Implementation Example with Ultra-Small PCs for Human Tracking System Based on Mobile Agent Technologies

Masaru Shiozuka†‡, Tappei Yotsumoto†‡,
Kenichi Takahashi‡, Takao Kawamura‡, Kazunori Sugahara‡

†System Engineering Department,
Melco Power Systems Co. Ltd.
Kobe, Japan
email: {Shiozuka.Masaru@zd, Yotsumoto.Tappei@zb}.MitsubishiElectric.co.jp

‡Graduate School of Engineering,
Tottori University
Tottori, Japan
email: {takahashi, kawamura, sugahara}@eecs.tottori-u.ac.jp

***Abstract*—In order to take security measures and crime prevention measures, companies have introduced human monitoring systems. However, operators are required to monitor many devices, such as cameras and sensors to track suspicious persons. We have proposed an automatic human tracking system based on mobile agent technologies. In this paper, we propose an implementation example with Ultra-Small PCs for the human tracking system. By using this Ultra-Small PCs, it is possible to construct a low cost human tracking system that is highly extensible.**

Keywords-Human tracking; Mobile agent; Raspberry Pi; Beacon.

I. INTRODUCTION

In order to take security measures and crime prevention measures, companies have introduced human monitoring systems. Operators monitor suspicious persons using cameras and sensors. However, if the number of cameras and tracking targets increase, tracking all targets must be difficult. As a result, operators could lose targets.

We have proposed an automatic human tracking system based on mobile agent technologies [1][2]. This system tracks targets by mobile agents instead of operators. The automation of tracking persons can reduce the burden of tracking and the number of incorrect tracking instances. There is another usage example of this system, that is, this system can track customers' behavior in department stores or collect their movement information for marketing decisions.

This paper proposes an implementation example with Ultra-Small PCs for human tracking system based on mobile agent technologies. Due to the recent advances in Internet of Things (IoT) devices, the costs of these Ultra-Small PCs have been lower. These Ultra-Small PCs, such as Raspberry Pi [3], can connect to the Internet, equip many sensors and easily collaborate with each other. We consider that we can construct human tracking system with low cost and highly extensible by using these Ultra-Small PCs.

We evaluate our implementation by the correctness of human tracking. We introduce the system into a real-life office, where a target person walks around. Then we observe and see if mobile agents can track the person correctly.

The remainder of this paper is structured as follows. In Section II, we provide a survey of related works. Section III illustrates the overview of our human tracking system. Section IV explains an implementation example. In Section V, the implementation is evaluated. Concluding remarks are provided in Section VI.

II. RELATED WORK

[4] and [5] proposed the method that tracks persons by analyzing a shooting range of camera. We propose the method that tracks persons by using many sensors, and we illustrate the simple implementation of the algorithms. When the system tracks persons, the system uses a Beacon Tag to identify each person. The burden of carrying a Beacon Tag is smaller because the Beacon Tag is compact.

Correa [6] proposed the method that tracks a person by analyzing a received signal strength indication (RSSI) value. We propose the method that can easily extend the area of tracking persons by placing sensors depending on the size of the floor. In our system, each sensor only needs to know its neighbor relation sensors, that is, each sensor does not need to know where all sensors are placed. By constructing the system like this, it is easy to connect or disconnect to or from the network. Suwa [7] also proposed the method that tracks a person by using a RSSI value of Bluetooth Low Energy (BLE). The system sends a RSSI value to a database server and estimates the persons' positions. Our system does not need such servers.

Stefano [8] proposed the system that supports visitors to browse artworks in museums. The system presents the cultural information when visitors come close to the artworks. In order to track the persons, visitors need to wear wearable

