Management Of Networked IoT Wearables – Very Large Scale Demonstration of Cultural Societal Applications
(Grant Agreement No 732350)

D3.1 IoT Enabled Devices and Wearables 1

Date: 2017-12-27

Version 1.0

Published by the MONICA Consortium

Dissemination Level: Public

Co-funded by the European Union's Horizon 2020 Framework Programme for Research and Innovation under Grant Agreement No 732350

Document control page

Document file: MONICA_Deliverable_D3.1_vs1.0
Document version: 1.0
Document owner: DEXELS

Work package: WP3 – IoT Platform, Wearables & Sensors
Task: T3.4 – IoT Enabled Wearables, T3.5 - IoT Enabled Devices
Deliverable type: R

Document status: Approved by the document owner for internal review
Approved for submission to the EC

Document history:

Version	Author(s)	Date	Summary of changes made
0.1	Arjen Schoneveld (DEXELS), Francesco Sottile (ISMB)	2017-10-11	First ToC
0.2	René Laterveer (DEXELS), Arjen Schoneveld (DEXELS), Khaled Sarayeddine (OPTIN), Marco Cipolato (DIGISKY), Orlando Tovar Ordonez (ISMB), Jacopo Foglietti (ISMB), Francesco Sottile (ISMB), Shumei Zhang	2017-11-16	General input
0.3	Nathalie Frey (DEXELS)	2017-11-17	Adjustments in content en layout resulting from telco 2017-11-16
0.4	Nathalie Frey (DEXELS) Khaled Sarayeddine (OPTIN), Marco Cipolato (DIGISKY) Orlando Tovar Ordonez (ISMB), Shumei Zhang (LBU) Claus Blaabjerg (B&K) Ben White (VCA) Arjen Schoneveld (DEXELS) Nathalie Frey (DEXELS)	2017-12-08	Complete further input / finalise layout
0.5	Arjen Schoneveld (DEXELS) Nathalie Frey (DEXELS) Peeter Kool (CNET) Shumei Zhang (LBU) Orlando Tovar Ordonez (ISMB)	2017-12-21	Executive summary, introduction, conclusions Input from CNET, EKF addition, corrections/addition of missing formules section 4.3.5.
0.6	Arjen Schoneveld (DEXELS) Nathalie Frey (DEXELS)	2017-12-27	Adjustments/comments after reviews
1.0	Arjen Schoneveld (DEXELS) Nathalie Frey (DEXELS)	2017-12-27	Final version submitted to the European Commission

Internal review history:

Reviewed by	Date	Summary of comments
Paolo Scalambro /Roberto Gavazzi (TIM)	2017-12-21	Minor comments
Bappaditya Mandal (KU)	2017-12-22	Minor comments

Index:

1. Executive Summary

This report is part of WP3, “IoT Platform, Wearables & Sensors”. The objective of WP3 is to integrate and amend existing IoT enabled devices and wearables, including a suitable architecture and middleware platform. The focus of this report is on the capabilities and integration of both IoT enabled devices and wearables. In the original MONICA DoA this deliverable is supposed to report on the activity in Task 3.4, “IoT Enabled Wearables”. However, we have decided to include Task 3.5, “IoT enabled devices” as well in this deliverable because in our opinion it is an omission in the original DoA. The first part of this deliverable concerns the wearables, the second part of this deliverable discusses the IoT enabled devices.

The most important aspect or capability of the wearables is their locatability. Depending on the type of (radio) technology and hardware features on both the wearable and the RF reader, several ranging techniques exist to determine distances. These include, signal strength (RSSI), Angle of Arrival (AoA) and Time (difference) of Arrival (TdOA). The estimated distances using one of these techniques is used to estimate a location either using lateration or proximity. The former uses distances to calculate an intersection that corresponds with a location, the latter simply uses a proximity measure to determine a very rough location.

There are three different types of wearables:

(1) Crowd wearables, based on sub-GHz radio

(2) Staff wearables, based on UWB radio

(3) Smart glasses

The crowd wearables are lower cost devices compared to the staff wearables. The crowd wearables are suitable for large scale deployment (100,000 s) and allow for rough estimation of locations. The estimated locations are perfectly suitable to get an indication of crowd densities to assist in crowd management functionality. In combination with the built-in two RGB LEDs which can be leveraged for notification purposes, allowing for e.g. visitor guidance to specific exits. The button that is available on the crowd wristband can be used to implement alert/alarm functionality. The crowd wearables use a network of base stations for communication with the wearables and for offloading the data to a central on-site server or gateway. In turn the gateway is connected to the MONICA cloud via the SCRAL layer.

The staff wearables are more feature-rich and expensive wearables. They have much more accurate and real-time location capabilities, i.e. <30 cm accuracy and update rates up to 20 Hz. In addition these wearables have a built-in IMU sensor. Optionally they are equipped with an LED screen, buzzer and Bluetooth (BLE) communication. The LED screen and buzzer can be used to efficiently notify staff members of important information regarding e.g. incidents. The BLE connectivity allows for more advanced (multi-media) messages by leveraging the more advanced display capabilities of a mobile phone. The IMU can be used to detect posture (falling, fighting, etc.) and moving pace (resting, walking, running) of a staff member. Similar to the crowd wristband, the staff wristband relies on special purpose base stations for connectivity.

The smart glasses offer hands-free operation. Important information, alerts and notifications, including rich media messages, can be shown on the display that is directly projected to the eye. Built-in GPS, WiFi and Bluetooth connectivity allow for both standalone operation as well as operation in conjunction with the staff wearables or mobile phones. Also the built-in IMU can be used to track movement and posture of the person wearing the glasses. Note that this type of functionality can be implemented on either a mobile phone, staff wristband or smart glasses or even a combination of the three to improve credibility. For the smart glasses at least a WiFi network is necessary for stand-alone operation. When operation in cooperation with a mobile phone, a functional mobile LTE/3G network is required. When working via WiFi the smart glasses use the SCRAL layer to connect to the MONICA cloud.

The IoT enabled devices include several types of cameras, environmental sensors, microphones and blimps.

There are basically two different types of cameras: CCTV or monocular cameras and time-of-flight cameras. CCTV cameras can be used for people counting, fight detection and generic anomalies. The time-of-flight cameras can be used for more advanced people tracking and queue detection. All the camera’s are using sophisticated local processing nodes to run their algorithms. Instead of raw footage, processed events are communicated to the MONICA cloud.

Microphones are autonomous devices that have GPS built-in and also use local processing to prevent sending large amounts of raw data to the MONICA cloud. The microphones rely on 3G/4G connectivity to offload their data.

The environmental sensors are able to measure wind speed, temperature and humidity. These low-power sensors rely on a IEEE802.15.4 network using a Raspberry Pi as a gateway to upload their data to the MONICA cloud for further data fusion.

Since drones or similar powered UAVs are not allowed to fly over urban areas, an alternative non-powered UAV is chosen: the blimp. Blimps can carry several payloads, like small cameras or microphones. Connectivity to the MONICA cloud will need to be provided by the carried device though.

3. Introduction

In this chapter three different types of wearables are discussed: crowd wristbands, staff wristbands and smart glasses. The crowd wristbands use sub-GHz radio technology and are targeted for tracking the location of a very large number of people at an event. The major purpose of the crowd wristband is crowd monitoring. The staff wristbands use ultra-wide band (UWB) radio for more accurate and real-time tracking of staff members at an event. The third type of wearables are smart glasses. Those are essentially full-fledged Android devices in a glasses form factor. This device is targeted for usage by security staff.

For each of the wearables the technological capabilities and characteristics are considered. Device specific tracking of the wearables is extensively discussed. Infrastructure setup and connectivity to the MONICA cloud are described in detail. In addition, the services enabled by each of the wearable types as defined in the Use Case Groups are enumerated. For the staff wristbands there is and additional section on algorithms for posture analysis based on IMU data. Note that these algorithms are also applicable to mobile phones and smart glasses IMU data. Since the staff wearables have been piloted at the Kappa FuturFestival in July 2017, the results of this pilot are reported as well.

Localisation of the wearables is an important aspect, therefore first some theory of localisation methods will be discussed in the following section.

3.1. Overview of localisation systems

3.1.1. Introduction

Network-based positioning systems operate in two steps: ranging and positioning. Ranging performs the estimations of distance between two nodes of interest as ranging measurements, while positioning uses the measurements obtained from ranging to infer the locations of the unknown nodes (wearable devices).

During this section the concepts of both mobile nodes (i.e., wearable device) and anchor nodes will be introduced. In particular, a mobile node is a device, with a radio interface, whose position is unknown and needs to be localised. An anchor node is a device, with the same radio interface as the mobile device, whose position is fixed and well-known.

3.1.2. Ranging methods

Ranging is a process to determine the distance between two positions i.e. it is an estimation of position-related parameters. This section provides a brief description of the most common ranging methods in radio frequency (RF) devices: received signal strength indication (RSSI), time of arrival (ToA), time-difference-of-arrival (TDoA) and angle of arrival (AoA).

3.1.2.1. RSSI

RSSI, is a power indicator of the received RF signal. The ranging estimation based on RSSI, first introduced in (W. Figel et al. 1969), is the most widely used ranging method, since it is easy to implement in low-cost devices. It is available in almost all the wireless communication hardware whenever a data packet is received. However, RSSI measurements have a strong variability due to environmental conditions since they depend on the quality of the wireless signal which is affected by multi-path, attenuation, interference and other factors. This brings about the main drawback of the RSSI based ranging method: low accuracy.

Typically, RSSI measurements, expressed in dBm, are modelled by using the Log-Normal model (S. Rao. 2007). It represents the received signal power P' , in dBm, as a logarithmic function of the exact distances between two wireless devices performing ranging.

$$P' = P_0 - 10\alpha log_{10}(\frac{d}{d_0}) + X_\sigma, \quad (1)$$

where $P_0$is the received power (dBm) at a reference distance d_0, typically one meter, α is the path loss exponent determined by the environment (H. Hashemi. 1993), and X_σ is an additive Gaussian noise $X_\sigma \sim N(0, \sigma_{db}^2)$, where σ_{db}^2 is the variance of the shadowing effects.

Given the parameters α, P_0 and σ_{db}^2, the unbiased ranging measurement estimator from the Log-Normal model is (Patwari et al. 2003):

$$d' = Cd_0 10^{\frac{P_0 - P\prime}{10\alpha}}, \tag{2}$$

where C is the bias factor depending on the channel model parameters α and σ_{db}, it is defined as:

$$C = exp(-0.5(\frac{ln(10)\sigma_{db}}{10\alpha})^2). \tag{3}$$

As reported in (Y. Qi et al. 2003), the RSSI based range estimator accuracy is proportional to the exact distance:

$$\sqrt{var(d' - d)} \geq (\frac{ln(10)\sigma_{db}}{10\alpha})d, \tag{4}$$

From (4), it can be observed that the range estimator accuracy degrades also for higher values of σ_{db} and lower values of α.

3.1.2.2. Time of Arrival (ToA)

The ToA ranging approach, also known as time of flight (ToF), measures the RF signal travel time between transmitter and receiver. Typically, ToA measurements are normally distributed with zero mean and variance ${\sigma^2}_{d_{ref}}$. In particular, the distance between transmitter and receiver can be obtained from the measured signal travel time by multiplying the signal propagation speed, i.e., the speed of light $c = 3 * 10^8 m/s$. Besides, the ToF can be estimated by using two approaches: one-way ranging (OWR) and two-way ranging (TWR). The OWR approach measures the one-way ToF; its procedure is depicted in Figure 1. In particular, device A sends to device B a ranging packet including the sending time stamp t_0. Then, device B receives this packet and registers the receiving time stamp t_1. Finally, the propagation time T is estimated as $T = T_1 - T_0$. In this case, both A and B need to be synchronised with a common clock. Even a small synchronisation error of ten nanoseconds will lead to a ranging error of three meters. Since the OWR method requires accurate time synchronisation between transmitters and receivers, it is usually difficult to be implemented.

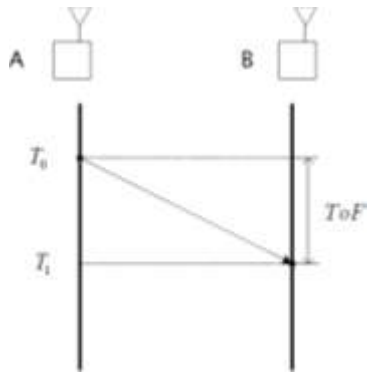


Figure 1: one-way ranging

On the other hand, the TWR approach measures the round-trip time (RTT) of the RF signal between two transceivers. As depicted in Figure 2, device A sends at time T_0 a ranging request to device B, who replies after a replying time (T_R). When the response is received at time T_1, device A is able to determine the RTT as $T_1 - T_0$. Then the ToA is given by $(RTT - T_R)/2$. In this case, the two devices are not required to be

synchronised, since only the clock of device A is used to estimate the ToA. Device B, however, has to send packets back to agent A, which means more traffic is generated in the network.

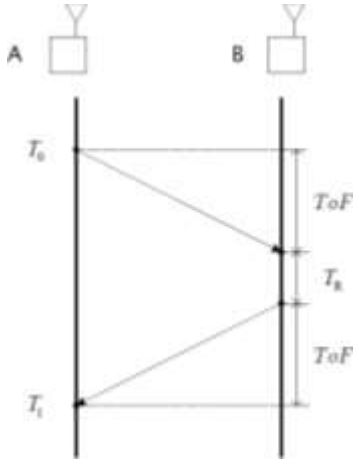


Figure 2: two-way ranging

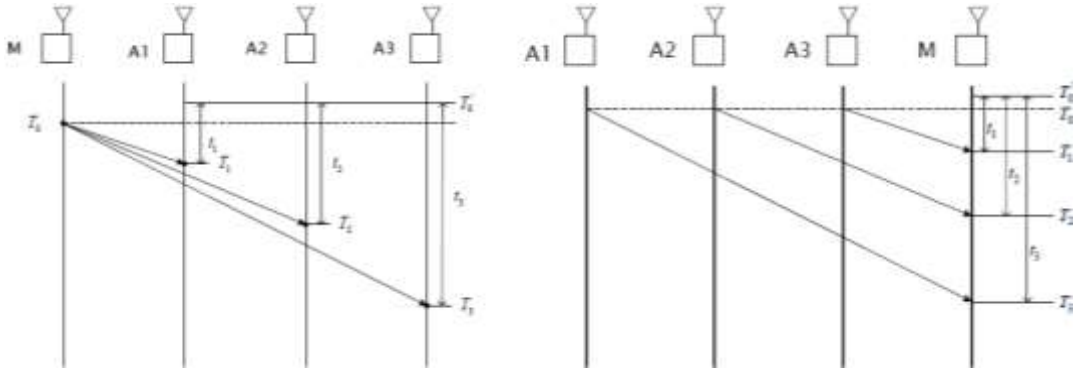
The best achievable accuracy of the ToA-based distance estimate, under single path Additive White Gaussian Noise (AWGN) channel satisfies the following inequality (C.Cook et al. 1970) (H.V.Poor. 1994):

$$\sqrt{var(d' - d)} \geq \frac{c}{2\sqrt{2\pi}\sqrt{SNR\beta}}, \quad (5)$$

where d' is the estimated distance, d is the corresponding exact distance, c is the speed of light, SNR is the signal-to-noise ratio (SNR), and β is the effective bandwidth of the transmitted signal. Hence, the ToA ranging accuracy is improved by increasing the SNR or the effective signal bandwidth. This is the main reason why ultra-wide band (UWB) technology is widely used in time-based ranging methods.

3.1.2.3. Time Difference of Arrival (TDoA)

The TDoA technique is employed when there are both (clock) synchronised and non-synchronised devices in the network. Typically, the synchronised devices are the anchor nodes, since they are more powerful than the wearable devices, and thus, it is possible to synchronise their clocks.



(a) **Mobile ranging** (b) **Anchor ranging**

Figure 3: Time-difference-of-arrival ranging

The ranging procedure is shown in Figure 3. In particular, there are two scenarios with three anchor nodes (A1, A2, A3) and one wearable device (mobile node) (M). In Figure 3(b), each of the anchor nodes sends a ranging message including its common time stamp T_0. When M receives the messages, it measures the receiving times (T_1, T_2, T_3) and calculates the ToFs (t_1, t_2, t_3) based on its own clock t_0. Then, two independent

TDoA measurements are estimated as $t_1 - t_2$ and $t_2 - t_3$. Besides, the clock bias between anchors and M is removed from the TDoA estimation because of the subtraction. Moreover, in Figure 3(a), M sends the ranging message at time t_0 and each anchor node measures the receiving time. When the different receiving times become available, the ToFs and TDoAs can be calculated like before. Finally, the TDoA estimates are mapped into distance differences by multiplying with the speed of light.

