3D Radio Tomographic Imaging with Low-Power Wireless Signals

Marvin Herrmann
University Duisburg Essen
Essen, Germany
marvin.herrmann@stud.uni-due.de

Caius-Christian Kurth
University Duisburg Essen
Essen, Germany
caius-christian.kurth@stud.uni-due.de

Abstract

Our work utilizes the commonly known 2D Radio Tomographic Imaging (RTI) technique named "Marauder's Map" on three distinct layers in order to figure out the possibility to detect static objects or a moving person on multiple dimensions in an indoor environment. We studied if this approach is capable of being used for this intention and if the information of several layers can be combined to gain reliable results. Our results show that it is partially possible to combine this information to detect objects or persons, whereby detecting a moving person depends on the grids granularity. A fairly small amount of ten devices is suitable to detect non moving persons on two of three layers. The observations on the top layer highly depends on the persons height, whereas the bottom layer is too inaccurate to detect a non moving person. The middle layer is the most suitable for this purpose.

I. Introduction

The field of Wireless Sensor Networks (WSN) also known as the Internet of Things (IoT) is a diverse field that gained more importance in recent years, especially regarding the topic of indoor localization. A variety of WSN development areas may utilize technologies based on indoor localization and benefit in many ways from it. Radio Tomography Imaging (RTI) is an indoor localization technique that allows locating objects in a certain area without the need to carry a device. Many applications for ambient-assisted living can use the information retrieved by indoor localization to check if people are in need of help, so that caregivers can react in time. Another future application of Radio Tomographic Imaging (RTI) may help police and fire brigades to detect and track offenders or victims through building walls without carrying a device [8]. By detecting the locations of people within a building during emergency situations, RTI can help emergency responders to identify people in need and therefore clarify where to focus attention. Another purpose may be the automatic monitoring and/or controlling of the home environment, called smart home or smart living systems, and buildings in general.

Some building control systems already detect motion in rooms and use this information to control lighting, heating, air conditioning and even noise cancellation [5]. RTI systems could further determine how many people are in a specific room and where they are located, providing even more precise control to the superior system. Applications in this field have high requirements regarding the accuracy of the localization data collected as well as the protection of privacy of the monitored people. To protect privacy a device-free-localization technique (DFL) was developed, which doesn't require a person to carry a device to be monitored.

Instead multiple devices are interconnected with each other, thereby building an area where the object positions can be identified and captured. Every device sends data packets continuously to all adjacent devices in the network and calculates the received signal strength (RSSI) value of packets in decibel-milliwatts (dBm). Objects or human beings that cross the given area have an impact on the communication links (radio links) between those devices and influence the RSSI values. This creates the possibility of determining the location of the objects via transmission signal strengths. In our research, we used the technique known as the Marauder's Map as the base concept and applied the 2D Radio Tomographic Imaging (RTI) technique on three distinct layers. The intention was to find out if it is possible to combine these layers to measure the impact of objects in more dimensions.

II. Related work and prerequisites

For the aforementioned use cases of indoor localization there are already products available on the information technology market using for instance a phased array of radars transmitting ultra-wideband pulses and measuring echoes to estimate ranges and positions of objects. The devices used are precise in short distances but have problems regarding higher distances and increased bandwidths, due to their characteristic of using ultra-wideband pulses [7]. There were attempts to enhance the systems coverage using these devices [6]. Nevertheless, they are too expensive and complex for

today's use cases, also the technique is not based on simple packets transmitted over device radios that use low-power frequencies and are cheap.

Decreasing costs to buy radio frequency integrated circuits (RFICs) that are implemented in today's devices, as well as improvements to WSN communication mechanism, made it possible to deploy an increasing number of devices in our work environments and daily lives. These aspects lead to the circumstance that RTI techniques are more likely to be deployed. Using 28 nodes for their study, Wilson and Patwari [7] figured out that RTI is capable of imaging the Radio Frequency (RF) attenuation influenced by objects or humans in WSN with high density, thus using cheap and commonly available devices.