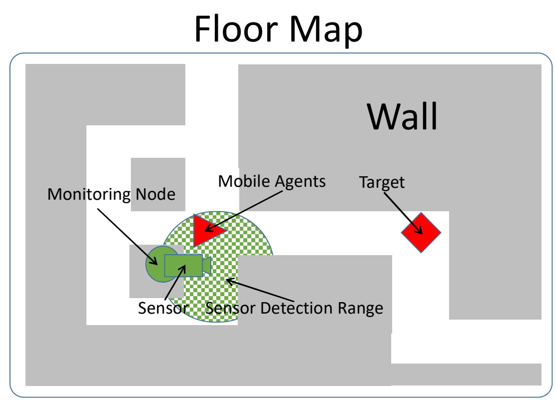

Figure 1. System Overview

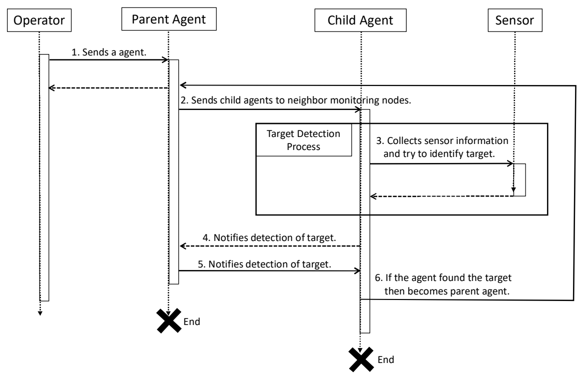

Figure 2. Human Tracking Algorithm

devices which collect the radio of Bluetooth and images of artworks.

In our system, the target person only needs to have the small Beacon Tag. Our system can keep the cost low even if the number of target persons increases because the Beacon Tag is compact and low price.

III. Human Tracking System Based on Mobile Agent Technologies

In this section, we explain the human tracking system in detail.

A. *System Overview*

Figure 1 shows our human tracking system overview. This system consists of the following elements.

- A Target is a person who is being tracked.
- A Sensor is a device which collects surrounding information and sends it to the connected monitoring node, that is, a Beacon receiver or a camera is a sensor. A sensor needs to be placed together with a monitoring node.
- A Sensor Detection Range is a range of detection which can track targets, such as a radio reception range or a camera's shooting range.

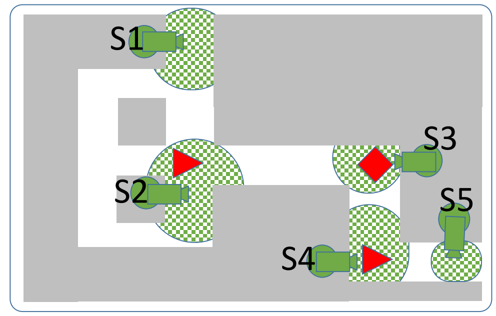

Figure 3. Example Of Neighbor Relations

- A Monitoring Node (referred to as a node) is a device which collects sensor information of Beacon Tags or cameras. The node constructs networks between each node.
  We define neighbor relations of each node. The node provides an execution environment for mobile agents.
- A Mobile Agent (referred to as an agent) is a program which identifies and tracks a person using sensor's information. An agent has features data (e.g., facial features, Beacon IDs) of a target person. A human tracking is implemented by moving an agent between nodes.

B. *Human Tracking Algorithm*

Figure 2 shows human tracking algorithm by using agents. A target is tracked using the following 6 steps.

1. In order to track a target, an operator sends an agent with feature data to the node where the target is.
2. When the agent arrives at the node, the agent creates its copies and sends them to other nodes. We define an original agent as a parent agent, and a copied agent as a child agent.
3. Child agents collect information from the node where they are sent. Each agent identifies persons using the sensor information and feature data. If it identifies a target person, it goes to step 4. If not, retries step 3 from head.
4. The child agent notifies the target detection to the parent agent.
5. The parent agent notifies the target detection to all child agents, and exits.
6. The child agent who detects the target becomes a new parent agent and goes to step 2, the other child agents exit. By repeating steps 1 to 6 agents can move nodes and automatically track the target.

C. *Neighbor Relations Of Nodes*

1) *Algorithm To Calculate Neighbor Nodes*

In order to move child agents between nodes, we define neighbor relations of nodes which a node has the possibility of passing next time. Child agents move to nodes based on these neighbor relations. For example, Figure 3 shows the floor map where S1 to S5 are placed. There are five nodes on this floor. Suppose a target person is in the S3 node, then the S2 and the S4 are only possible nodes that the target person passes next because the aisle from the S2 node to the S4 is a single road. Therefore, the neighbor relations of the S3 node are the S2 node and the S4 node. If the person passes the S1 node or S5 node, the person needs to pass the S2 node or the S4 node previously.