This approach minimises network traffic compared to ToA since ranging messages go in one direction, either from anchor nodes to mobile nodes or the opposite way. However, synchronisation of anchor nodes is still needed, and the ranging performance relies greatly on it.

3.1.2.4. AoA

The AoA technique estimates the arrival angle of an incident signal. Usually, the angle is measured by using an array of antennas. The straightforward method is to measure the phase difference of a receiving signal on different antenna elements and then to convert it to the AoA estimate. The position of the mobile node is obtained by the intersection of minimum two straight lines, as depicted in Figure 4.

Compared to other approaches the main disadvantage of the AoA approach is the requirement of larger antennas, which means large size of hardware and high power consumption. Moreover, the accuracy of AoA based estimation degrades when the mobile node moves farther away from the anchor nodes.

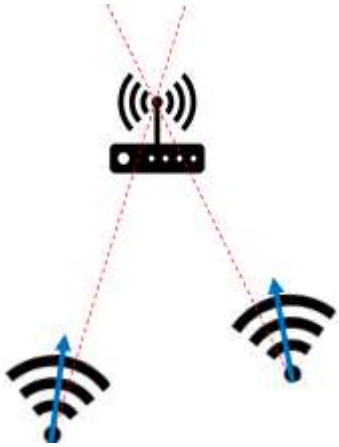


Figure 4: Angle-of-arrival

3.1.3. Localisation algorithms

This section describes some of the most representative positioning techniques. As mentioned before, they are applied after the ranging step is completed and the position-related signal parameters have been collected.

3.1.3.1. Geometric techniques

The geometric approaches exploit geometric relationships between anchor nodes and the wearable device to localise it.

3.1.3.1.1. Lateration

Lateration uses the intersection of lines, curves, circles or spheres to determine the location in 2D or 3D. When dealing with distance measurements from RSSI and ToA, the wearable position is the intersection of circles (2D localisation) or spheres (3D localisation) centred at three or four anchors respectively.

An example of 2D lateration is shown in Figure 5, where the intersection M, depicted with a small red circle, is the position of the mobile node. Moreover, with TDoA measures the location is estimated as the intersection of hyperbolas with foci at the positions of anchors as shown in Figure 6.

It is worth mentioning that there is never a perfect intersection as shown in Figure 5 and Figure 6. Some additional optimisation strategies are taken into account to select the closest solution to the right location.

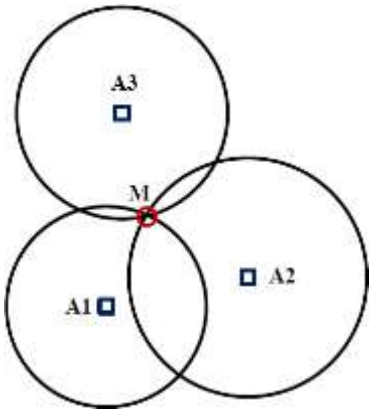


Figure 5: Lateration of three anchors based on RSSI and ToA

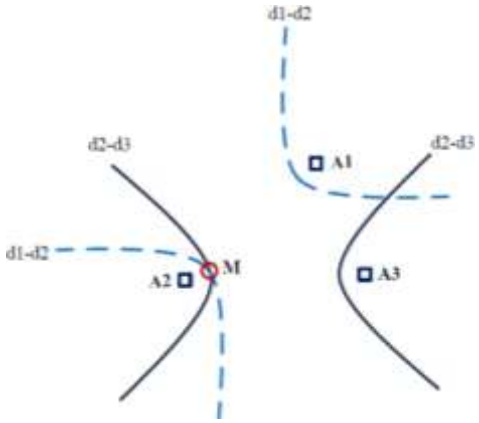


Figure 6: Lateration of three anchors based on TDoA

3.1.3.1.2. Proximity localisation

Proximity localisation can be done also by exploiting the anchors geometry. In this case, instead of using intersecting lines, the intersection of the coverage area is used. Figure 7 shows how to locate a mobile with proximity information. In particular, the mobile's position (M) is considered at the centre of the overlapping area, and its error depends on the size of such area. The positioning error may be large, but it could be better than unavailability.

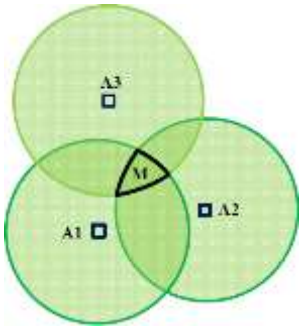


Figure 7: Locate the mobile with proximity information

Although the proximity localisation is not accurate, it can help to locate the wearable devices with few ranging measurements, when there are not enough measurements or these measurements are biased. Figure 8 shows an example of proximity location. In particular, it shows only two available ranging measurements from anchors (A1 and A2) which results in an ambiguity of the estimated position; but using the connectivity of A3 (green circular area), the ambiguity is solved.

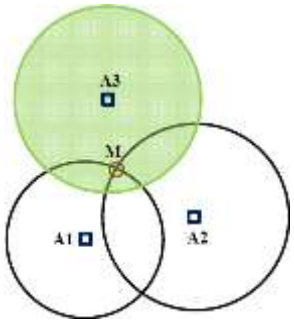


Figure 8: Proximity helps to locate the mobile

3.1.3.2. Bayesian approaches

In the geometrical localisation approaches, statistics of position-related signal parameters are not taken into account. Instead, Bayesian approaches consider the probability and statistics of these signal parameters and the involution of wearable positions. Besides, they model the dynamic positioning problem as a discrete-time stochastic process as:

$$x_k = f_k(x_{k-1}, w_{k-1}), \tag{6}$$

where x_k and x_{k-1} are state vectors at time step k and $k-1$ respectively, and w_{k-1} is the process noise from time step $k-1$ to k, which simulates the effects of mis-modelling and other unpredicted disturbances. $f_k(x)$ is the state transition function defined from time step $k-1$ to k, which can be a linear or nonlinear function.

The relationship between the state and measurement is called observation model, expressed as:

$$z_k = h_k(x_k, v_k), \tag{7}$$

where z_k represents the available measurements or observations at the current time step k and v_k is the measurements noise. $h_k(x)$ is the observation function, at time step k, which relates the state x_k with the observations; it can be linear or nonlinear.

From (6) and (7), it can be seen that Bayesian approaches model the dynamics system problem as a first order Markov chain (Figure 9), whose state x_k is not directly observable but can be inferred from measurements.

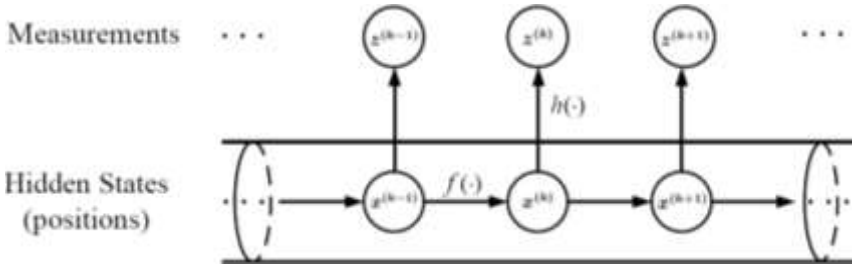


Figure 9: Hidden Markov model for Bayesian tracking

From the Bayesian point of view, the tracking problem is to recursively estimate a new state x_k, taking into account all available measurements up to time step k ($z_{1:k}$). It becomes a problem of the calculation of the marginal distribution $p(x_k|z_k)$, which in principle can be estimated by two stages: prediction and update. In the prediction stage, the a priori probability distribution function (p.d.f.) $p(x_k|z_{1:k-1})$ of the current state x_k is obtained. While, in the update stage, the a posteriori p.d.f. $p(x_k|z_{1:k})$ is obtained.

Extended Kalman Filter: The Extended Kalman Filter (EKF) provides an efficient recursive solution for non-linear discrete filtering problems with low complexity (G. Welch et al. 2006), and it is widely used in positioning and tracking applications. It models a dynamic system using the same equations of (6) and (7), but the process noise (w_k) and the measurement noise (v_k) are supposed to be Gaussian distributed, $w_k \sim N(0, Q_k)$, $v_k \sim N(0, R_k)$. Moreover, the EKF estimates an *a posteriori* state vector by using a feedback control approach. In particular, the current state vector (x_k) is predicted to produce an *a priori* estimate (first step), then it is refined by using the feedback from the measurements (second step). These two steps are also known as *predict phase*, and *update phase*.

The *predict phase* provides an estimate of both *a priori* state $x'_{k|k-1}$ and error covariance matrix $P_{k|k-1}$. These estimates are based on the previous *a posteriori* estimates of both the state x'_{k-1} and the error covariance matrix P_{k-1}, by using the following equations:

$$x'_{k|k-1} = \mathrm{f}(x'_{k-1|k-1}, \mu_k) = \mathrm{F_k} x'_{k-1|k-1} + \mathrm{B_k}\mu_k, \tag{8}$$

$$\mathrm{P_{k|k-1}} = \mathrm{F_k P_{k-1|k-1} F_k^T} + \mathrm{Q_k}, \tag{9}$$

where $\mathrm{F_k} = \frac{\partial f}{\partial x}\Big|_{x'_{k-1|k-1}}$ is the Jacobian matrix of the state transition function $\mathrm{f}(x)$, $\mathrm{B_k}$ relates the input (μ_k) with the dynamics of the system and $\mathrm{Q_k}$ in the process noise covariance matrix.

The *update phase* performs the feedback control, where both the state vector (x_k) and the error covariance matrix (P_k) are updated using the measurements vector ($\mathrm{z_k}$). The optimal Kalman gain K_k and innovation vector y'_k are calculated as follows:

$$\mathrm{K_k} = \mathrm{P_{k|k-1} H_k^T}\left(\mathrm{H_k P_{k|k-1} H_k^T} + \mathrm{R_k}\right)^{-1}, \tag{10}$$

$$y'_{\mathrm{k}} = \mathrm{z_k} - \mathrm{h}(\mathrm{x'_{k|k-1}}), \tag{11}$$

where $\mathrm{H_k} = \frac{\partial h}{\partial x}\Big|_{x'_{k|k-1}}$ is the Jacobian matrix of the observation function $\mathrm{h}(x)$ and $\mathrm{R_k}$ is the observation error covariance matrix. Typically, $\mathrm{R_k}$ is modelled as uncorrelated white Gaussian noises and depends on the variance $\sigma^2_{d_{ref}}$ of the measurements vector $\mathrm{z_k}$. Finally, the state estimate $\mathrm{x'_k}$ and the error covariance matrix $\mathrm{P_k}$ are updated as follows:

$$\mathrm{x'_{k|k}} = \mathrm{x'_{k|k-1}} + \mathrm{K_k y'_k}, \tag{12}$$

$$\mathrm{P_{k|k}} = (\mathrm{I_n} - \mathrm{K_k H_k})\mathrm{P_{k|k-1}}, \tag{13}$$

where $\mathrm{I_n}$ is an identity matrix whose dimension is the same as $\mathrm{P_k}$.

Particle Filter: The Particle filter (PF) is a category of Monte Carlo methods that approximates the discrete *a posteriori* distribution of a generic state vector x_k at time $\mathrm{t_k}$ by employing a set of particles and associated weights $\{\mathrm{x_k^i}, w_k^i\}_{i=1}^N$. The estimated *a posteriori* distribution is given by:

$$\mathrm{p}(\mathrm{x_k}|z_{(1:k)}) \approx \sum_{i=1}^{N} w_k^i \delta(\mathrm{x_k} - \mathrm{x_k^i}), \tag{14}$$

where $z_{(1:k)}$ denotes the observations up to $\mathrm{t_k}$, w_k^i is the weight associated to the i-th particle, N is the total number of particles and $\delta(x)$ is the Dirac delta function, defined as zero everywhere except for $\delta(0) = +\infty$, with $\int \delta(x)dx = 1$.
The sample x_k^i is generated as $x'_{k|k-1}$ in (8) (i.e., it depends on the state model), and its associated weight is given as:

$$w_k^i = w_k^{i-1} * \mathrm{p}(z_k|x_k^i), \tag{15}$$

where $\mathrm{p}(z_k|x_k^i)$ is the likelihood function which is defined as follows:

$$p(z_{m,k}|x_{m,k}^i) = \prod_{n \in M_m,k} p_{nm}(d'_{refnm,k} - \|p'_{refn,k} - p_{m,k}^i\|), \tag{16}$$

where p_{nm} is the probability density of measurements, $p'_{refn,k}$ is the reference anchor position from which $d'_{refnm,k}$ has been collected, and $p_{m,k}^i$ is the wearable device position for the particle i.

4. IoT enabled wearables

4.1. Introduction

In this section the smart wearables provided by Dexels will be discussed. There are two different types of wearables: a *crowd wristband* and a *staff wristband*. The crowd wristband is a less expensive wristband that allows for supporting 100,000's of visitors. The staff wristband has more features and is aimed at 1,000's of users. The crowd wristband allows for crowd monitoring, leveraging the bi-directional 100 m range radio that is integrated in the wristband. The crowd monitoring feature can be used to create heat maps of the crowd, showing visitor densities. In addition RFID is integrated in the crowd wristband to support access control and cashless payments. The LEDs on the crowd wristband can be used for entertainment purposes and for crowd control as well. The staff wristband has a more accurate tracking capability. The typical accuracy of crowd wristbands is 10 m while the staff wristband has an accuracy of less than 50 cm. In addition the staff wristband has an integrated BLE radio that can be used to communicate with a smartphone or smart glasses. The wristband version of the ultra-wide band (UWB) wearable also features a LED screen that can be used to notify or instruct the user.

4.2. Crowd wristbands

4.2.1. Previous large scale deployments

The first large scale implementation of the crowd wristband was deployed in 2014 during two weekends of the Tomorrowland festival in Boom, Belgium. Each weekend a total number of 125,000 crowd wristbands were active during three days. The system supported "scanning" all 125,000 wristbands in 8 minutes. A wristband communication time slot was used to send a message containing: "closest base station", voltage, temperature and a unique wristband ID. A total number of 60 base stations, spread over the festival area, connected using ethernet cables and powered by Power over Ethernet (PoE) was deployed. Each base station was running a Salt stack[1] for supporting remote configuration.

Figure 10: LED show with crowd wristbands in front of the main stage at Tomorrowland 2014

1 https://saltstack.com/

4.2.2. Technology overview

The crowd wristband can appear in multiple bracelet incarnations. It could be a leather bracelet, a textile wristband or a silicon wearable. What really matters is in the inside. The current version of the wristband contains the following components:

- Radio/microcontroller (MCU)
- CR2032 battery
- Two bright RGB LEDs
- RFID/NFC chip
- Button
- Clock
- Antennas (HF RFID and UHF)

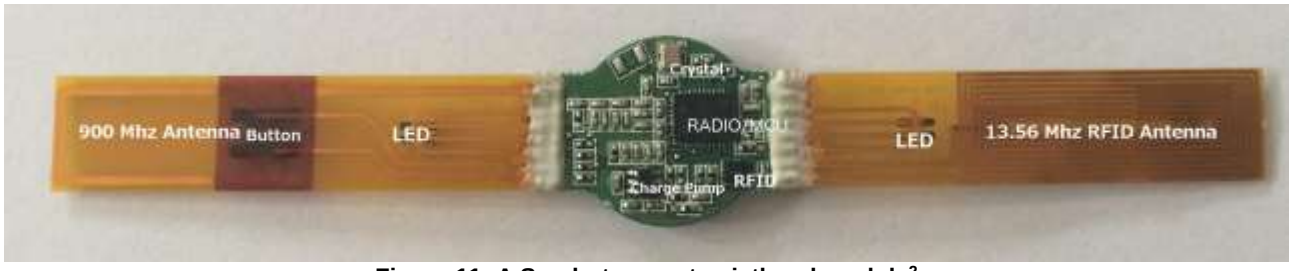


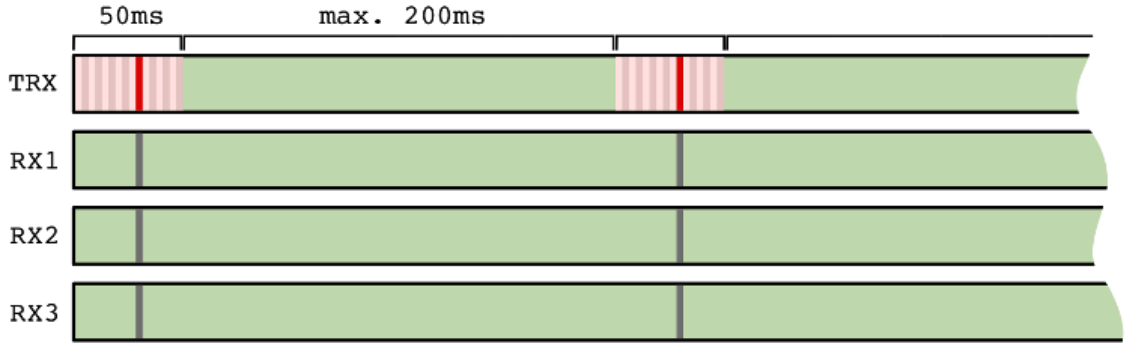
Figure 11: A Sendrato smart wristband module[2]

Together these parts cooperate to form the crowd wristband solution. The software running on the MCU controls the operation of the wristband. It controls the radio communication, the LEDs and the behaviour of the button press.