Bocca, Kaltiokallio and Patwari [1] proved in their work that an object or a person's position can be captured relatively precise in a single layer environment. A layer is a collection of interconnected devices and placed on the same height. They used RTI to produce real-time images of the monitored experimentation environment in which persons were moving via transmission signal strength (RSSI value) variations. The positions of individual persons in the monitored environment were derived from the estimated RTI images that are based on these variations. Their study was conducted over a long period of time within an apartment with one bedroom. In order to cope with a dynamic environment characterized by transmission signal strength variations through moving persons, they came up with a procedure to calibrate the RTI system automatically to prevent shadow effects caused by old RSSI values and by changing a person's position. Their results showed that the developed system only suffers from very few localization deficits. That means the system can adapt well to changes in an indoor environment and the accuracy of localizing a person over longer lasting periods of time is suitable to be utilized for indoor localization [1].

III. System Design

A. Requirements

In order to implement the technique defining suitable requirements beforehand is necessary. This includes requirements regarding the communication between the involved nodes, the collecting of transmission signal strength values and further processing of collected data.

All motes in one distinct layer continuously exchange packets so that the RSSI values of each individual radio link can be estimated. The total amount of radio links on a layer form a grid, which depends on the number of nodes involved and their placement in the real world environment (see Figure 2). The RSSI values are stored in each node and further forwarded to a designated root node, which functions as a collecting instance. Besides collecting RSSI values the root node initializes the process of exchanging packets to achieve a controllable time scheduling (see Figure 1). This time schedule involves a scalable amount of rounds. Each round has a nearly equal time interval, which is further divided into three sub-rounds that also have a nearly equal time interval. A round with its sub-rounds is visualized in Figure 3. In each sub-round the nodes use a different channel to exchange data and therefore collect RSSI values on different channels via switching the channel after receiving the root node's initialization packet. After each round is completed the root node prints the rounds values sequentially to open the possibility of processing them in another application, specifically a mathematical python script that evaluates the collected data and estimates RTI images.

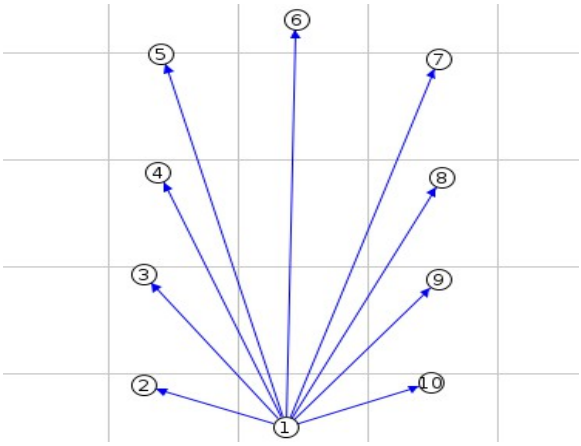

Figure 1: The root node (ID 1) initializes the process of exchanging data.

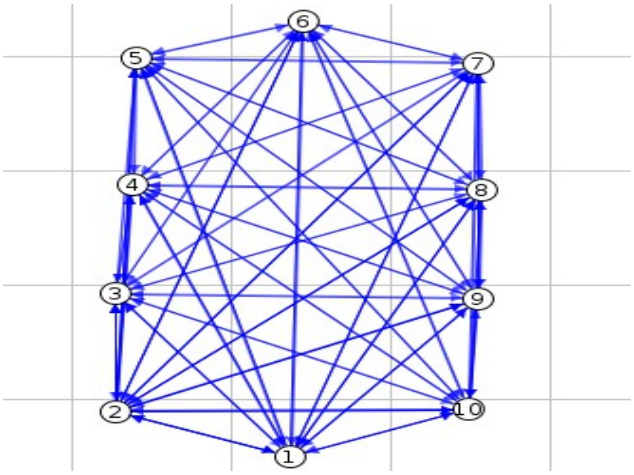

Figure 2: The exchanging of packets between nodes forms a grid.

