present KinFrame, a framework that considers challenges and requirements of designing practical systems for vulnerable people and allows application developers to easily and efficiently setup large scale, multiple depth camera deployment. In this paper, we present how far we have developed KinFrame and discuss some of our future research directions.

II. IMPLEMENTATION

As shown in Figure 1, the five core parts of KinFrame are: client, server, cloud, database, and web. First, for the client, we use Kinect to gather RGB and depth information. After extracting data using Kinect SDK and packaging data in the local machine, the client connects and transfers the data to the server. On our first attempt of designing the client-server layout, we used two different servers for receiving and sending data to be viewed by the client. Many related works use a similar architecture [1]. However, over time, this design becomes problematic as the time synchronization between uploading and presenting data goes off. Therefore, we use SocketIO4Net, which enables socket communication to the Node.js server, and makes it capable to utilize a single server for uploading data from the client and presenting the data to the web browser. The data is then streamed and stored in the cloud and MongoDB. Simultaneously, the data is classified in real-time while presenting RGB and depth information in the web based GUI.

Using what we have implemented, we design a pilot study to demonstrate KinFrame. With a purpose to ensure school safety, we have deployed 10 Kinects in a hallway of a school. Figure 2 shows a screenshot of our web-browser based user interface where RGB images and skeleton joint data from multiple Kinect are displayed in real-time.

III. FUTURE WORK

Utilizing and connecting multiple depth cameras is challenging due to the difficulty of concurrently aggregating streaming data from all of the devices. Additionally, having this complex setup brings large scale issues such as bandwidth and data management problems. As part of our future work, we plan to adapt a flow control method to solve bandwidth issues when streaming data from multiple devices and use a data management technique to control constant flow of real-time information. For improved usability of the system, we also plan to design an alerting mechanism for quick emergency reports to parents and caregivers, and layout a user interface for them to verify emergency situations or analyze behavioral patterns of the vulnerable person being monitored.

ACKNOWLEDGMENTS

This research was supported in part by the DGIST Research and Development Program of the Ministry of Science, ICT and Future Planning of Korea (CPS Global Center), and the ICT R&D program of MSIP/IITP (14-824-09-013, Resilient Cyber-Physical Systems Research).

REFERENCES