The MCU can wake up from its deep sleep mode in several ways. On wakeup the MCU starts its normal operation by listening to radio messages transmitted from a base station. The base station messages synchronises the wristband clocks and sends commands either to a particular wristband or to all wristbands. The commands instruct the MCU to light up the LEDs. Each wristband can be addressed separately by means of a unique ID. This unique ID is associated with personal details of the visitor wearing the wristband.

The button can be used for several user inputs. Two examples are “Music Like” and “Friend-Connect”. The Music Like works as follows: when the button on the wristband is pushed for a moment, the location and time will be registered in that user’s data file. This is then correlated with the music played. We detect this music with a real time music discovery method (continuous and automated). This information together can be used

2 Note that the battery, LEDs and button are positioned on the flip side.



to send a “Playlist” to the user in an app, on a website or via email. The music tags can also be used in more advanced communication to the visitor, like sending YouTube videos or music fragments.

Figure 12: Crowd wristbands TDMA protocol

A friend is connected by exchanging personal information (which is enabled during registration) by holding the button for 2 s until the LEDs light up blue. By holding the wristbands in close proximity of each other, the LEDs flash green and a connection is made (in the cloud database).

4.2.3. Crowd wristband TDMA Protocol

The crowd wristband protocol uses parallel time slots to allow for higher throughput of wristband messages. Four radio channels (TRX, RX1, RX2, RX3) are used simultaneously. Wristbands messages are sent using a TDMA protocol. A CSMA phase in the TDMA scheme is used for sending “urgent” messages like “wristband connects” and button pushes (aka likes).

The wristband protocol relies on tight clock-synchronisation to support the TDMA protocol. The clocks on the wristband are synchronised by pilot messages sent by each base station. Depending on a unique ID (with a cardinality of 2^{16}), a base station sends its pilot message in a predefined time slot. The length of a time slot, for both pilots and wristband messages, is set to 3 ms. A maximum number of 16 base stations time slots initiate a new “communication window” (see Figure 12). Hence, the pilot-phase of the messaging windows always takes 50 ms. A wristband uses the pilot message to synchronise its local clock. Besides clock synchronisation, a pilot message contains the “wristband ID range” parameters and optional LED commands. The “wristband ID” range is used to define the logical range of wristband IDs (WUIDs) that need to be polled. The range is defined by a start- and an end-value. The WUID of a wristband is first masked to fall within the polling range before its reporting time slot/channel is determined.

The remainder of the messaging window is used to send a wristband “reporting message”. Each wristband is assigned its unique time-slot and channel (1 out of 4) to send this message. The reporting window size can be adjusted from 0 ms to 200 ms, resulting in a total communication window of 50 - 250 ms. The 50 ms mode is used for low-latency LED operation, allowing a new LED command to be sent every 50 ms. This comes at the expense of not being able to report wristband messages during 50 ms-operation. The default operation mode is 250 ms, resulting in a maximum reporting throughput of almost 3,200 wristbands per second (remember the 4-fold parallelism due to the multiple channels).

The addition of an urgent message mode for “wristband connect” and “button push” messages using a CSMA protocol results in a responsive and correct implementation of the so called “Friend-Connect” feature. The wristband connect feature is enabled by holding the buttons on each wristband for 2 s. This will trigger the wristband to send, at lowest TX power, its unique WUID and start listening for a maximum of 10 s to WUIDs sent by neighbouring wristbands. Whenever a neighbouring WUID is received the “wristband connect” message is constructed, containing both WUIDs, a nearest base station ID and a time stamp. This message is sent “immediately”. The wristband will wait for an ACK of the base station ID that corresponds to the nearest base station ID defined in the wristband connect message. When the ACK is not received after 1 s, this process is retried for a maximum number of 10 times. If the ACK is still not received after 10 retries, the wristband

connect message will be sent at the first occurrence of a reporting time slot for this specific wristband; this is done only once. The same procedure is used for single button push events.

The Sendrato protocol supports a maximum of 16 base stations per “Pilot Channel” (TRX). A total number of 6 Pilot Channels can be defined, resulting in a maximum infrastructure of 96 base stations. A wristband determines the “strongest” Pilot Channel by scanning other channels every 30 s. By keeping a list of strongest channels it decides whether it is time to switch to another Pilot Channel. This effectively implements a channel handover procedure for the wristbands allowing for larger areas that can be supported by the protocol. This still limits the maximum area size that can be covered by the wristbands; exploring ways to bypass this limitation is part of future work.

4.2.4. Infrastructure for crowd wristbands

The crowd wristbands need a dedicated infrastructure of base stations that communicate with each other and with the wristbands. The maximum safe range between a wristband and a base station is 75 m. This implies that a wristband must always be at maximum 75 m away from a base station in order to have coverage. This characteristic can be used to design and setup the base station infrastructure for a specific venue. Since there is a limit to the number of base stations (96) there is a limit to the maximum area that can be covered. Hence, currently the spatial scalability is limited. The number of wristbands that is currently supported is limited by the 3 bytes that are used to identify a wristband. There is no inherent limitation to the number of wristbands in the protocol itself.

The base station radio is controlled by an ARM based PC board running Linux and the base station software. The base stations themselves are joined together in a software cluster. A unique redundant-communication protocol has been developed that enables the use of multiple physical communication layers between the base stations.

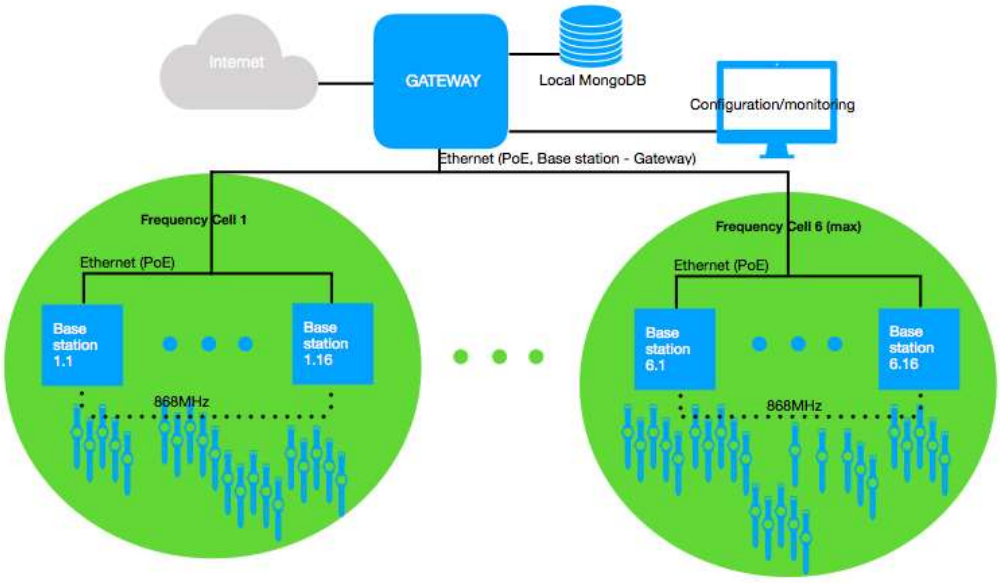


Figure 13: Crowd wristbands infrastructure deployment

TCP/IP based communication, both Ethernet and WiFi, as well as several low-bandwidth wireless communication technologies (Plexus) are supported. Altogether this creates a highly fault-tolerant communication channel between the base stations. If for example the Ethernet or WiFi infrastructure fails, the messages are still sent using the alternative available wireless infrastructure, making the system independent of the festival’s infrastructure. A typical communication use case is a message that originates from a wristband, being received by one or more base stations and further transported to our server node(s).

The server infrastructure is partly deployed locally on the festival premises and partly in the MONICA cloud. This setup enables the mobile apps that are running on the visitor’s smartphones to interact with the system. Again, for fault-tolerance reasons, the server infrastructure can be set up redundantly. Failure of a server node does not result in failure of the entire system. A management console is available for operators to control the entire system. Furthermore, the software on each base station can be updated simultaneously with a single mouse click in a matter of seconds. The messages that are received from the wristband are used to perform

real-time triangulation to drive the crowd monitoring system, heat map visualisation and individual wristband tracking.

Over the course of an event millions of messages are being collected. These messages are stored in a highly scalable distributed MongoDB database. The database contains information of all the visitors. It is completely up to the festival organiser to determine which kind of registration information is mandatory or optional. This data can be imported from the ticketing database or, depending on the capabilities of the ticketing database, automatically synchronised. The visitor database is replicated in real time to cloud instances.

4.2.5. Services enabled by crowd wristbands

4.2.5.1. Localisation service

Since there is an approximate location for every wristband in near real time, every few minutes, this feature can be leveraged to implement a Location Service for visitors. In case a visitor needs to be found the last known location can be queried from the COP in the MONICA cloud.

4.2.5.2. Crowd density detection

The location of the visitors can be used to calculate a crowd density. The resolution of this discrete density field is typically 5 m x 5 m. This is a useful feature for so called crow monitoring, i.e. knowing the number of visitors in various event areas at any instant. This could also be used to detect high-risk queues (or at least high risk densities) based on the maximum capacity of these areas. The location collected by the crowd wristbands is used to create a current overview of the crowd distribution in the event area; the DSS can implement algorithms to detect over capacity or high risk queues. The crowd density can be visualised in a dashboard application running in the Central Command Center (CCC) of the event. In addition to the DSS, this information can be used by CCC staff to detect hot spots in crowd densities.

4.2.5.3. Find the exit

The two RGB LEDs on the crowd wristband can be used to guide people based on colour codes. A venue can have “coloured exits”. The LEDs of the wristbands can be controlled by individual base stations. A wristband will give priority to LEDs command from the base station that is nearest; based on this proximity characteristic the wristbands can be guided to the nearest exists indicated by a colour command of the LEDs.

4.2.6. Localisation algorithms for crowd wristbands

In this section several localisation methods are discussed that are applicable to the crowd wristband.

4.2.6.1. Multi-lateration NLLSQ

The received signal strengths (RSSI) give a rough estimate of the distance of a wristband to the base stations. From these distances the location of the wristband can be determined. This is called lateration (see 3.1.3.1.1.). The nonlinear least squares method minimises the difference between the localised distances and the measured ones. The localised distances are the distances between the localised wristband position and the base stations. The Levenberg Marquardt[3] method is used to minimise the least squares which is an iterative algorithm. At each step the localised distances and Jacobian are calculated. Usually the algorithm converges in a few steps.

4.2.6.2. Particle Filter

The Particle Filter (PF) is a category of Monte Carlo methods that approximates the discrete a posteriori distribution of a generic state vector x_k at time t_k by employing a set of particles and associated weights $\{x_k^i, w_k^i\}_{i=1}^N$. The estimated a posteriori distribution is given by:

$$p(x_k|z_{(1:k)}) \approx \sum_{i=1}^{N} w_k^i \delta(x_k - x_k^i), \quad (8)$$

3 https://en.wikipedia.org/wiki/Levenberg%E2%80%93Marquardt_algorithm

where $z_{(1:k)}$ denotes the observations up to $\mathrm{t_k}$, w_k^i is the weight associated to the i-th particle, N is the total number of particles and $\delta(x)$ is the Dirac delta function, defined as zero everywhere except for $\delta(0) = +\infty$, with $\int \delta(x)dx = 1$. Compared to Kalman filters, PFs are able to deal with nonlinearity of the system and non-Gaussian distributed observations (F. Sottile et al. 2011).

The weights are recursively updated by using the importance sampling principle (A. Doucet. 1998). In particular, the performance of the PF is strongly influenced by the choice of the importance density $q\left(x_{\mathrm{k}}|x_{0:\mathrm{k}-1}, z_{(1:k)}\right)$ (M.S et al. 2002).

In many practical tracking problems, the importance density is chosen to be the a priori density $p\left(x_k|x_{k-1}^i\right)$. Thus, depending on the state model, a sample x_k^i is generated as $\hat{x}_{k|k-1}$ in (Caceres et al. 2009), and the associated weight is given by:

$$w_k^i = w_k^{i-1} * \mathrm{p}\left(z_k|x_k^i\right), \tag{9}$$

where $\mathrm{p}\left(z_k|x_k^i\right)$ is the likelihood function.

According to the simulator that will be presented in section 4.3.6.1 as well as the RSSI-measurements distribution that will be provided in section 4.3.4.6, simulation results showed an RMS error of 2.32 m. Moreover, as benchmarking, the simulation results has been compared with a linear least squares (LSS) algorithm (S. Gezici et al. 2008) which has produced a RMS error of 8.5 m.

Likelihood Function for RSSI Measurements

The likelihood function for RSSI is defined as follows:

$$p\left(z_{m,k}|x_{m,k}^i\right) = \prod_{n\in M_m,k} p_{\mathrm{X}_\sigma}\left(\hat{P}_{refnm,k} - P_{refnm,k}\left(d\left(p_{refn,k}, p_{m,k}^i\right)\right)\right), \tag{10}$$

where p_{X_σ} is the probability density of the RSSI measurement, which is Gaussian distributed with zero mean and standard deviation σ_{db}, and $P_{refnm,k}\left(d\left(p_{refn,k}, p_{m,k}^i\right)\right)$ is the RSSI value from (1), which depends on the Euclidean distance between the mobile node position ($p_{m,k}^i$) particle i and the position ($p_{refn,k}$) of the neighbouring peer at time t_k.

4.2.7. RSSI measurements campaign and GDOP analysis

A RSSI measurements campaign has not been foreseen during the Kappa FuturFestival (KFF) described in section 4.3.9. Consequently, real measurements were not available to perform a reliable Cramér–Rao lower bound (CRLB) analysis. Despite this, a preliminary CRLB analysis, based on (N. Patwari et al. 2003), has been done considering the input channel model parameters (η =2.31,σ_{dm} =2.292) from (Chruszczyk et al. 2016) (Chruszczyk. 2017) with the same deployment used in KFF as geometric reference. It is worth mentioning that an analysis based on real measurements will be provided in the next iteration of this document.

Figure 14, shows the lower bound localisation error considering the already mentioned parameters. The average localisation error is 2.93 m with a minimum of 1.92 m and a maximum of 4.63 m, respectively; and a standard deviation of 0.45 m. The localisation has a lower error when the mobile node is close to the anchor nodes, since the RSSI measurement is more reliable. Also, when the mobile node is in the middle of the localisation area since there is good geometry with respect to the anchor nodes.

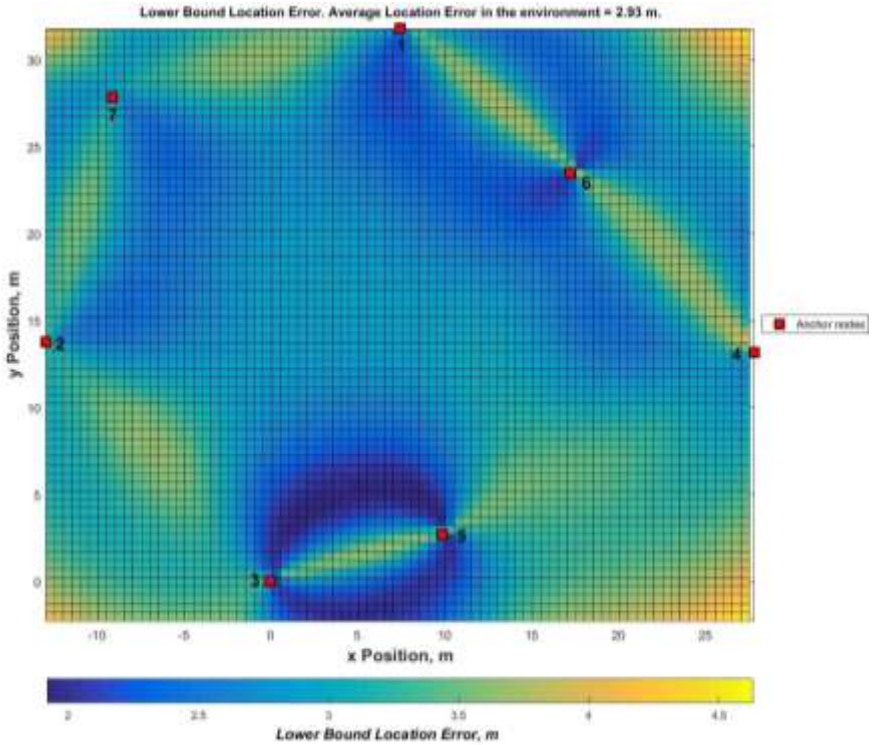


Figure 14: RSSI measurements CRLB analysis

4.2.8. RSSI-based simulator

The RSSI-based simulator is MATLAB-based software which simulates the behaviour and evaluates the performance of the algorithm presented in section 4.2.6, used to localise and track position of some mobile nodes. In particular, the software provides as output the cumulative distribution function (c.d.f.), the p.d.f., the convergence time (CT) of the algorithm, and plots the estimated positions. Moreover, it allows to simulate an indoor environment, without obstacles, with static nodes or pedestrian mobility nodes. Besides, it also simulates urban environments. Even if the software has a centralised CPU, mobile nodes position estimations are performed as the algorithm would separately run on each mobile node (i.e. distributed localisation).