We define the algorithm to calculate relations between nodes. Defining vectors *S*, *B*, *D* and *P* as a set of node points, a set of branch points, a set of a sensor detection points and a set of all these points, respectively, we can obtain matrix *X* and *Y*, as *S*×*P*, as *P* × *P*, respectively. Here, matrix elements of *X* and *Y* are defined as follows:

$$X_{ij} = \begin{cases} 0, & \text{where the monitoring range of the camera } S_i \text{ does not include the point } P_j. \\ 1, & \text{where the monitoring range of the camera } S_i \text{ includes the point } P_j. \end{cases} \quad (1)$$

$$Y_{ij} = \begin{cases} 0, & \text{where the point } P_i \text{ and the point } P_j \text{ are not neighboring each other.} \\ 1, & \text{where the point } P_i \text{ and the point } P_j \text{ are neighboring each other.} \end{cases} \quad (2)$$

When $E_{ij} \geq 1$ in (3), the neighboring camera is overlapped with (n-1) points away from the monitoring range of the camera S_i.

$$E = X \bullet Y^n \bullet X^T \quad (3)$$

2) Neighbor Relations Localized Algorithm

In order to enhance the extension of neighbor relations, we define neighbor relations localized algorithm. By localizing the neighbor relation, each node does not need to know all the nodes on the floor but to know some nodes which have localized relationship. Matrix Xs_{ij} and Ys_{ij} are defined as follows.

$$Xs_{ij} = \begin{cases} 0, & \text{where the monitoring range of the camera } S_i \text{ does not include the point } Ps_j. \\ 1, & \text{where the monitoring range of the camera } S_i \text{ includes the point } Ps_j. \end{cases} \quad (4)$$

$$Ys_{ij} = \begin{cases} 0, & \text{where the point } Ps_i \text{ and the point } Ps_j \text{ are not neighboring each other.} \\ 1, & \text{where the point } Ps_i \text{ and the point } Ps_j \text{ are neighboring each other.} \end{cases} \quad (5)$$

Then, the neighbor nodes are calculated as follows.

$$Es = Xs \bullet Ys \bullet Xs^T \quad (6)$$

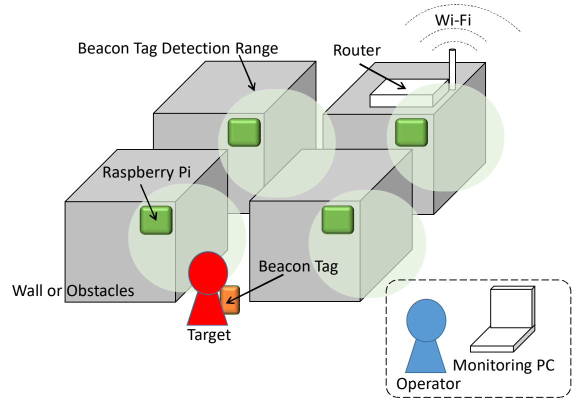

Figure 4. System Architecture

TABLE I. SYSTEM CONSTRUCT DEVICES

Raspberry Pi	
Machine model	Raspberry Pi 2 Model B
CPU	ARM Cortex-A7 900MHz
Memory	1GB RAM
OS	Raspbian GNU/Linux 8
Java	version 1.8.0
Bluetooth Dongle	PLANEX BT-Micro4
Wireless Wi-Fi Device	BUFFALO WLI-UC-GNM2
Beacon Tag	Aplix MB004
Monitoring PC	
OS	Windows7
Web Browser	IE11

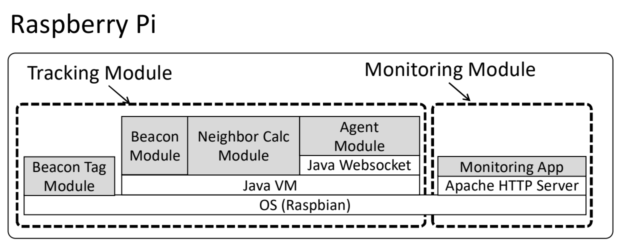

Figure 5. Software Architecture

IV. IMPLEMENTATION

In this section, we explain the implementation with Ultra-Small PCs of the human tracking system.

A. System Architecture

Figure 4 shows the system architecture. We use a Raspberry Pi as a node. One Raspberry Pi works one node. Table 1 shows the details of devices. Each Raspberry Pi is equipped with a Wireless Wi-Fi Device to connect to Wi-Fi and with a Bluetooth Dongle to receive the radio signal of the Bluetooth [9] from Beacon Tag which a target person has. Each node is connected to the router. We use the Bluetooth Dongle as a sensor. Our system can use a camera as a sensor. But we did not adopt camera systems because of their cost and their hard setting and managing points of view.

Moreover, by using cameras, privacy issues have to be considered. Finally, we use a monitoring PC to track the target persons.