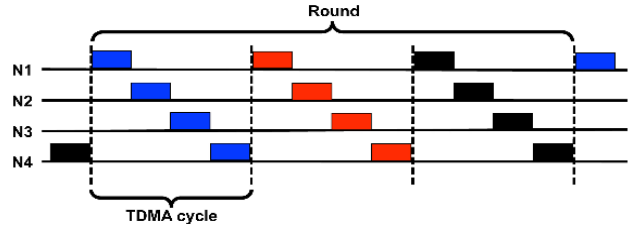

Figure 3: The multi-spin protocol [3] on which our time schedule is based on. A cycle is a sub-round, timeslots and different channels are colored.

The time schedule is based on a MAC primitive called multi-spin [3]. We basically implemented a similar approach but less optimized regarding the timing of transmissions, explained in the following implementation details.

IV. System implementation

A. Parameters

The application includes many parameters implemented as macros in file project-conf.h to finely adjust it and to accomplish the behavior defined in the system requirements. It is possible to change the transmission power being used for transmitting packets (TX_POWER). Furthermore the different channels used in each round for transmitting packets (CHANNEL_A, CHANNEL_B, CHANNEL_C) and their amount (CHANNELS) can be specified. Other channels can be added via inserting for instance CHANNEL_D and increasing the amount of channels. After a defined number of rounds switching through the channels starts, which can be altered via the macro START_SWITCH.

Moreover, the number of nodes can be adjusted to generate different grids (NODE_AMOUNT), an important aspect for system experimentation. The node identifier is hard coded (NODE_ID), since generating it from the automatically generated IPv6 address does not work as intended due to the circumstance that addresses are not always the same.

The moment of time at which nodes start to send packets (START_DELAY) and the time interval over which they exchange packets (ITERATIONS) can be lengthened or shortened. One iteration represents one sub-round and three iterations one round. Also, the UDP port on which packets are received from other nodes can be adjusted (MCAST_SINK_UDP_PORT). For debugging purposes in the WSN simulator "Cooja", defining the macro COOJA with value 1 enables debug printing.

B. Data types

The root node saves its own and the received radio link specific signal strength data (RSSI values) into a two dimensional array. Each row and column is assigned to a specific node id to have easy access to specific radio link data (see Figure 6). For instance accessing array position row three and column four returns the radio link strength between node three and four. The other nodes have one dimensional arrays implemented in which they store the strengths of links to all adjacent nodes. Those arrays are accessed via the same principle as the two dimensional root array (see Figure 7). For instance accessing column four in node with identifier (id) two returns the signal strength of the radio link between node four and two. The one dimensional arrays are nested in structures (structs) consisting of an array and the associated node id, visualized in Figure 4. These structs are send over the radio to adjacent nodes by first creating the struct, secondly assigning the individual node id to it, thirdly copying the one dimensional array into it, and then sending the struct over a UDP connection. These structs are processed at the root node that copies the content (RSSI values) into its two dimensional array at the radio link specific position (Figure 7). The other nodes continuously measure the RSSI values of the packets received and transmit these to all adjacent nodes. These nodes form a multicast group.

```
/* message to multicast
*/
typedef struct msg{
        uint8_t nodeId;
        signed char rssi_collect[NODE_AMOUNT+1];
} msg_t;
```

Figure 4: The radio link individual RSSI values are stored in a struct with the node identifier.

```
/* prepare message: copy collected RSSI values into temporary message struct
*/
msg_t msg_send;
msg_send.nodeId = node_id;
memcpy(&msg_send.rssi_collect, &rssi_collect, sizeof(rssi_collect[0]) * (NODE_AMOUNT+1));
```

Figure 5: The radio link individual RSSI values are copied to the struct that will be sent over a UDP network connection to the multicast group.