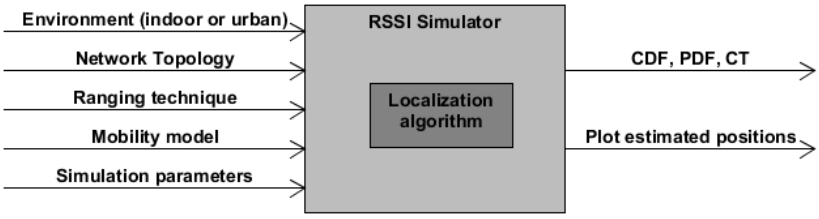


Figure 15: RSSI-based simulator

RSSI measurements are modelled as described in section 3.1.2. Pedestrian mobility is modelled with a random way point model or with a Brownian model. The random way point model generates mobile nodes which are moving at a certain minimum/maximum speed each time while in the Brownian model mobile nodes move adopting different standard deviations with respect to the direction (i.e., East-West, North-South and up).

4.2.9. Integration of crowd wristbands into the IoT middleware

The integration of the crowd wristbands is realised via the Gateway Node of the crowd wristband (Figure 16). The gateway collects all the information coming from the various base stations that in turn receive all the wristbands messages. The gateway is interfacing with the SCRAL layer in the MONICA cloud by pushing JSON messages to an MQTT end-point offered by the SCRAL. The following MQTT endpoints are supported:

Table 1: REST API

API Name	Description

REST/MQTT: /monica/wristband	Gets the current status of a wristband based on a unique wristband ID. Current status contains wristband ID: • lat/lon coordinates • time stamp • velocity (only UWB) • bearing (only UWB) • closest area • x,y position (only UWB) • temperature • battery level The data format is JSON
REST/MQTT:/monica/buttonpress	• wristband ID • time stamp The data format is JSON
REST:/monica/wristbands	Get a list of all active wristband IDs: • wristband ID The data format is JSON

4.3. Staff wristband

4.3.1. Introduction

The fundamental building blocks of the staff wristband system are ultra-wide band (UWB) anchors and tags. Anchors are fixed location UWB nodes, containing at least one so called master anchor that is responsible for collecting all the data (wirelessly or wired) from the other anchors. Anchors send/receive messages to/from mobile tags. These messages are used in the localisation process as well as for communicating so called user payloads. These payloads can include e.g. data from sensors attached to the tags.

The system uses UWB-based geometrical localisation. For ranging TWR ToA (see section 3.1.2.2.) is used. TWR does not require any synchronisation of the clocks at all, however this comes at the expense of having to communicate at least three messages between tag/anchor before the range can be determined. This means that with TWR less tags can be tracked in a certain amount of time compared to T(D)oA. Still, the system uses TWR and supports 1,200 location updates per second. Hence, 1,200 tags can be ranged running at an update rate of 1 Hz.

The next step in the process is localisation. Localisation calculates the position based on the distances (calculated during the ranging phase) between the tag and the (visible) anchors. The position is calculated using lateration. Since the accuracy of the Decawave UWB chip is +/- 10 cm in LOS (Line of Sight) conditions and +/- 30 cm in NLOS (Non Line of Sight) conditions, there is always additive (white) noise present in the calculated distances. Therefore, an exact (closed form) solution of the lateration problem is not possible. One has to rely on an optimisation procedure to calculate the location. Typically a nonlinear least square (NLLS) method is used. In case of not just localisation, but also tracking a moving object, additional methods are used. Jitter in the calculated track is usually mitigated using some smoothing or filtering method. In our case we are using either an Extended Kalman Filter or an Extended Finite Impulse Response (FIR) in combination with an NLOS detection and mitigation. These methods result in low jitter while still having acceptable latencies (<500 ms in case of a 20 Hz update rate).

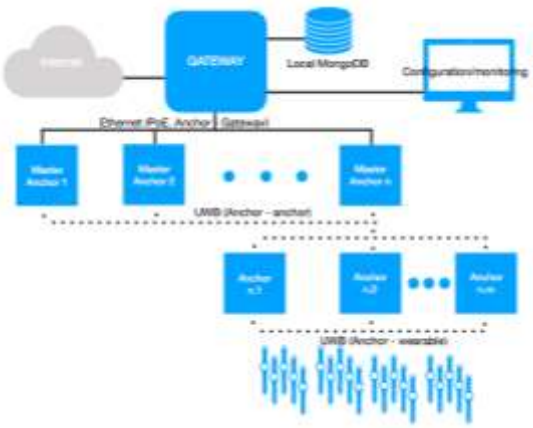


Figure 16: Staff wristbands infrastructure deployment

4.3.2. Technology overview

The UWB module contains a number of components in its current incarnation:

- 1.3” 176x176 color display
- Bluetooth LE
- USB and wireless charging
- Decawave DW1000 ultra-wide band radio
- UWB Low Noise Amplifier
- ARM Cortex M4
- 300 - 400 mAh battery

Furthermore several sensors and actuators are available:

- Light sensor
- IR proximity
- Pressure sensor
- Temperature sensor
- Humidity sensor
- Microphone
- 9 axis IMU (Accelerometer, Gyroscope, Magnetometer)
- 2 buttons
- Haptic feedback

4.3.3. Infrastructure for staff wristbands

The infrastructure consists of a number of components. The total system setup is comprised of anchors, tags, a network router, a network switch and a server running the location engine and configuration software. A browser is used to setup and monitor the system which can run on anything from a PC, a tablet to a smartphone.

The Indoor Positioning System (IPS) or Location Engine (LE) runs on a standard Linux PC. Our platform of choice is currently a System 76 Meerkat[4]. The Meerkat is connected to the same Local Area Network as the master anchor(s). The software running on the Meerkat receives the tag distances from the master anchor via UPD messages. Subsequently the IPS on the Meerkat calculates the position of the tag using a selected localisation algorithm, either NLLS (Non Linear Least Squares), EKF (Extended Kalman Filter) or EFIR (Extended Finite Impulse Response filter). The position of the tag can be visualised on the browser together with a modelled blue print of the environment. In addition, the calculated positions can be “published” using several

4 https://system76.com/desktops/meerkat

available publish methods or a user defined publishing method. Current publish methods support JSON messages over MQTT or a HTTP REST endpoint. The MQTT server and REST endpoint can be dynamically configured using the IPS configuration software running in the browser.

4.3.4. Localisation algorithms for staff wristbands

4.3.4.1. Introduction

Recall that an ultra-wide band (UWB) system consists of anchors at known locations and mobile tags at unknown locations. The staff wristband contains an UWB tag. We want to determine the location of the mobile tags over time. At each time step the anchors measure its distance to the mobile tag which can be used to estimate the position of the tag.

A state model is an algorithm that uses a series of measurements observed over time, containing statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more accurate than those based on a single measurement alone, by using Bayesian inference and estimating a joint probability distribution over the variables for each timeframe. In our case the state consists of the position and velocity of the tag.

We can distinguish two types of errors in the measured distances, statistical noise due to e.g. atmospheric fluctuations and inaccuracies when the direct line-of-sight is blocked: non-line-of-sight (NLOS). Statistical noise is assumed to be Gaussian distributed and can be both positive and negative. The line-of-sight can be blocked by metal objects such as reinforced concrete walls or containers of water like human bodies. In a NLOS situation the signal between the tag and an anchor is delayed by reflections or attenuation resulting in an increased measured distance. The detection and mitigation of NLOS is important for accurate localisation.

In the section below we introduce some theory on the state- and the observation models that are used in the several versions of the Extended Kalman (EKF) and Finite Impulse Response (FIR) filters.

4.3.4.1.1. State model

The state model position-velocity (PV) has been adopted, it considers dynamics with constant speed. According to this, the state vector $(\hat{\mathrm{x}}_\mathrm{k})$ is expressed as:

$$\hat{\mathrm{x}}_\mathrm{k} = [p^n v^n]^\mathrm{T}, \tag{11}$$

where, p^n and v^n are n-dimensional position and velocity vectors, respectively. Following, the equations that describe the dynamics of the system:

$$p_k^n = p_{k-1}^n + v_{k-1}^n \Delta t_k + 0.5 a^n \Delta \mathrm{t}_\mathrm{k}^2, \tag{12}$$

$$v_k^n = v_{k-1}^n + a^n \Delta t_k, \tag{13}$$

where, Δt_k is the elapsed time between the previous estimation time t_{k-1} and the current estimation time t_k, and a^n is a n-dimensional vector of independent random accelerations normally distributed.

The dynamic model is presented in its matrix form, below, in order to relate it with the EKF. The Jacobian matrix of the state transition function f is defined as:

$$\mathrm{F}_k = \begin{bmatrix} \mathrm{I}_n & \Delta \mathrm{t}_k \mathrm{I}_n \\ 0_n & \mathrm{I}_n \end{bmatrix}, \tag{14}$$

where I_n is a n xn identity matrix and 0_n is a n xn matrix with all zero entries.

The process noise covariance matrix Q_k is defined as:

$$\mathrm{Q}_k = \begin{bmatrix} 0.5\Delta \mathrm{t}_k^2 \mathrm{I}_n \\ \Delta \mathrm{t}_k \mathrm{I}_n \end{bmatrix} diag\left(\sigma_{an}^2\right) \begin{bmatrix} 0.5\Delta \mathrm{t}_k^2 \mathrm{I}_n \\ \Delta \mathrm{t}_k \mathrm{I}_n \end{bmatrix}^T, \quad (15)$$

where $diag\left(\sigma_{an}^2\right)$ is a diagonal matrix with the variances of a^n that allow to track the different forces that could temporally affect the target's dynamics (e.g., friction) (Caceres et al. 2009).

4.3.4.1.2. Observation model

The ranging measurements are the observation vectors ($\mathrm{z_k}$), for the EKF. They are defined as the distances between a mobile node and L anchor nodes within signal range and are normally distributed with zero mean and variance $\sigma_{d_{ref}}^2$.

$$\mathrm{z_k} = \left[\tilde{d}_{ref1} \cdots \tilde{d}_{refL}\right]^T, \quad (16)$$

where the distance $\tilde{d}_{refl}$ with $l \in [1, L]$, is the estimated ranging measurement. The measured distance is modelled as:

$$\tilde{d}_{refl} = d_{refl} + \eta, \quad (18)$$

$$\eta \sim N\left(0, \sigma_{d_{ref}}^2\right), \quad (17)$$

where η is the observation error which is Gaussian distributed with zero mean and standard deviation σ_{dref} as in (17). d_{refl} is the true ranging measurement defined as:

$$d_{refl} = dist(p, p_{refl}) = \sqrt{\sum_{i=1}^{n}\left(p_i - p_{i,refl}\right)^2}, \quad (19)$$

Typically the observation errors are modelled as uncorrelated white Gaussian noises so the covariance matrix R_k depends on the variance of the measurements $\sigma_{d_{refl}}^2$,

$$\mathrm{R}_k = \mathrm{diag}\left(\sigma_{d_{ref1}}^2, \cdots, \sigma_{d_{refl}}^2, \cdots, \sigma_{d_{refL}}^2\right), \quad (20)$$

Consequently, the observation function h is defined as the distances between the position component of the state vector and the reference nodes,

$$\mathrm{h}(x_k) = \begin{bmatrix} \mathrm{dist}\left(\mathrm{p_k}, p_{ref1}\right) \\ \vdots \\ \mathrm{dist}\left(\mathrm{p_k}, p_{refL}\right) \end{bmatrix}. \quad (21)$$

Since $\mathrm{h}(x_k)$ is nonlinear, the Jacobian matrix H needs to be computed around the a priori state $x_{k|k-1}$,

$$\mathrm{H_k} = \begin{bmatrix} \frac{\mathrm{p}_{1,\mathrm{k}|\mathrm{k}-1} - p_{1,ref1}}{\mathrm{dist}(\mathrm{p_k}, p_{ref1})} & \frac{\mathrm{p}_{2,\mathrm{k}|\mathrm{k}-1} - p_{2,ref1}}{\mathrm{dist}(\mathrm{p_k}, p_{ref1})} & \frac{\mathrm{p}_{3,\mathrm{k}|\mathrm{k}-1} - p_{3,ref1}}{\mathrm{dist}(\mathrm{p_k}, p_{ref1})} & 0_v & 0_v \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\mathrm{p}_{1,\mathrm{k}|\mathrm{k}-1} - p_{1,refL}}{\mathrm{dist}(\mathrm{p_k}, p_{refL})} & \frac{\mathrm{p}_{2,\mathrm{k}|\mathrm{k}-1} - p_{2,refL}}{\mathrm{dist}(\mathrm{p_k}, p_{refL})} & \frac{\mathrm{p}_{3,\mathrm{k}|\mathrm{k}-1} - p_{3,ref1}}{\mathrm{dist}(\mathrm{p_k}, p_{refL})} & 0_v & 0_v \end{bmatrix}, \quad (22)$$

where, 0_v is a row vector with n zero entries.

4.3.4.2. EKF with NLOS detection

The Extended Kalman Filter (EKF)(H. Cox. 1964) is a suboptimal estimator for nonlinear state models. We can track an UWB tag with the EKF and a position-velocity (PV) model (Caceres et al. 2009). The EKF estimates the tag position from the state at the previous time step and the measured distances at the current step. Important parameters of the filter are the measurement errors and process noise. The errors in the distances are around 5 cm for the Decawave UWB chip. The process noise can be treated as a tuning parameter to adjust the EKF to smooth out either more or less data as a tradeoff between estimation accuracy and time lag. Setting the process noise to a wrong value can even lead to divergence of the filter.

To detect NLOS we have tried two different methods, both of them use estimated distances. At each time step we calculate the distances of the estimated position to the anchors giving estimated distances. We compare the estimated to the measured distances at the next time step. The first method detects NLOS when a measured distance jumps too much from the estimated one. The second method looks at jumps in jumps which corresponds to acceleration. For both methods the NLOS is mitigated by replacing the measured distance by the estimated one.

4.3.4.3. EFIR with NLOS Detection

A disadvantage of the EKF is that it requires estimates of the measurement and process noise, which may not be known exactly in practice especially for time-varying models. Therefore the EKF may lose in accuracy or precision so it is not a robust estimator. The Kalman filter is an Infinite Impulse Response (IIR) filter which means that the effect of outliers may last for a long time.

An alternative to the Kalman filter is Extended Finite Impulse Response (EFIR) filtering (Shmaliy et al. 2017) which uses a limited number of samples to estimate the position. The EFIR filter uses the same state model as the EKF but it does not need the measurement of the process noise and is therefore more robust. The EFIR filter only requires to set the number of samples used. A disadvantage is that the EFIR filter has more lag than the EKF. The lag can be minimised by choosing the right length of the filter.

For the EFIR filter we use the same NLOS detection and mitigation as for the EKF.

4.3.4.4. EKF with outlier mitigation

The EKF provides an efficient computational recursive algorithm that estimates the process state by minimising the mean of the squared error. However, the performance of KF, in general, degrades when the observed data contains outliers (J. A. Ting et al. 2007). The proposed EKF, identifies outliers while tracking the observed data. Moreover, it is adaptive and there is no need for parameter tuning or the use of heuristic methods. The outlier mitigation has been implemented by adopting the approach proposed in (J. A. Ting et al. 2007). In general, the EKF with outlier mitigation is implemented as described in (G. Welch et al. 2006) with a variation in the update phase. In particular, during the update phase, the observations are dynamically weighted accordingly with the error.

According to the simulator in section 4.3.6.1 as well as the UWB measurements distribution presented in section 4.3.4.6, simulation results showed an RMS error of 0.07 m with 60% presence of outlier measurements during the Monte Carlo simulation. Besides, as benchmarking, these results has been compared with a PV EKF algorithm (Caceres et al. 2009) with a RMS error of 0.54 m.

4.3.4.5. Particle Filter

The adopted particle filter is the same as in section 4.2.6.2. The main difference is in the likelihood function which is derived for ToA measurements. Thus, the likelihood function depends on the p.d.f. of ToA (ranging) measurements, tag and anchors positions and the related particles.