B. Tracking Flow

The flow of tracking a person using this system architecture is as follows. Operators send an agent with feature data to the node where the target is. The feature data consist of a set of data, namely a person id and a Beacon Tag id. The person id identifies a person. A Beacon Tag id identifies a Beacon which the person has. When the person approaches a Raspberry Pi, the Bluetooth Dongle detects the person. The agent on the Raspberry Pi analyzes the radio of Bluetooth, and if the Beacon Tag id corresponds to the feature data, the agent judges that the person is the target.

C. Implementation Detail

Figure 5 shows the software architecture. There are two modules, as follows: the left-hand tracking module which is placed in all Raspberry Pi, and the right-hand monitoring module which is placed in an arbitrary Raspberry Pi.

1) Tracking Module

The tracking module implements the human tracking function. This module consists of four sub modules.

- The Beacon Tag Module implements a function that receives a radio signal from the Beacon Tag. We implemented it in C language and used BlueZ [10] whose open source projects provide Bluetooth stack protocol library. When this module receives a radio signal from the Bluetooth, it passes the radio signal to a Beacon Module.
- The Beacon Module is implemented in Java language and connects to a Beacon Tag module by TCP sockets and reads the radio signal of the Bluetooth.
- The Neighbor Calculation Module implemented in Java language obtains the neighbor relations and decides the destination where agents are sent to.
- The agent module is implemented in Java language and provides four functions as follows.
    - Sends a parent agent with the feature data to an arbitrary Raspberry Pi.
    - Identifies the target person by the radio signal of the Bluetooth from the Beacon Module.
    - Sends child agents based on the neighbor relations to the nodes. The child agents are serialized when they are sent, and de-serialized when they arrive at the next node.
    - Notifies the tracking information to the monitoring module by Websocket [11]. The tracking information consists of the agent status (parent or child), tracking human id, and its IP address.

2) Monitoring Module

The monitoring module is implemented in JavaScript and uses Websocket. This module shows the target person and child agents to the Web Browser on a floor map. To display the map, we use the Open Layers [12], which is an open source project map display library.

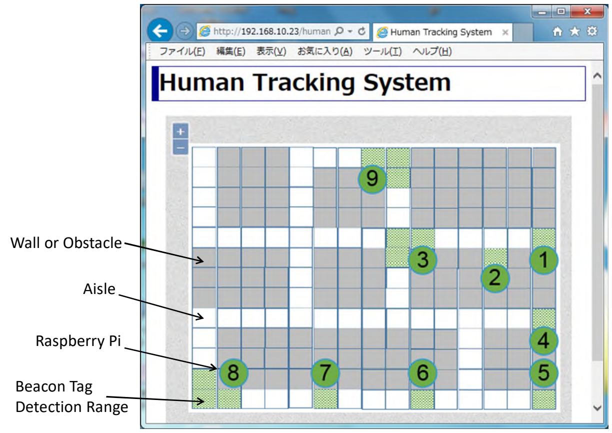

Figure 6. The Monitoring PC Web Browser

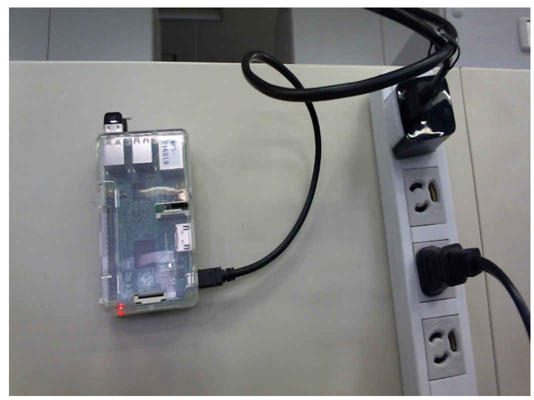

Figure 7. Example of installing a Raspberry Pi

TABLE II. NEIGHBOR RELATIONS OF NODES

Node Number	Neighbor Relation Nodes
1	2, 3
2	1, 3
3	1, 2, 4, 5, 6, 8, 9
4	3, 5, 6, 8, 9
5	3, 4, 6, 8, 9
6	3, 4, 5, 7, 8, 9
7	6, 8
8	3, 4, 5, 7, 9
9	3, 4, 5, 6, 8

V. EXPERIMENT

In this section, we evaluate our implementation of the human tracking system. We introduce the system into a real-life office.