C. Time schedule

Time scheduling is the most important aspect of the system, since it ensures transmissions between nodes will not interfere with each other and all signal strength values can be collected, ideally without packet loss. In order to accomplish this behavior, the multicast engine of contiki is changed to SMRF via setting the parameter UIP_MCAST6_CONF_ENGINE to UIP_MCAST6_ENGINE_SMRF. RPL routing has to be disabled, because otherwise packets are forwarded over the fastest route to a sink node. Furthermore, radio duty cycling that comes along with wakeup packets between nodes is disabled by using the nullrdc_driver of contiki. This means nodes do not transmit at least two packets before actually transmitting their packets containing RSSI data and node identifier to other nodes. MAC protocols and their characteristic to control the channel utilization by nodes and send, in our case not desired, acknowledgement packets are not used via selecting the nullmac_driver of contiki. With these approaches, we have maximized control over the communication behavior of the WSN and the number of transmissions between nodes is reduced to a minimum, so that transmissions will not overlap and packets will not need to be re-transmitted because of packet loss, which otherwise occurs due to busy node radios.

All nodes, including the root node, are in the same multicast group with a network address individually assigned to a layer, causing every node that is part of a layer to receive the transmitted packets of every node that is part of the layer. Receiving packets and transmitting packets is handled in separate threads. The root node, with the assigned id 1,

```
/* array of calculated & collected RSSI values (all links between nodes)
*/
signed char rssi_complete[NODE_AMOUNT+1][NODE_AMOUNT+1];
```

Figure 6: The root nodes array containing all radio link signal strength values.

```
rssi_complete[node_id][payload] = rss;          // save RSSI
```

Figure 7: Save a RSSI value into the root nodes array.

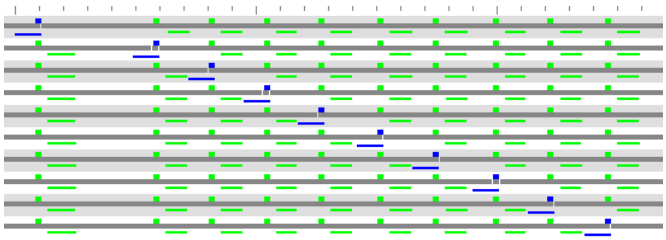

Figure 8: The implemented time schedule of one sub-round which is repeated on a different channel afterwards.

```
/* calculate sending timeslot (17-26ms accuracy) & set timer
*/
s_time = ((CLOCK_SECOND/42)* (my_timeslot-1))+(my_timeslot*NODE_DELAY);
etimer_set(&et, s_time);

/* calculate change channel timeslot & set timer
*/
c_time = ( (CLOCK_SECOND/42) * ((NODE_AMOUNT+2)-my_timeslot) ) +
         ( ((NODE_AMOUNT+1)-my_timeslot) * NODE_DELAY ) +
         s_time;

etimer_set(&et2, c_time);
```

Figure 9: The different calculations of the timeslots to transmit a packet and to change the channel.

initializes each sub-round with a packet sent to such a multicast group network address and calculates its own timeslot after transmitting. The root node's timeslot basically represents the sub-round duration. As soon as the nodes of the multicast group (distinct layer) have processed the initialization packet, they calculate individual timeslots (see Figure 9). When a timeslot is exceeded they transmit their RSSI values collected up to that point to the other nodes assigned to the multicast group. Each node can send packets in time intervals that are about 20ms long.

After the root node's timeslot is exceeded the packets of the next round are sent over one of the different channels defined as parameters (in file project-conf.h). This means all nodes calculate a second timeslot after they receive an initialization packet, visualized in Figure 9. These additional timeslots are necessary to change the channel after all nodes have transmitted and processed the packets of one sub-round. Conclusively, calculating the moment of transmitting a message differs from calculating the moment of changing the transmission channel. Both calculations depend on the amount of nodes used for a layer and the configured node transmission delay (see Figure 9). The timeslots are implemented via etimers (platform independent, event timer) [9] with the side effect of small inaccuracies in the timings. These inaccuracies can be explained by understanding the CLOCK_SECOND macro of contiki. For sky motes such a second has 128 ticks and one tick is about 8ms. Referring to this small calculation, 3 ticks are needed for a timeslot of size 26ms. This is the reason for dividing a second through the value 42. In the actual deployment of the system we observed that the nodes transmit their messages within a 17-25ms timeslot. To increase the accuracy of the time schedule it is possible to change the timer utilized to rtimers (platform dependent, real timer) [9], since they directly are more accurate than etimers, due to calculating the time on hardware basis (direct access to clocks on device). Nevertheless, the mathematical basis of the timeslot calculation is very similar. The calculation of the timeslots is further exemplified in Figure 10. After three sub-rounds, the root