Simulation results provided a RMS error of 0.28 m, outperforming the LSS algorithm (S. Gezici et al. 2008) with a RMS error of 0.79 m, under same simulation conditions.

4.3.4.6. UWB ranging measurements campaign and CRLB analysis

The UWB ranging measurements campaign has been foreseen at the KFF, already described in section 4.3.9. Analysis performed at collected data showed 0.13 m of standard deviation for ranging measurements. Besides, the mean ranging error has been 0.18 m with respect to a few anchor nodes and 1.46 m for the majority of them. Previous results show that there were outlier ranging measurements, probably, due to interferences from the environment. As a consequence, this took place to biased ranging measurements.

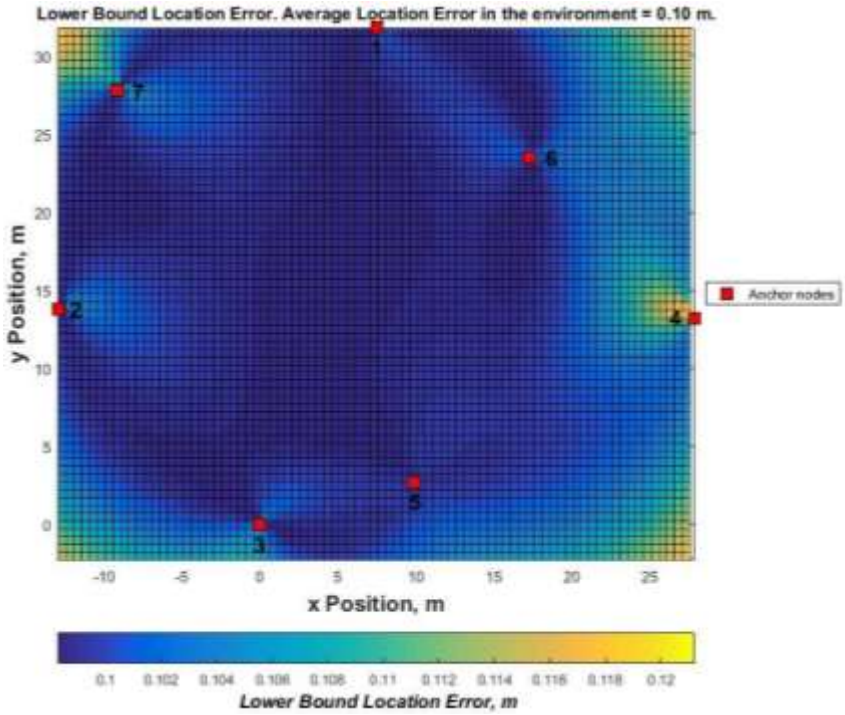


Figure 17: UWB measurements CRLB analysis

The CRLB analysis has supposed the ranging model for ToA ranging measurements (i.e., normally distributed with zero mean and standard deviation σ_{dref}), which represents the UWB measurements. From KFF outcomes, σ_{dref} has been set to 0.18 m. Figure 27 shows the CRLB analysis results. The expected location error considering anchor nodes distribution and the ranging model is 0.10 m, with a standard deviation about zero meters. In general, a low localisation error is expected over the localisation area, in presence of good environmental conditions (i.e. no presence of obstacles nor possible disturbances) and a good distribution of anchor nodes, as demonstrated in the picture above.

4.3.5. Posture algorithms for staff wristbands

4.3.5.1. System configuration

In the initial stage, an Axivity AX3 accelerometer was used for activities data collection. The AX3 sensor was tied at the righthand wrist using a wristband as shown in Figure 20. Also the orientation of the AX3 axis is shown, inside and outside the wristband.

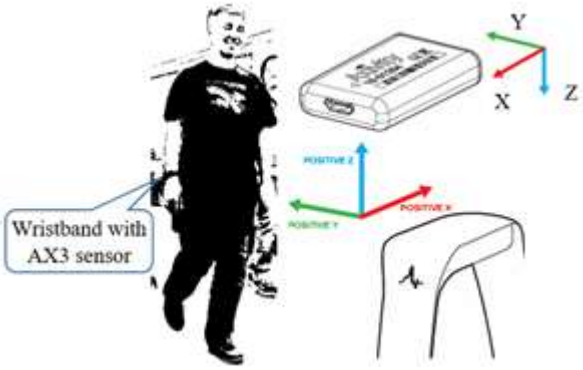


Figure 18: System configuration

4.3.5.2. Data collection and features extraction

The raw data vectors $(t, Ax(t), Ay(t), Az(t))$ were collected from the arm attached accelerometer. Then more feature vectors $(Axyz(t), \Delta A(t), \varphi(t), \theta(t),)$ were extracted using the following equations from (1) to (4) respectively:

$$Axyz(t) = \sqrt{A_x(\mathrm{t})^2 + A_y(\mathrm{t})^2 + A_z(\mathrm{t})^2} \quad (1)$$

$$\Delta A(t) = \left|A_{xyz}(t) - A_{xyz}(t-1)\right| \quad (2)$$

$$\varphi(t) = Atan\left(\frac{A_y(t)}{A_z(\mathrm{t})+0.00001}\right) \times \frac{180^{\circ}}{\pi} \quad (3)$$

$$\theta(t) = Atan\left(\frac{-A_x(t)}{\sqrt{A_y(\mathrm{t})^2+A_z(\mathrm{t})^2+0.00001}}\right) \times \frac{180^{\circ}}{\pi} \quad (4)$$

where Axyz is the three dimensional acceleration, ∆A is the absolute Axyz change over a time interval; φ and θ are the sensor rotation angles around the X and Y axis respectively. The number 0.00001 in (3) and (4) is an adjustment constant studied to avoid the zero divisor in occasion, and also keep the same results for all non-zero divisor cases. Thus, there are 7 features (Ax, Ay, Az, Axyz, ∆A, φ, θ) used for the model training and testing.

4.3.5.3. Model training using MLP algorithms

The multi-layer perceptron (MLP) algorithm is used for NN classifiers training. MLP trains using gradient decent with back-propagation. It can learn one or more nonlinear layers (called hidden layers) between the input and the output layer.

In this study, the input $X = \{X_i|x_1, x_2, \dots, x_7\} = \{X_i|A_x, A_y, A_z, A_{xyz}, \Delta A, \varphi, \theta\}$ and the output $Y = \{Y_i|y_1, y_2, \dots, y_8\} = \{Y_i|walking, running, waving, punching, fistclenching, slapping, throwing, still\}$

How many hidden layers and how many neurons for each of the hidden layers should we set up for our MLP? There are no theoretical answers to this question, so an experiment was done for comparison of the classification accuracy using different hidden layers and different neuron numbers at each of the layers. The experimental results are shown in Table 2.

One hidden layer MLP										
Neuron_No	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
Acc.(%)	0.69	0.67	0.68	0.70	0.70	0.71	0.69	0.61	**0.72**	0.71
Two hidden layers MLP										

Neuron_No	*(6,6)*	*(7,7)*	*(8,8)*	*(9,9)*	*(10,10)*	*(11,11)*	*(12,12)*	*(13,13)*	*(14,14)*	*(15,15)*
Acc.(%)	0.71	0.66	0.68	0.70	0.71	0.71	**0.73**	0.63	**0.73**	**0.73**
Three hidden layers MLP										
Neuron_No	*(6,6,6)*	*(7,7,7)*	*(8,8,8)*	*(9,9,9)*	*(10,10,10)*	*(11,11,11)*	*(12,12,12)*	*(13,13,13)*	*(14,14,14)*	*(15,15,15)*
Acc.(%)	0.56	0.58	0.56	0.70	0.68	0.54	**0.72**	0.67	**0.72**	0.67

Table 2: Comparison of classification accuracy for different hidden layers and different neuron numbers at each of the layers

Table 2 illustrates that two hidden layers with 12 neurons for each of the 2 layers has better performance for our human activity classification. Therefore, two hidden layers MLP (12,12) is selected in this study, as shown in Figure 21.

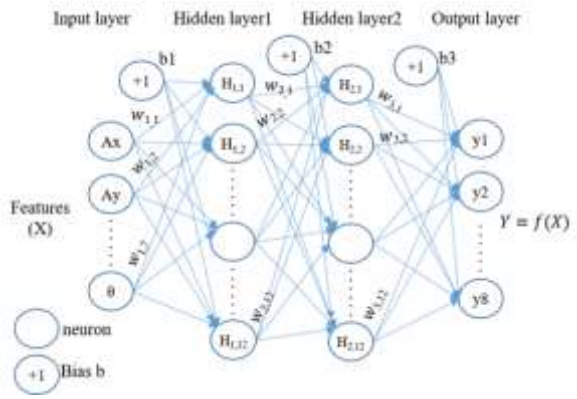


Figure 19: Two hidden layers MLP used in this study

Where the $w_{i,j}$ values are called weights. They represent the "strength" of the connection between two neurons. Bias nodes (b1, b2, b3) are added to increase the flexibility of the model. Specifically, it allows the network to fit the data when all input features are equal to 0. The value of a bias node is set to 1 without regard for the data in a given pattern.

MLP learns the function y = f(x,w) from Eq. (5), and calculates the probabilities of the sample x_i belonging to each of the 8 class. The output is the class with the highest probability:

$$y = \sum_{k=1}^{12} w_{3,k} \cdot f2\left(\sum_{j=1}^{12} w_{2,j} \cdot f1\left(\sum_{i=1}^{7} w_{1,i} \cdot x_i + b1\right) + b2\right) + b3 \quad (5)$$

where w_1, w_2 and w_3 are the weights of the input layer, first and second hidden layers respectively; b1, b2 and b3 are single bias nodes added for the input layer, first and second hidden layers respectively. The f1 and f2 are the activation functions applied for the two hidden layers respectively.

For multi-class classification the softmax function is used as activation function f1 and f2, which is written as Eq. (6):

$$Softmax(z)_i = \frac{\exp(z_i)}{\sum_{i=1}^{k} \exp(z_i)} \quad (6)$$

where z_i is the i^{th} element of the input to softmax, which corresponds to class i, and k is the number of classes.

The weights (w) start from initial random values, then MLP updates these weights repeatedly using the loss (or error) function. In a multi-classification problem, the logarithmic loss function (Log Loss) is used and defined as (7). After computing the loss, a backward pass propagates it from the output layer to the previous layers, providing each weight parameter with an update value aiming to decrease the loss. In order to calculate Log Loss, the classifier must assign a probability to each class rather than simply yielding the most likely class:

$$Loss(L, y) = -\frac{1}{s}\sum_{i}^{s}\sum_{j}^{c} L_{i,j}\, log y_{i,j} \quad (7)$$

where s is the number of training samples, c is the number of classes, y is the predicted value and L is the target value. The minus sign at the beginning aims to minimise the function result as positive values, since log of a number ($y_{i,j}$) between 0 and 1 is negative.

Log Loss quantifies the accuracy of a classifier by penalising false classifications. Minimising the Log Loss is basically equivalent to maximising the accuracy of the classifier. For example, if the prediction ($y_{i,j}$) is very close to 1 then log of that number will be very close to zero, which means the error for that particular case will be very close to zero.

4.3.5.4. Plurality voting algorithm

In order to improve the activity classification accuracy and robustness, a plurality voting algorithm (pluralityVA) was used to adjust the predicted result by an original classification algorithm (originalA). The originalA can be any of the machine learning algorithms. In this case MLP has been introduced. Details of this pluralityVA algorithm is described below:

(4) There exists a classifier that defined a list of class labels$L = [l_1, \dots, l_c]$, where c is the total class number

(5) The predicting result by the originalA is denoted as a list $P = [p_1, \dots, p_n]$, where for all $p_i \in L$, and n is the sample number of the testing set in total

(6) A window size is set as 1 s period of time (25 samples in this study). Count the number of each class N_{l_i} ($N = [N_{l_1}, \dots, N_{l_c}]$) for every sliding window (w) from the predicted result $P(w)$ in (8)

(7) Obtain the relevant majority class label key in (9), and use this key value to replace all values in $P(w)$using $P_{plur}(w)$ in (10).

$$\exists \begin{cases} L = [l_1, \dots, l_c] \\ P_{orig} = [p_1, \dots, p_n], \forall p_i \in \mathrm{L} \\ w = 1s = 25samples \\ N = [N_{l_1}, \dots, N_{l_c}] \end{cases}$$

$$N = counter(P(w) = [p_i, \dots, p_{i+w}]|L) (8)$$

$$key = l_i : \max(N) \ (9)$$

$$P_{plur}(w) = [key] * w (10)$$

Using the pluralityVA algorithm can improve the robustness of the overall system, since it sets up the same class label using a relevant majority class value for every sliding window P(w), thus reducing the number of possible miss-classified samples.

4.3.5.5. Experiments

Data from 17 subjects were collected. All subjects performed 8 actions thus there are 8 classes: {walking, running, waving, punching, fist clenching, slapping, throwing, still}. The experimental results were validated against synchronised videos, recorded with 3 cameras installed on the ceiling or top of a wall.

The experiment protocols were performed as follows: first, the subject 1 (sub1) performed the 8 actions in a stated order and the collected dataset saves into a file as singSub. Subsequently the 17 subjects (sub1~sub17) were organised in two groups (5+12), and each of the 2 groups performed the 8 actions at the same time twice in the stated order and in a random order respectively. In total 35 datasets (1+17*2) have been collected, organised as different types of training sets and testing sets as shown in Figure 22.

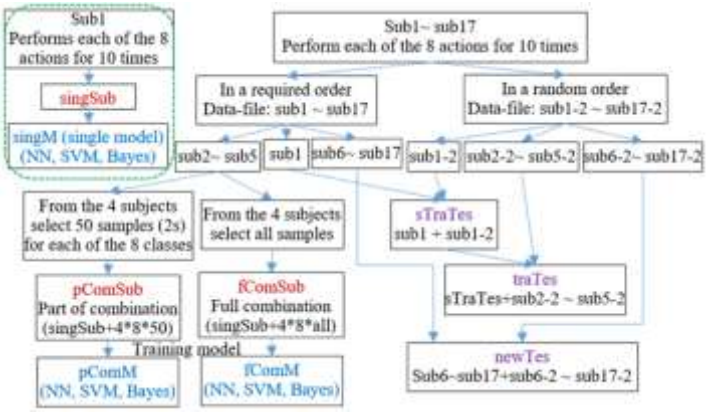


Figure 20: Datasets organisation

The 35 datasets were collected from 17 subjects and organised as three types of training sets (in red colour) with three types of models (in blue colour) and three types of testing sets.

Three types of training sets and testing sets were organised in different ways as shown in Table 3. Three types of models (singM, pComM, fComM) were trained based on three corresponding training sets respectively.

Experimental results were demonstrated and compared in Table 3, which demonstrates that the classification accuracy is improved decisively by using the pluralityVA, compared to each of the 3 models. For example, it is 92% vs. 76% compared to singM for the sTraTes set.

Testing sets	Algorithms	Classification accuracy for three models		
		singM	pComM	fComM
sTraTes	*originalA*	0.76	0.75	0.73
	pluralityVA	**0.92**	**0.91**	**0.90**
traTes	*originalA*	0.42	0.49	0.54
	pluralityVA	0.50	0.57	0.67
newTes	*originalA*	0.40	0.48	0.53
	pluralityVA	0.47	0.58	0.64

Table 3: Three types of testing sets classified by 3 models and compared between originalA and pluralityVA algorithms

The fully combined model fComM has better performance for the two testing sets traTes and newTes. However, the single model singM has the best result for the single testing set sTraTes. For example, the three models singM vs. pComM vs. fComM is 47% vs. 58% vs. 64% for the newTes set, is 50% vs. 57% vs. 67% for the traTes set, and is 92% vs. 91% vs. 90% for the sTraTes, using the pluralityVA algorithm.

A plurality voting mechanism can be used to adjust the original prediction result for improving the robustness of the overall system. Every subject has a different performance for the same action, for example, somebody performs a slapping action with a stretched arm, while some others do this with a bent arm. For this reason, training an individual model is a very efficient way for improving the system robustness and reliability.

For example, Figure 23 shows a punching action scenario. It shows a frame of the synchronised video screenshots and the corresponding Ay signals. For clarity, only 4 subject Ay signals are shown in this figure. These signals illustrate that different subjects have a different behaviour even though they perform the same action (punching in this case). Hence, how to train a model for different subjects is a challenging task.