A. Environment

Figure 6 shows the view of the monitoring PC using a Web browser. The map shows that it is one floor, the gray

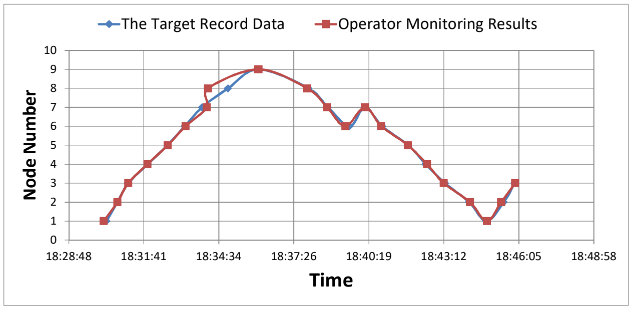

Figure 8. The First Experiment Result

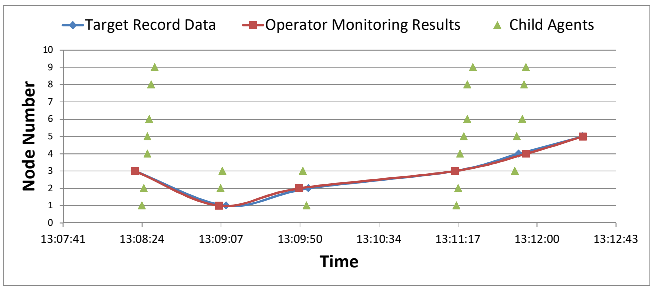

Figure 9. The Results Of Child Agents Moving

blocks are walls or obstacles, such as bookshelves or desks. The white blocks are aisles. The width and height of these blocks are about 2m. The horizontal length is about 30m, and the vertical length is about 25m. The green blocks are sensor detection areas. The numbers in the green circles are nodes (Raspberry Pi). Figure 7 shows an example of installation of a Raspberry Pi. The Raspberry Pi units are set on walls, bookshelves or desks. We set nine Raspberry Pi units on the floor and we set one router in the center of the floor. Table 2 shows the neighbor relation of nodes. For example, node number 1 has node neighbor relations with nodes number 2 and 3. Each Raspberry Pi unit is set more than 4 meters apart from other Raspberry Pi units.

B. Methods

The target person walks on the floor with a Beacon Tag. When the person passes the nodes, the person records the time and the node number. At the same time, the operator records the time and the node, watching the monitoring PC by using a Web browser. To compare both results, if the time and the position are close, the tracking by agents is correct.

In order to evaluate the correctness of tracking by agents, we conducted two kinds of experiments as follows.

1) First Experiment

The person walks slowly and stops for about 10 seconds to wait for the node to detect him.

2) Second Experiment

The person walks at normal pace.

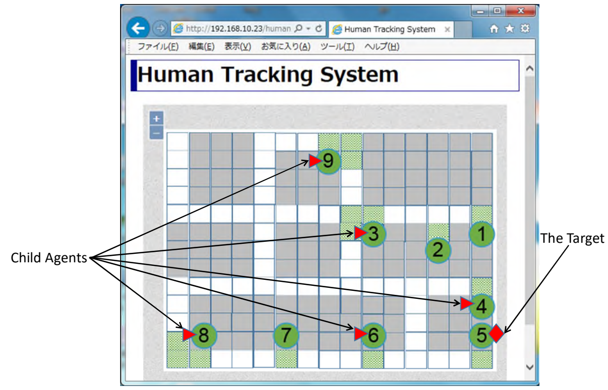

Figure 10. The Monitoring PC Web Browser at Node 5th

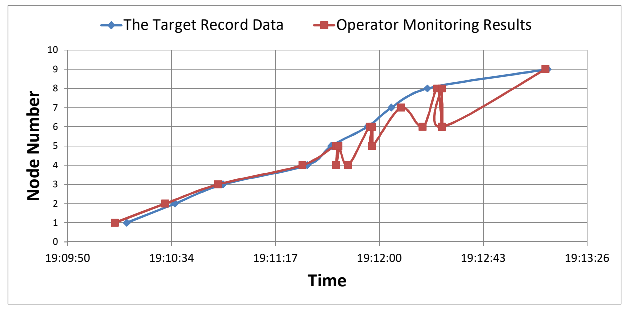

Figure 11. The Second Experiment Result

C. Results

1) First Experiment Result

Figure 8 shows the result of the first experiment. The target person moved to the nodes as follows.

1→2→3→4→5→6→7→8→9→8→7→6→7→6→5→4 →3→2→1→2→3

The first experiment shows that the agents were able to track the person correctly.

Moreover, we confirmed that the algorithm showed in Section II works correctly. Figure 9 shows the results of the sent child agents. The target person moved to the nodes as follows.