e.g. NODE_DELAY=2

ID2:	3*1 + 2*2	= 7	ID2: 3*10 + 9*2 + 7 = 55
ID3:	3*2 + 3*2	= 12	ID3: 3*9 + 8*2 +12 = 55
…			…
ID10:	3*9 + 10*2	=47	ID10: 3*2 + 1*2 + 47 = 55

Figure 10: Example: transmit and change channel timeslot.

node prints a timestamp of the runtime and the two-dimensional array with all collected radio link RSSI values in node id ascending order.

Due to our implementation one round took about 1,2ms and one sub-round about 400ms. This is not sufficient for requirements that include real time constraints, but feasible for experimentation.

V. Experimentation

A. Setup & data set

Our test environment was a 2x2x1.65(LxWxH) meters cuboid with 3 layers that differ in height.

- Layer 1: Height H=0.10 meter
- Layer 2: Height H=1.00 meter
- Layer 3: Height H=1.65 meter

Every horizontal layer had 10 devices which were adjacent and connected to each other. This setting consisted of 90 radio links for every distinct layer. In Figure 4 the test environment of one layer is visualized.

To gain reasonable data we calculated the RSSI values of each radio link for each horizontal layer on three different radio channels.

- Layer 1: Channels A=14, B=17, C=20
- Layer 2: Channels A=15, B=18, C=21
- Layer 3: Channels A=16, B=19, C=22

This means a total amount of 810 RSSI values were collected each round that need to be evaluated.

Because the root node was plugged into a workstation (laptop) over a USB connection, we had the possibility to store all collected data, previously received from the other nodes and processed by the root node, into a text file.

N. Patwari published a python script on GitHub [4], which we used to evaluate our measured data and visualize it in animated graphics (image sequences). For testing purposes, we used a person who stood in the middle of the test cuboid and moved one step to the front and to the back.

B. Results & Conclusions

In the beginning of our test phase, we used 8 devices for each layer and found out that the number of devices is relevant for the accuracy of the localization of the test object. This is the reason why we decided to increase the number of devices to 10 for each layer which resulted in a higher accuracy.

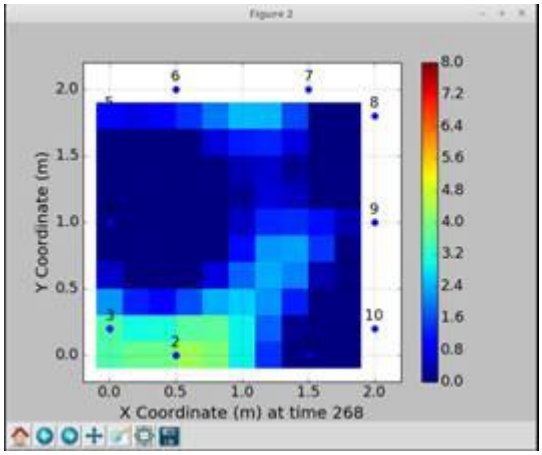

Figure 11: Layer 1 measurement - lowerer part of the body.

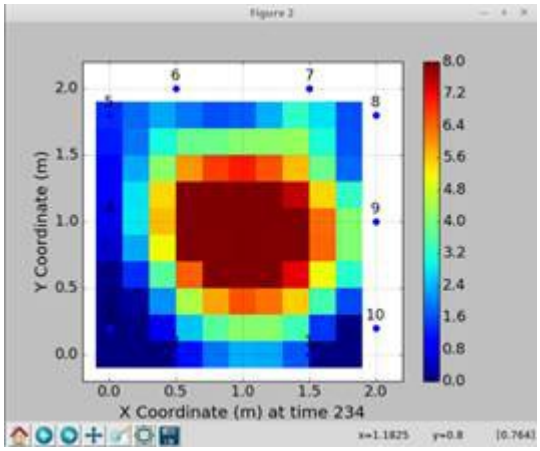

Figure 12: Layer 2 measurement - middle part of the body.