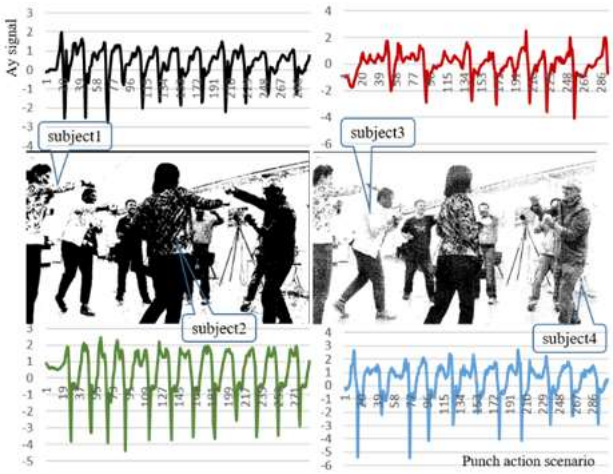


Figure 21: Synchronised frames and accelerometer Ay signals for a punching action

The activity classification results can be saved into a file, while can be visualised using acceleration signals (in this case only a Ay signal is used). Figure 24 illustrates the synchronised frames and visualised experimental results that are classified by using the two hidden layers MLP (12,12) with the introduced pluralityVA algorithms.

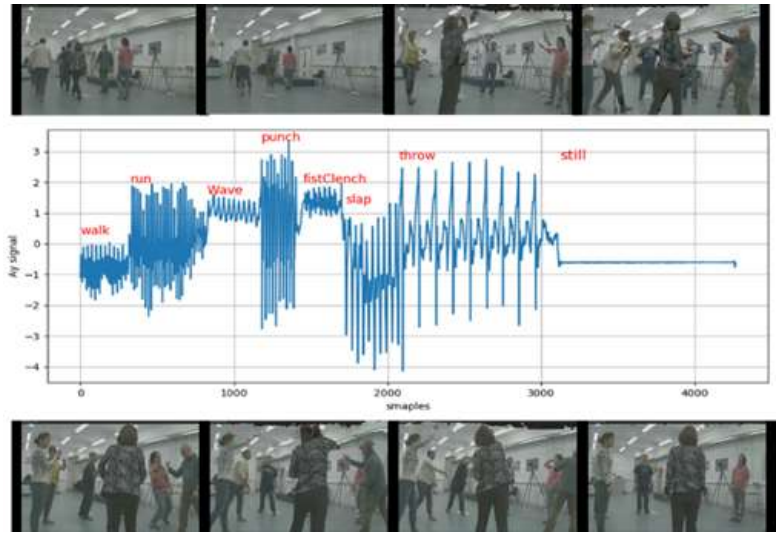


Figure 22: Synchronised frames and visualised experimental results for 8 actions, using MLP with pluralityVA algorithms

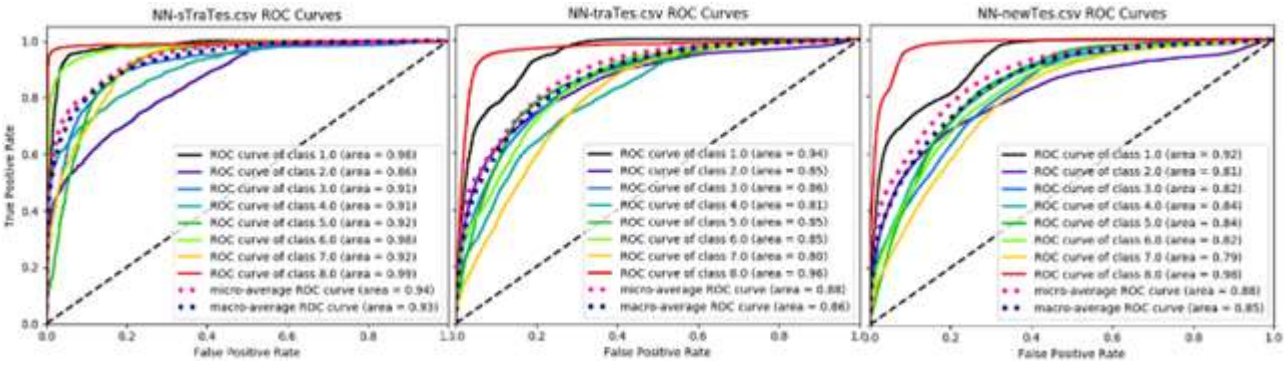


Figure 23: ROC curves for 3 types of testing sets (sTraTes, traTes, newTes)

Figure 25 illustrates the experimental results for three different testing datasets (sTraTes, traTes, newTes) using receiver operating characteristic (ROC) curves, which can show the trade-off between sensitivity and specificity for each of the 8 classes. For the sTraTes dataset (left figure), there are 3 curves (C8, C1, C6) in the ideal position (top left corner of the figure) and these curves also have highest steepness, which means these 3 classes got excellent prediction to maximise the true positive rate while minimising false positive rate. The area under the curve (AUC) is a measure of predicted accuracy for each of classes. An area of 1 represents a perfect prediction; an area of 0.5 (the diagonal line) represents a random prediction (worthless). Therefore, for the sTraTes set, the AUC for most of classes are more than 0.9, it is excellent prediction. However, for the traTes and newTes datasets, only class 8 and class 1 got an excellent prediction, most of the classes got a good (AUC >0.8) or fair (AUC > 0.7) prediction.

4.3.6. Anchor calibration

After the anchors are installed in their proper location and everything is powered and running, the system needs to be configured. Configuration is done by pointing a browser to a URL hosted by the Meerkat. The Meerkat runs a web server to configure and monitor the entire system. The following phases need to be completed before the system is fully functional:

- Anchor localisation
- Coordinate alignment

The position of the anchors can be found by completing a distance matrix including distances between all anchors. Based on the distance matrix, the anchors positions are calculated using an optimisation procedure. This optimisation procedure minimises the error between the measured and the calculated distances. The calculated distances are derived by applying the Euclidean distance formula that takes as input the estimated positions. The optimisation of the positions of the seven anchors took a couple of minutes. Simulated Annealing (S. Kirkpatrick et al. 1983) is used as the optimisation algorithm. It is worth highlighting that the anchors' positions will be expressed according to a relative reference system that depends on the optimisation algorithm and its initial parameters.

After the anchors' positions are determined, the reference system set by the optimisation algorithm can be aligned according to a user defined reference system. This step is not mandatory, but it helps in reasoning about the tracked tags positions as visualised on the screen. The coordinate alignment works by marking three known points in the area, typically the origin (0, 0) and two other points. These points can be measured using a (laser) measurement tool. After the coordinates of the marked points are measured, the values can be entered into the alignment tool offered by the web interface. Subsequently, three tags need to be positioned on the three points at a fixed height (z-axis). The system will figure out automatically which tags correspond to which three entered positions. Subsequently, the initial coordinate system (determined by the optimisation procedure) will be levelled to the user defined coordinate system.

Figure 24: Result of the automatic anchor positioning procedure

The result of the anchor positions after the optimisation procedure is shown in Figure 26. As it can be observed, there is a symmetric matrix reporting the distances between the anchors while the colour reflects the distance error calculated as the difference between the measured and the calculated distances. More specifically, a bright green colour means that the distance error is smaller than 2 cm, an orange colour means that the error is larger than 20 cm and red colour error indicates errors larger than 50 cm.

4.3.6.1. UWB-based simulator

The UWB-based simulator is composed by two types of MATLAB-based software. The first is based on the simulator presented in section 4.2.8 but set with the ToA approach as ranging technique with the respective parameters describing the UWB measurements model. In particular, it is used to simulate and evaluate the behaviour of the PF in section 4.3.4.5. The second simulator aims to simulate and evaluate the behaviour of the EKF with outlier mitigation in section 4.3.4.4 and to benchmark it with a traditional EKF (Caceres et al. 2009). Both of them implement the same dynamic model (PV model). Its main features are random generation of outlier measurements and profile generation for the kinematics modelling. In particular, the outlier measurements are generated adopting two p.d.f., Gaussian and Uniform distributions respectively. Thanks to the kinematics modelling, the simulator provides tracking parameters such as: acceleration, speed, location and attitude, for a mobile node. These parameters are generated by defining segments which are classified as constant-velocity straight, constant-acceleration straight, constant-altitude and constant-radius turns.

4.3.7. Integration of staff wristband into the IoT middleware

Same as integration of crowd wristbands (see section 4.2.9.).

4.3.8. Services enabled by staff wristbands

4.3.8.1. Security staff localisation

The staff wristbands allow for the localisation of staff members. As such they help to implement several use cases. Besides staff localisation the wristbands can be used to notify staff members by sending text messages that are displayed on the LED screen of the wristband.

4.3.8.2. Health/security incidents

By leveraging the IMU of the staff wristband certain health incidents can be detected. A wrist-worn accelerometer is used for recognition of abnormal activities related to stewards and visitors in crowded environments.

4.3.9. Preliminary testing of UWB at Kappa FuturFestival 2017

This paragraph reports on the ultra-wide band (UWB) tracking test that has been performed during the preparation of the Kappa FuturFestival (KFF) in Turin on the 6th and 7th of July 2017 and the first day of the KFF event on the 8th of July 2017. The deployment of the system and initial tests started on Thursday July 6th. In particular, the installation involved liaising with the KFF organisers to find the best positions of the fixed nodes in such a way the devices did not hinder the passage of people. An additional day was allocated to the setup time frame in case of unexpected setbacks and delays. Various tests have been performed on July 7th as well. On Saturday July 8th the festival was up and running as from noon. This paragraph describes the test area, the test setup, the calibration procedure, the test results and conclusions.

4.3.9.1. Test area and setup

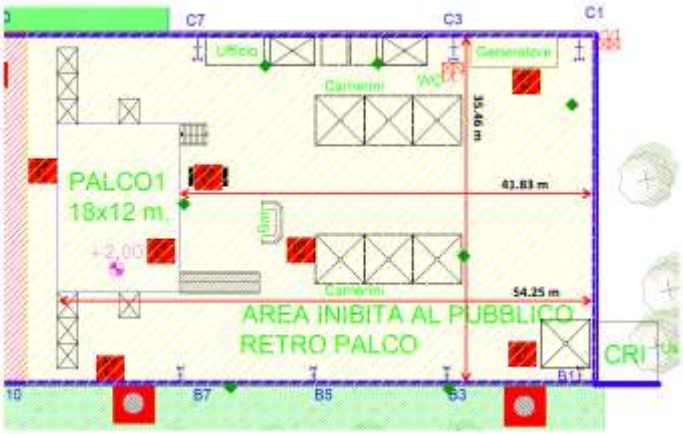


Figure 25: Blueprint, anchor locations and dimensions of the main backstage test area

The assigned test area is the so called “main backstage” area, see Figure 17. This is the area behind the main stage (Jäger stage). This area is designated for production, hospitality, first aid, storage and it contains the artist area. The backstage area contains a couple of structures, including (office) containers, power generators and tents. The area is located under the characteristic old factory roof. The area is not crowded with people, although there was a reasonable “activity” with people moving around during the event. The locations of the seven anchors are indicated by the green diamond shapes.

For the test, seven UWB anchors were deployed (see Figure 17 for the locations). The anchors were mounted on tripods with a height of 4m. Six of the seven anchors were powered by using a power bank battery. Besides power and a UWB interface, no additional (network) connections were required for these six anchors. One anchor was designated as the master anchor. The master anchor was connected to a Power over Ethernet (PoE) switch providing power as well as network connectivity. A router was connected to the same switch to provide a DHCP server. In addition, a small Linux server (Meerkat from System76) was connected to the same switch acting as the wristband gateway (GW) (in line with the MONICA architecture). Finally, a Macbook Pro laptop was connected to the switch to visualise the tracking results and for configuration purposes.

Testing entailed tracking of a set of small battery-powered UWB tags. Every tag continuously communicated via the UWB radio with the anchors to determine the range to each of the visible anchors. All the collected ranges from anchors were sent to the master anchor (wirelessly, using the UWB radio). The master anchor published the collected ranges in a UDP packet. An application running on the Meerkat consumed this UDP

packet. Subsequently, the position, based on the available tag-anchor distances was calculated on the Meerkat itself using an Extended Kalman Filter (EKF) algorithm. The application on the Meerkat also provided a web-based application that was used for visualisation and configuration. A browser running on the Macbook was used to connect to the application running on the Meerkat.

Figure 26: Setting up the anchors in the test area

Setting up the seven anchors, the network and the server, after approval of the positions, took about one hour.

4.3.9.2. Test results

During the three test days several more tests have been performed. The results reported here summarise the most important findings. Since all the tests have been performed in an area that was not crowded, it is difficult to extrapolate these results to crowded areas.

Figure 27: Tracking of a walk through the artist area

Figure 19 shows a “walk” through the artist area. The screenshot also shows different (green) lines between the tag and each of the anchors representing the UWB connectivity. In the top part of the screen the distances and signal strengths (RSSI) are shown in a table together with the current tag position and speed. On the bottom part of the screen, various time plots are depicted showing distances over time to each of the seven anchors. In particular, the blue points show the calculated distances (from the current estimated position), while the green points are the time-corresponding measured distances by using UWB messages. An orange

point means that a possible non line of sight (NLOS) situation has occurred which means that the line of sight between the tag and the corresponding anchor was blocked. A blocked object may cause attenuation of the signal or the ranging of a secondary (due to reflection) path. Both result in a measured distance that is too inaccurate. The NLOS determination tries to mitigate this situation by disabling the use of the anchor that has NLOS in the calculation of the position.

The official test area only measured 40 m x 40 m. The seven anchors were positioned at the edges. However, the tracking system also allows for tracking tags outside this area. The size of this extended area depends on the communication range between the tag and the anchors. As long as at least three anchors can be “seen” by the tag, the position can be calculated in two dimensions (i.e. x,y). This is shown in the following screenshot. The person with the tag went outside of the test area (going from the back stage to the front of the stage, public side). Two different ways of holding the tag have been used: (1) making sure that the tag is never blocked by a body (2) making sure that the tag is always blocked by a body. It is shown that the first situation results in a better range, i.e. the tag can be tracked at longer distances. In the first situation the tags is still tracked at distances 80 m away from the nearest anchors. In the NLOS situation, the tag is “lost” much sooner.

4.3.9.3. Conclusions

The results show that it is perfectly feasible to track persons in large areas using UWB. The accuracy of the tracking is well within 1 m, which seems to be a reasonable figure for the crew tracking use case. A major advantage of this tracking method compared to e.g. GPS is accuracy and independence of mobile network infrastructure. The UWB infrastructure is completely under self-control without any third party dependencies.

Since a Kalman filter is used, which will keep on predicting new positions even in the absence of new measurements, it can happen that completely false paths are reported. It would be better to show the last estimated position based on at least three recent measurements instead of showing a completely artificial predicted position. This will be changed in the next version of the algorithm.

NLOS mitigation is a very important requirement that will need more work. Implementing more localisation algorithms is also something that will be useful in selecting the right localisation method for this use case. Besides the current Kalman Filter, a FIR- (Finite Impulse Response) and a PM (Particle Method) filter will be implemented and tested within the Task 3.4 activities. The data collected during this measurement campaign can be used to test these new algorithms on exactly the same benchmark data.

The current maximum number of anchors that can be supported with one master is 16. For larger areas multiple masters need to be used. We will work on supporting multiple masters in our software such that for a large scale deployment more than 16 anchors can be used.

With respect to ease of installation, several things can and need to be optimised. The anchors need to be able to run for several days, therefore a high capacity battery is needed that can be easily charged as well. An IP67 grade battery/charging unit will be developed for the next deployment. In addition, IP67 grade anchors will be used instead of the indoor anchors that have been used in this test.

Since all the tests have been performed in an area that was not crowded, it is difficult to extrapolate these results to crowded areas. Additional tests are needed to gain insight on tracking in crowded areas.

4.4. Smart glasses

ORA-2 Smart Glasses are fully produced by Optinvent. The product allows a Full See-through feature (Transparent display) for a mobile wearable device. The ORA-2 product uses proprietary & patented Optinvent’s Display technology.

4.4.1. Technology overview

ORA-2 is a full standalone Android device that could run any Android application and connects smoothly to any other device.

Here is a summary of ORA-2 product features:

- Standalone wireless product with embedded battery, weight ~90 g
- Runs generic Android Kitkat (4.4.2) and has open platform to create and execute any Android applications.
- Use Arm 9 Dual Core 1.2GHz processor with GPU & 5.3 Gb Storage Memory and 1Gb Flash Memory

- Track pad on rigid side of the frame (True Mouse not only swipe)
- Embedded 1200 mAh rechargeable battery
- Can be worn over most of the user's glasses

Has the following sensors:

- 5 megapixel Auto-Focus camera
- Light sensor to adjust automatically display brightness to environment brightness
- Active GPS
- 9 axis sensors (Gyro/Accelerometer/Compass; MPU9250 from Invensens)
- Low noise microphone and mono audio out through µUSB provided accessory

Connectivity:

- WiFi b,g,n; 2.4GHz
- BT 4.0 Low Energy
- µUSBS 2.0 for charging, data exchange and transporting mono audio to an audio headset accessory

Display:

- See-through Display Feature with 50% photopic transparency
- 800x480 pixels native resolution RGB colour display with 42 pixel/deg
- Field Of View of 22deg with Flip-Vu feature to move the image location 5deg up, centred to eye sight and 20deg down
- High brightness

Figure 28: ORA-2 Product - Android standalone Smart Glasses eyewear

4.4.2. Services enabled by smart glasses

Communication with app, UWB wristband using BLE Show position based context on screen.