3→1→2→3→4→5

The triangles in Figure 9 are the child agents. The child agents that line up vertically and correspond to the same time are the agents that are sent from the same parent agent previously. It can be seen that the human tracking worked correctly because the target moved to the nodes where the child agents were.

Figure 10 shows an example of the monitoring PC Web browser. The target was node 5. The rectangle at node 5 is the target person and the triangles at nodes 3, 4, 6, 8, 9 are the child agents.

2) Second Experiment Result

Figure 11 shows the result of the second experiment. The target person passed to nodes as follows.

1→2→3→4→5→6→7→8→9

The second experiment shows that the agents were able to track the person almost correctly. The monitoring result shows that it shifted a little on the way from the actual target positions. As the cause of these results, we consider that it is the timing when the child agent detected the target and made itself the new parent agent. Each agent communicates asynchronously. Therefore, when the new parent agent is invoked, the old parent still existed. As a result, the plural parents existed at the different nodes. For example, when the target was at node 5, the operator's monitoring results show that the target was at node 5 and at node 4. This means that the old parent agent was at node 4. In the next moment, the old parent agent exited asynchronously.

We consider that this is not a serious problem because the agent could track the person correctly in the end.

VI. Conclusion And Future Work

This paper proposes an example of implementation using Ultra-Small PC Raspberry Pi for a human tracking system based on mobile agent technologies. We introduced the system to a real-life floor and evaluated the correctness of the implementation and tracking algorithms.

We have proposed the tracking method in the circumstance where some nodes are down or the sensor misses the tracking person [2]. In this paper, we evaluated the case that the person walks at a normal pace. However, when the person runs around, the sensor may not be able to detect the person. We are going to evaluate these methods and situations in the future.

In case when the nodes are placed very close, where the nodes are placed within 2 meters, the Beacon Tag is detected by these nodes at the same time. We need to manage this situation by changing the strength of the radio of Beacon Tag in the future.

References

[1] T. Yotsumoto, K. Tanigawa, M. Tsuji, K. Takahashi, T. Kawamura, and K. Sugahara, "Automatic Human Tracking System using Localized Neighbor Node Cluculation," Sensors & Transducers, Vol. 194, No. 11, pp. 54-61, 2015.

[2] T. Yotsumoto, M. Shiozuka, K. Takahashi, T. Kawamura, and K. Sugahara, "Hidden neighbor relations to tackle the uncertainness of sensors for anautomatic human tracking," 2017 Second IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT 2017),. Coimbatore, India, pp. 690-696, 2017.

[3] Raspberry Pi, http://www.raspberrypi.org/, September, 2017.

[4] L. Wenxi, C. Antoni, L. Rynson, and M. Dinesh, "Leveraging long-term predictions and onlinelearning in agent-based multiple person tracking," IEEE Transactions on Circuits and Systems for Video Technology, 25.3, pp. 399-410, 2015.

[5] J. Rivera-Rubio, I. Alexiou, and A. A. Bharath, "Appearance-based indoor localization: A comparison of patch desciptor performance," Pattern Recognition Lettrs, Vol. 66, pp. 109-117, 2015.

[6] C. Alejandro, M. Antoni, B. Marc, and V. Jose, "Navigation system for elderly care applications based on wireless sensor networks," Signal Processing Conference (EUSIPCO 2012), Proceedings of the 20th European. IEEE, pp. 210-214, 2012.

[7] K. Komai, M. Fujimoto, Y. Arakawa, H. Suwa, Y. Kashimoto, and K. Yasumoto, "Elderly Prson Monitoring in Day Care Center using Bluettoth Low Energy," 10th International Symposium on Medical Information and Communication Technology (ISMICT 2016), Worcester, MA, USA, pp. 140-144, 2016.

[8] S. Alletto, R. Cucchiara, G. Del Fiore, L. Mainetti, V. Mighali, L. Patrono, and G. Serra, "An Inddor Location-Aware System for an IoT-Based Smart Museum." IEEE Internet of Things Journal, pp. 244-253, 2016.

[9] SIG Bluetooth. Core specication, https://www.bluetooth.com/specifications/bluetooth-core-specification, September, 2017.

[10] J. Beutel, and M. Krasnyanskiy, *Linux bluez howto: Bluetooth proto-col stack for linux*, http://www.tik.ee.ethz.ch/~jbeutel/pub/bluezhowto.pdf, September, 2017.

[11] I. Fette, and A. Melnikov, *The websocket protocol*, https://tools.ietf.org/html/rfc6455, September, 2017.

[12] OpenLayers, https://openlayers.org/, September, 2017.