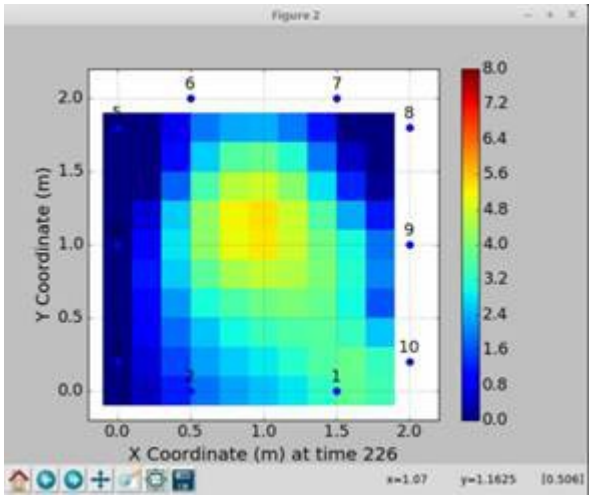

Figure 13: Layer 3 measurement - upper part of the body.

The measurements also showed that various parts of a human body have different impacts on the collected RSSI values.

Figure 11 shows a measurement run of Layer 1 with a height of 0.1m. The links were not significantly affected by the legs or foot of the test person, meaning that the measurement was not very accurate in regard to detecting this part of the body. Contrary, the middle part of a body has more impact on the different radio links signal strengths. This conclusion is based on Figure 12.

The last layer was supposed to measure the upper part of the body of our test person. Because the third layer was placed at a height of 1.65 meter, we discovered that the person's shoulders were influencing the signal strength of transmissions. This explains why we could detect a person on the highest layer (see Figure 13). If the test person is smaller and the shoulders are not captured, the results are similar to the observations on the bottom layer.

We also tried to analyze the movement of the test person inside our test environment and found out, that the length and width of the grid also has a significant impact on the overall measurements and the generated animated graphics. Since our distance between the devices was small, the various positions of the test person could not be determined precisely. In our opinion it is reasonable to conclude, that each layer should consist of a higher number of devices in order to have a finer grid and to better detect movements. This also means that the accuracy of detecting objects on multiple layers depends on the granularity of the grid.

References

[1] M. Bocca, O. Kaltiokallio, and N. Patwari. Radio Tomographic Imaging for Ambient Assisted Living. S. Chessa and S. Knauth (Eds.): EvAAL 2012, Communications in Computer and Information Science (CCIS) 362, pages108-130. Springer, 2013.

[2] O. Kaltiokallio, M. Bocca, and N. Patwari. Enhancing the Accuracy of Radio Tomographic Imaging Using Channel Diversity. 9th IEEE International Conference on Mobile Ad Hoc Sensor Systems (IEEE MASS 2012), October 8-10, 2012, Las Vegas, NV, USA, 2012.

[3] https://sites.google.com/site/boccamaurizio/home/software-data

[4] https://github.com/npatwari/rti

[5] D. Estrin, R. Govindan, and J. Heidemann, “Embedding the Internet,” Comm. ACM, vol. 43, pp. 38-41, May 2000

[6] A.R. Hunt, “Image Formation through Walls Using a Distributed Radar Sensor Network,” Proc. SPIE Conf. Sensors, and Command, Control, Comm., and Intelligence (C3I) Technologies for Homeland Security and Homeland Defense IV, pp. 169-174, May 2005.

[7] Joey Wilson, Neal Patwari “Radio Tomographic Imaging with Wireless Networks,” IEEE Transactions on Mobile Computing, Vol. 9, No. 5, May 2010.

[8] A. Hunt, C. Tillery, and N. Wild, “Through-the-wall surveillance technologies,” Corrections Today, vol. 63, July 2001.

[9] https://github.com/contiki-os/contiki/wiki/Timers