- Position from UWB + heading from smart glass
- ORA-2 could host any service already developed on Android smartphone & tablet. Optinvent already developed several services such as:
    - Specific launcher to better navigate between applications with large icons
    - Specific service to control the glasses wirelessly from an Android smartphone screen
    - Others free service to connect, manage the glasses with PC as well as with other devices

4.4.3. Integration of smart glasses with staff wristbands

ORA-2 could be connected to the wristbands platform by BT. However, a specific application should be built to allow both devices to have access to service in real time.

4.4.4. Integration of glasses into the IoT middleware

The ORA-2 glasses act primarily as a user interface device, i.e. running MONICA apps that the wearer interacts with. But it could also provide data from the built-in sensors that can be fed to the MONICA cloud. Examples of possible data sources are positioning from the GPS and data from the accelerometer.

The integration of ORA-2 with the MONICA IoT middleware should not present any problem, as it could be considered as a standalone Android device with WiFi & BT connectivity. So the integration would be done in the MONICA app that runs on the ORA-2 devices.

5. IoT Enabled Devices

5.1. Introduction

This section describes the work done within T3.5 IoT Enabled Devices until M12. The main objective of T3.5 is to provide integration of both fixed and nomadic devices into IoT enabled devices ready to integrate with the MONICA platform.

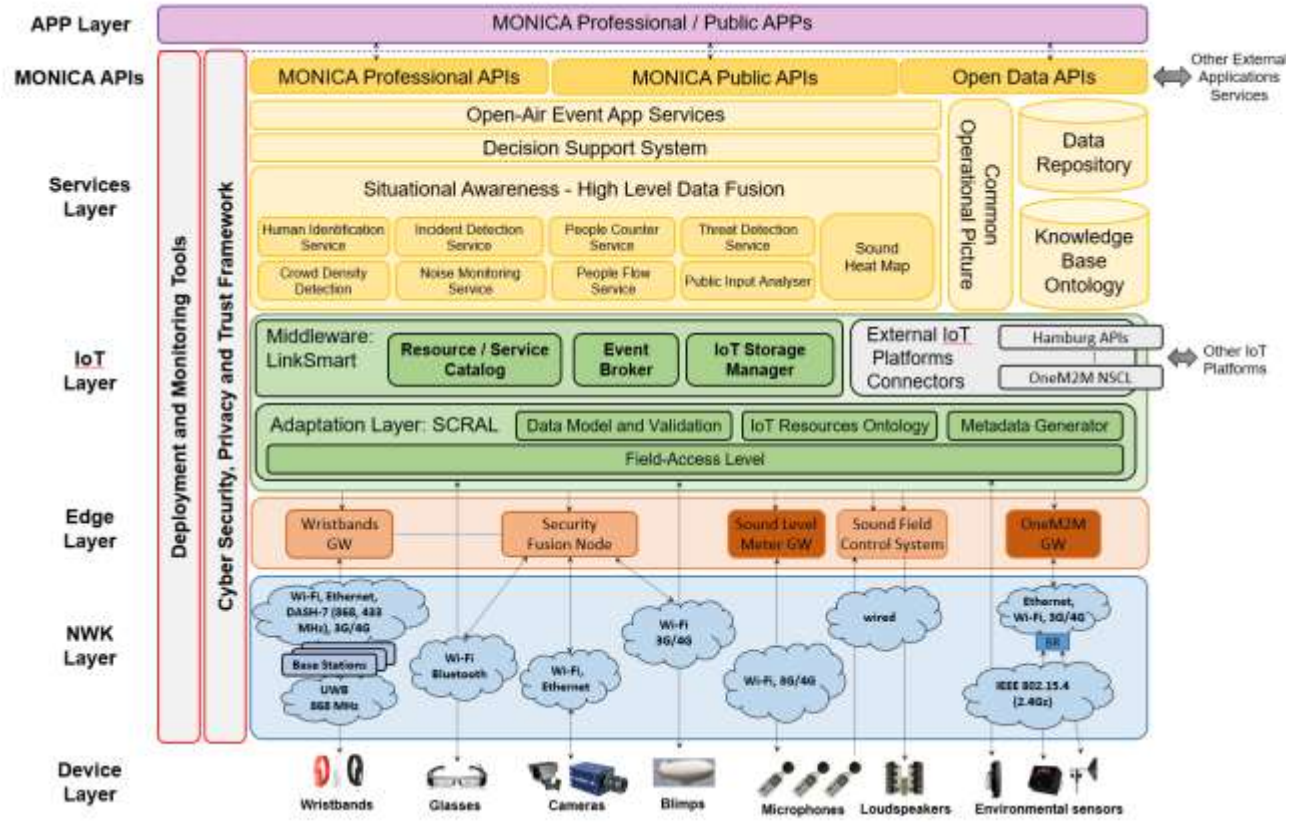


Figure 29: MONICA Architecture

Figure 29 shows the MONICA architecture, the work in T3.5 covers the part between the Device Layer and the IoT Layer. Depending on devices and functionality this involves “pure” devices as well as edge layer functionality. The basis for the MONICA IoT infrastructure is OGC SensorThings API, all the data handled in the IoT Layer is modelled according to this standard, see D3.3 IoT Secure Network Infrastructure and Semantic Middleware for a complete description.

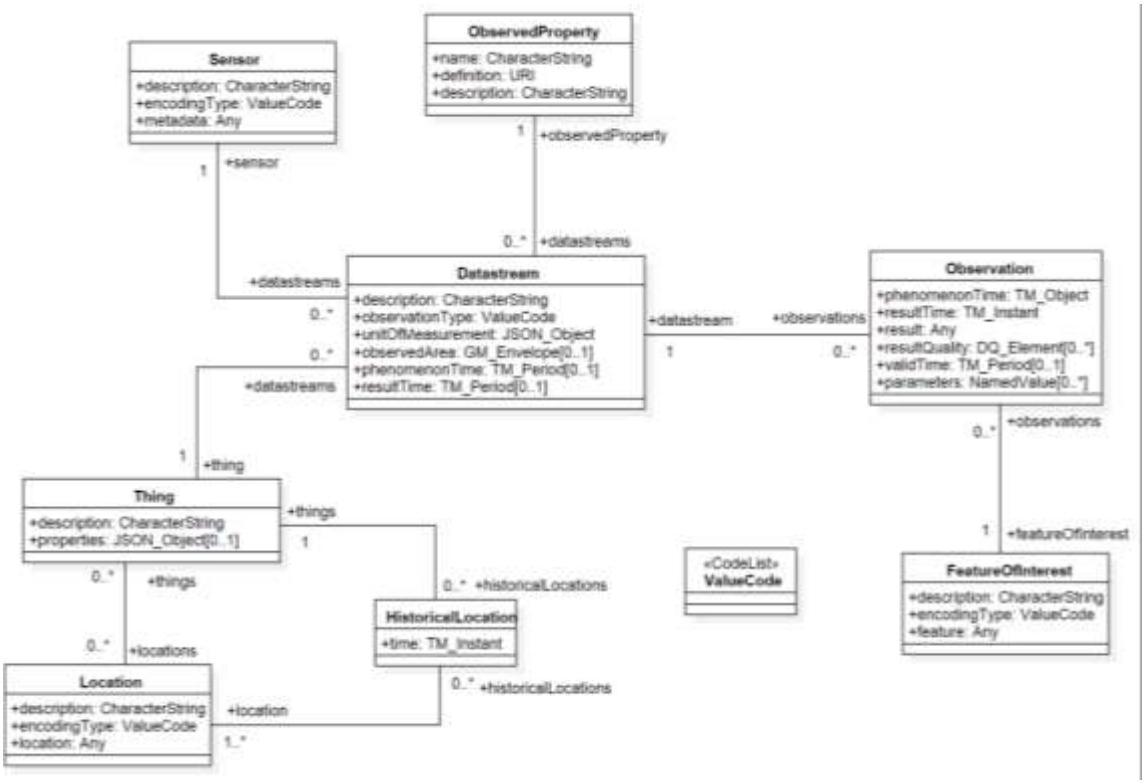


Figure 30: OGC SensorThings Model

The OGC SensorThings API Model, see Figure 30, is not sensor centric, it rather focuses on the things that are to be monitored. A thing in the MONICA case could, for instance be, guard A. For each thing there can be several sensors providing data-streams for different properties of the thing. Another thing to note is that the location is a special entity in the model, also providing historical locations. In MONICA the things that move we will not use this part of the model, the position will rather be treated as any data stream which makes it easier to handle at the data storage level without introducing a special position history.

In order to manage heterogeneous devices and inputs in MONICA the SCRAL provides an adaption layer for IoT devices and streams, transforming data to comply with OGC SensorThings as well as adding additional metadata according to the MONICA standard. Depending on the capabilities of the devices and the processing nodes in the Edge Layer, more or less adaptation is needed.

The following sub sections will describe the current state of the set of devices that are enabled to be part of the MONICA platform, but we foresee that additional devices will be added during the project duration. As soon as devices and their services are defined the actual IoT integration will be done. Devices that are already deployed at the pilot sites are not included but will depend on what is available at pilot sites and the selection of functionality deployed for a specific pilot. The same applies to external open data sources, such as weather services, that can be part of the MONICA platform.

5.2. Cameras and processing nodes

Detection of relevant events from video is enabled by running video analysis algorithms on the video streams in real time. Depending on the implementation of the analytics algorithm, it requires a specific amount of processing power to run in real time.

Whereas lightweight algorithms can run on the low-power CPUs embedded inside the cameras, some of the more advanced algorithms that implement computationally expensive approaches (such as fighting detection, crowd density estimation, etc.) cannot. These more expensive algorithms can only be run on a processing node with its more powerful processing abilities.

5.2.1. Technology overview

5.2.1.1. Cameras

Cameras with embedded analytics are available from VCA in two main modalities: monocular (regular CCTV cameras), and time-of-flight. Each camera type offers a range of video analytics with varying accuracies. The cameras feature an event alerting engine, that can send video and alerts in a range of formats such as HTTP, TCP, SMTP for integration into other layers of the MONICA architecture. Video is streamed in a range of formats including MJPEG over HTTP and H.264 over RTSP/RTP. Detailed technical specifications of all cameras are available in technical datasheets from the VCA website (VCA, 2017).

Figure 31: Monocular CCTV camera with embedded video analytics

Figure 32: Time-of-flight depth camera with embedded video analytics

The monocular cameras support a range of video analytics functions such as people counting, intrusion detection, loitering detection, etc. The time-of-flight cameras support highly accurate people tracking and can thus be used for accurate people counting and queue management.

While the monocular cameras with embedded analytics can offer good performance in constrained scenarios, they suffer from reduced accuracy under certain conditions such as crowded scenes, low light and severe occlusions. Time-of-flight cameras are not subject to the same limitations: they project a beam of infrared light, and measure the phase shift of reflected light, in order to build up a 3-dimensional point cloud consisting of horizontal and vertical displacement (x, y), as well as distance from the camera, depth (d).

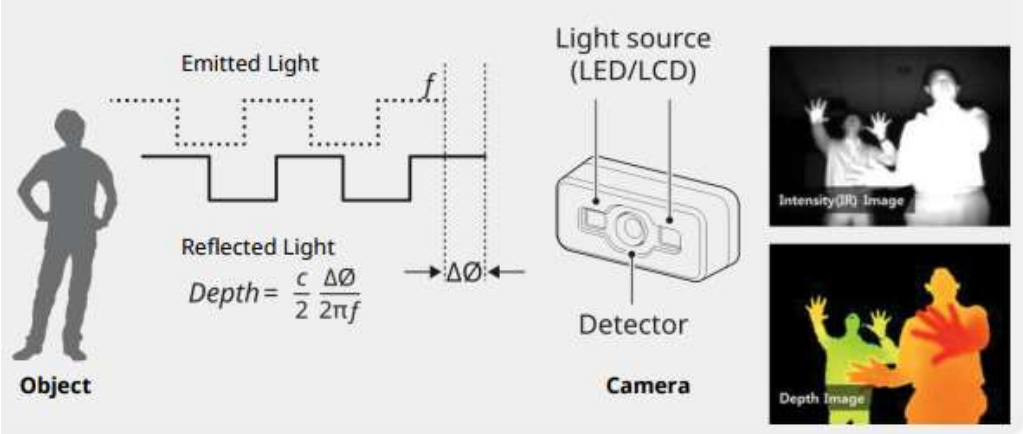


Figure 33: Phase shift of reflected light used to calculate the distance of objects from the sensor

Figure 34: (a) Infrared illumination image (b) Depth image (c) Illumination variation across the image for the selected horizontal line (d) Depth variation across the image for the selected horizontal line

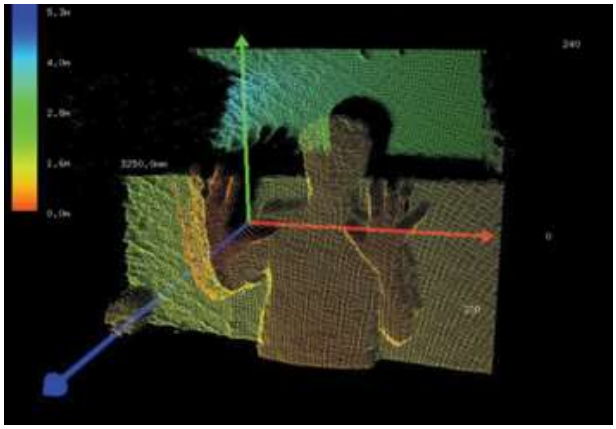


Figure 35: Three-dimensional point cloud showing x, y and depth values for each pixel

Since the time-of-flight sensor provides depth data for each pixel, it thus becomes much easier to resolve occlusions and track people accurately, even in cluttered and highly dynamic scenes.

5.2.1.2. Processing nodes

The specific hardware of the processing nodes can be relatively flexible: basically the more processing power required, the more/more powerful hardware can be installed. Nevertheless, VCA has a commercial appliance, the VCAbridge, which is a standalone hardware unit running a video processing framework that handles all of the monotony of interfacing with various cameras, and sending events off to third party systems.

Figure 36: VCAbridge Processing Node

The current version of the VCAbridge runs on an Intel i3 platform. However, with recent advances in deep-learning and the corresponding processing requirement to run on Graphics Processing Units (GPUs), VCA has started porting the processing framework to an embedded GPU, the nVIDIA TX1/2.

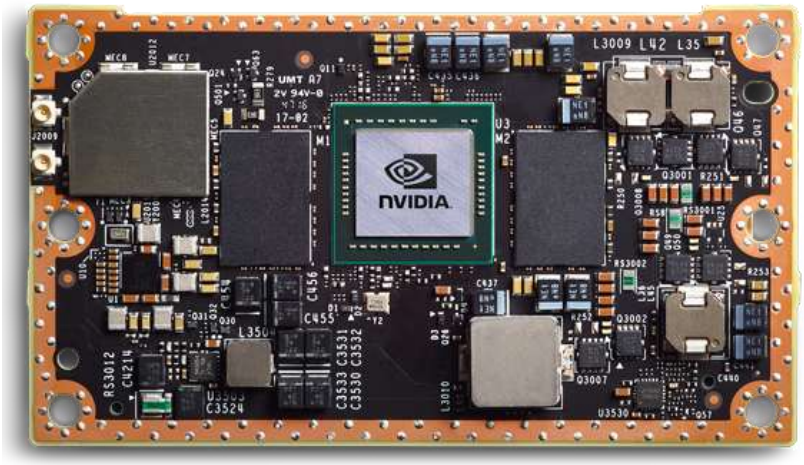


Figure 37: nVIDIA TX2 embedded GPU single-board-computer (SBC)

Therefore, some of the more demanding video analytics algorithms can be deployed on embedded GPU systems. An embedded GPU version of VCAbridge allows a specific installation at any given pilot event to rapidly up- and down-scale, depending on the number of cameras required, simply by adding or removing VCAbridge processing node units.

5.2.2. Infrastructure for cameras and processing nodes

The cameras and processing nodes both require power (12VDC for nodes POE, POE+ for cameras and time-of-flight cameras) and network connectivity (typically wired).

5.2.3. Services enabled by cameras and processing nodes

This section provides an overview of the services enabled by the cameras. For a more detailed description, please refer to deliverable 5.1 - Sensor Analytics and Information Fusion, which provides an in-depth on the current status of all camera analytic functions.

5.2.3.1. People counting (CCTV cameras)

Entrance/exit counter, up to 95% accuracy with adequate lighting in overhead installations.
Can be used to count people through gates to manage venue capacity or get an estimate of how many people are in a specific area of a venue.

5.2.3.2. People counting (Time-of-Flight cameras)

Entrance/exit or small area counting. Up to 99% accuracy. Has active illumination so can be used in a wide range of scenarios.

5.2.3.3. Fighting detection (Processing Node)

Can detect fighting in a range of scenarios. Raises an alert that can be used to direct a guard to diffuse the situation.

5.2.3.4. Object & human detection (Processing Node)

Detect and track humans and other relevant objects. Can be used for detecting the presence of a people in a specific area, or estimating a crowd count. Can be used for detecting e.g. vehicles in restricted areas.

5.2.3.5. Crowd density estimation (Processing Node)

Estimates the density (in terms of number of people) of a crowd. Can be used to manage venue capacity management or detect overcrowding in a specific area (e.g. queues).

5.2.4. Integration of cameras and nodes into the IoT middleware

For those algorithms that can run directly on the cameras, the cameras can communicate directly with the IoT middleware. However, due to the legacy nature of the camera APIs they are unable to natively interact with the middleware so an integration bridge has been developed that translates from the cameras event message format to the RESTful format required by the upper layers in MONICA. This is a simple Python script that can run almost anywhere, but for the purposes of the previous integration demonstrations has been installed on a Raspberry Pi Model 2B on the local network attached to the cameras.

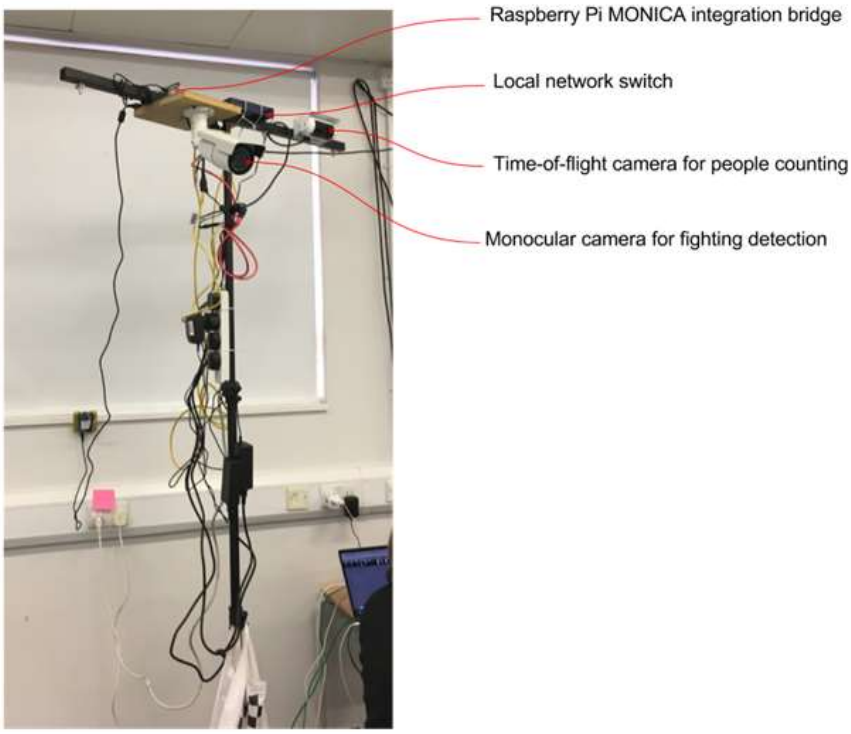


Figure 38: Demonstration of Time-of-flight camera for real-time tracking and counting with MONICA bridge running on a Raspberry Pi on the local network

For those algorithms that are too heavyweight to run directly on the camera and have to run on the processing nodes, the video processing framework has been extended to support a templated notification system. The notification system allows the format of any notification to be specified via a template language, as illustrated in Figure 39.

Figure 39: A templated alert that sends real-time count values from the processing node to the MONICA middleware

In MONICA the actual video streams will not be transmitted in to the MONICA cloud, only metadata that allows local connections to the video feed will be implemented, i.e. the cloud will contain meta data information that

enables connection to the video stream locally. However, snippets of video can be pushed to the MONICA cloud.

The services output, i.e. people counting et c., will create data streams that is pushed up in to MONICA cloud using the SCRAL component.

5.2.5. Future work

The main objectives for the future work with regards IoT middleware integration of cameras and processing nodes is to:

- Add functionality for automatically creating metadata, both for the actual cameras as well as for the recognised security incidents. For instance, the area the camera currently covers.
- Align the content of the data streams with the IoT Resource ontology with regards to identification and metadata.
- Model the services output and connect the outputs to the SCRAL.
- Investigate possibilities of creating generic links that can be used for accessing existing surveillance video storage systems at the pilot sites. This can be very useful for improving the recognition algorithms as well as for post event analysis.

5.3. Microphones / Sound level meters

A sound level meter (SLM) is a device consisting of a microphone, data storage, processing and a communication interface. The sound level meter can (of course) calculate the sound level, but also record and transmit the raw sound signal and act like a microphone. In the rest of this chapter the term sound level meter will be used to also cover microphone usage.

Deliverable D4.4 contains a more thorough description of the sound level meter and the services it enables.

5.3.1. Technology overview

The sound level meter is an autonomous device, which does not depend on an infrastructure. It will basically record the sound, potentially do data analysis and data reduction, store and transfer the results.

The analysis in the device is mainly the common standardised sound level calculations with various weighting filters (like A-weighting) and averaging intervals.

The sound level meter incorporates a GPS receiver which enables information of the location of the sound level meter to be read, but also to time stamp data from the sound level meter, which enables time alignment of data from several devices.

The sound level meter can connect wirelessly to an SLM Gateway using WiFi or 3G/4G.

Communication and data form the sound level meter to the MONICA cloud goes through the SLM Gateway by using a REST interface implemented in the SLM Gateway.

The SLM Gateway contains processing capabilities, which enables services that uses input data from several sound level meters and which can further reduce the data, before transferred to the MONICA cloud.

5.3.2. Infrastructure for microphones/sound level meters

The sound level meters are battery driven and should be able to run without external power for 8 hours. External power will have to be provided if longer operation is needed.

The sound level meter communicates with the SLM Gateway using either WiFi or 3G/4G, which means either should be available for the meters to operate. The needed bandwidth depends on the mode of operation. Most demanding scenarios are those which require raw microphone signals (around 2 Mbit/sec).

5.3.3. Services enabled by microphones/sound level meters

5.3.3.1. Sound heat map

Combining the sound levels recorded by the microphones with a computational sound propagation model, the signals from the PA system used in the concert and the weather conditions measured (wind, temperature and

humidity), the sound field produced for such a PA system is calculated. This map covers the area of the venue and the surrounding neighbourhoods, giving an overview of the levels during the concert.

5.3.3.2. Sound levels

Historical sound levels measured at the location of the sound level meter(s) can be retrieved within a specified time interval.

5.3.3.3. Contribution analysis

Measuring the sound level in the vicinity of an outdoor event like a concert will also include the sound from other sources, like cars driving by. By placing one or more sound level meters close to the concert, this service enables that the contribution from that concert can be separated at any other location where a sound level meter is placed, and the sound level caused by the concert at that location estimated.

5.3.3.4. Sound event detection

The SLM Gateway can be trained to automatically detect specific events from the recorded sound, like gun shots, screaming people and accelerating cars.

Information about event type, location and time will be sent to the MONICA cloud.

5.3.3.5. Integration of microphones into the IoT middleware

The sound level meters are interfaced into the MONICA cloud through the REST interface on the SLM Gateway. For details please see Deliverable D4.4.

5.3.5. Future work

The IoT enabled sound level meters are being developed during the MONICA project. Two prototype versions will be produced, where the first version will have reduced functionality and a more “bulky” configuration.

In between the first version (D4.4) and the second version (D4.5) more features will be enabled as they are finalised.

The main objectives for the future work with regards IoT Middleware integration of sound level meters is to:

- Integration of the SLM Gateway to the SCRAL using the SLM Gateway API
- Add functionality for automatically creating metadata, both for the actual sound level meters as well as for the recognised sound events. For instance, the position of the microphone.
- Investigate if we should mirror the historical data in the sound level meter in the MONICA cloud or if we can use the historical data through the sound level meter.
- Align the content of the data streams with the IoT Resource ontology with regards to identification and metadata.
- Model the services output and connect the outputs to the SCRAL so it will become available as data streams in the MONICA cloud.

5.4. Environmental sensors

5.4.1. Technology overview

In many of the MONICA pilots we expect that environmental sensors will be used. For instance, the wind speed is useful for the sound propagation models. We expect some of this data to be available from open data sources but depending on the needs of granularity and update rates there will be need for MONICA to deploy its own sensors or to reuse existing sensors on the pilot sites.

For the locally deployed environmental sensors a proof of concept installation was made as part of the M9 demonstration. The main purpose of the proof of concept was to showcase how IoT devices can easily be integrated with the MONICA backend cloud. At this stage we have not selected which sensors to use in actual deployments. The selection will depend on which pilots will require locally deployed MONICA sensors.

5.4.2. Infrastructure for environmental sensors

This is not decided yet since the sensor devices have not been chosen. In the proof of concept, a local wireless MESH network was used, see 5.4.4. In any case we do not believe that the infrastructure requirements are high because of the low bandwidth needed for transmission of data.

5.4.3. Services enabled by environmental sensors

5.4.3.1. Environmental data

Provides environmental data such as wind speed, temperature, humidity etc.

5.4.4. Integration of environmental sensors into the IoT middleware

A proof of concept demo was developed for the M9 demonstration to showcase how sensor nodes could easily be integrated with the MONICA cloud standard interfaces – namely the SCRAL middleware and the One M2M gateway – using open standard IoT technologies, protocols and software.

The basic idea is as follows: a sensor node makes periodic measurements (e.g. of environmental conditions) and sends acquired sensor data to the backend, which provides a web service with a RESTful API to process and store the measurement data. The IoT devices send their JSON encoded sensor data via CoAP [5]to the field gateway which translates from CoAP to HTTP and relays the data to the backend. Note, that CoAP is specifically designed to support RESTful services in constrained networks, i.e., it is compatible with HTTP but does not support all its features and is based on UDP instead of TCP. Hence, its rather simple to translate from CoAP to HTTP when using RESTful APIs.

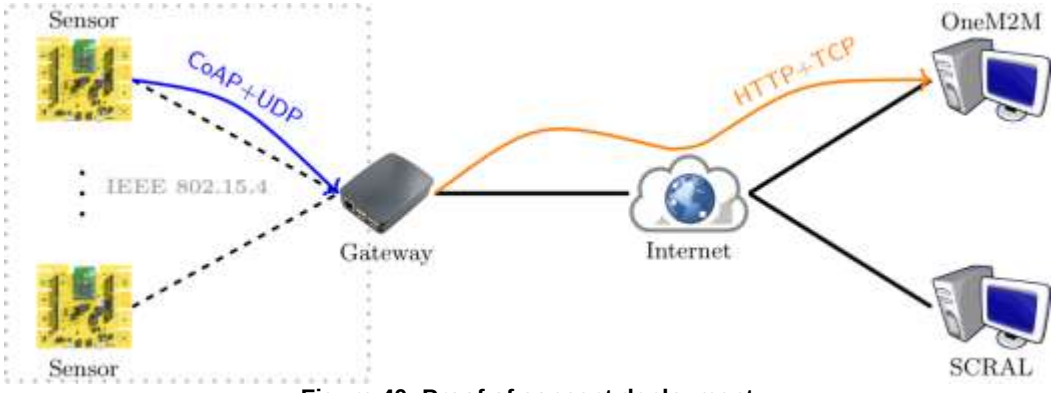


Figure 40: Proof of concept deployment

The demo consisted of three major components, see Figure 40:

- IoT sensor nodes
- field gateway
- MONICA cloud

For the sensor nodes, we used the IoT development board PhyNode from PHYTEC as our hardware platform and an application based on the Open Source IoT operating system RIOT[6]. For network connectivity, each sensor node is equipped with a low power radio transceiver that supports IEEE 802.15.4.
The field gateway is a Raspberry Pi 3 with a IEEE 802.15.4 transceiver to communicate with the sensor nodes as well as (W)LAN for Internet connectivity. On the software side we use standard Raspbian Linux and a CoAP-to-HTTP proxy implementation to relay data from sensor nodes to the backend.
The MONICA backend is provided either by the SCRAL middleware or a OneM2M gateway, both providing a RESTful web service API. These cloud services are deployed on the Internet and are accessible through well-known URLs.
The sensor data is encoded as JSON using a custom syntax that is based on SenML, a sample sensor data record for air pressure looks like the following:

[5] The Constrained Application Protocol (CoAP) https://tools.ietf.org/html/rfc7252
[6] RIOT-OS https://riot-os.org

{"bn":160,"bl":[53.56,10.02,1.00],"n":"P","v":1016.57,"u":"hPa"}

With the following attributes: node ID (bn), its location (bl), the type of sensor (P = pressure), the value (v), and unit (u).

The proof of concept contains several sensor nodes that send data with a RESTful POST request using CoAP+UDP to the Raspberry PI (field gateway) which relays the data as HTTP+TCP to the backend. It is worth noting that each sensor node as an IPv6 address assigned and the field gateway merely acts as a network interconnect to bridge network traffic from IEEE 802.15.4 to the Internet (LAN) and translate from CoAP to HTTP. Hence, it is possible to send data end-to-end directly between a RIOT-based sensor node and the MONICA cloud, with a very simple network gateway in the middle.

5.4.5. Future work

The main objectives for the future work with regards IoT Middleware integration of environmental sensors is to:

- Investigate which pilots will require MONICA deployed environmental sensors
  - What should be monitored, i.e. wind speed, wind direction et c.
- For existing environmental sensors at pilot sites need to be investigated if they can be integrated.
  - Involves interfacing the SCRAL and creating the necessary meta data.

5.5. Blimps

5.5.1. Technology overview

Why does MONICA need aerial vision? The higher point of view has been a predominant factor in most of the human activities, from archaeology to warfare; things and behaviours that apparently did not have a precise meaning on the ground were suddenly clear to the observer when he could climb uphill or enter into a flying machine. That is why one of the first activities performed with airplanes or Montgolfier was aerial photography.

Figure 41: First surviving aerial photograph, titled 'Boston, as the Eagle and the Wild Goose See It.'[7]

Nowadays, aerial video and photography is implemented in numerous fields such as movie industries, commercials, power plants, crops and pipelines monitoring just to name a few. It should not surprise then, that within the MONICA project an aeronautic component was foreseen to engage the different fields related to the project aim. An airship floating over events could simultaneously record audio, provide a global vision of the area and be used as a beacon for other devices.

As the number of Unmanned Aerial Vehicles (UAV) augmented in the last years (available both for recreational and working purposes), the European Aviation Safety Agency (EASA) was requested in 2015 to come up with a Prototype Regulation that should help preparing the formal rule-making process leading toward a unified regulation throughout Europe. This Prototype Regulation still needs to be continued and revisited by stakeholders and other entities, therefore EASA and the European Community currently demand to local aviation authorities to write and issue national regulations.

What was just mentioned is clearly stated in the Remotely Piloted Aerial Vehicles Regulation, Issue No. 2 dated 16 July 2015, revision 2 Dated 22 December 2016. This extract has been emanated by ENAC which is the Italian Civil Aviation Authority. In fact, one could read at Article 2, comma 2, the following:

"Pursuant to the Regulation of the European Parliament and of the Council (EC) No 216/2008, RPAS of operating take-off mass not exceeding 150 kg and those designed or modified for research, experimental or scientific purposes pertain to ENAC competence."

DigiSky operates in the Italian territory and therefore will try to develop an aircraft which best fits the Italian regulations. To not overload this section with unnecessary regulation details, let's sum up the requirements that led DigiSky to choose in favour of a blimp type solution:

ENAC considers RPAS (Remotely Piloted Aircraft System) as a system consisting of an aerial vehicle (remotely piloted aircraft) without persons on board, not used for recreation and sports, and the related components necessary for the command and control (remote ground pilot station) by a remote pilot. In other words, an RPA is an aircraft system that have on board equipment which enables its autonomous flight. This equipment is normally constituted by a GPS, an inertial platform and a propulsion system (propellers or engines).

7 Authors James Wallace Black and Samuel Archer King on October 13, 1860, it depicts Boston from a height of 630m

In any case, to overfly (with RPAS) gatherings of people during parades, sports events or different forms of entertainment or anyhow areas where there is an unusual concentration of people, is prohibited. (Article 10, comma 7).

Unfortunately, all the MONICA events take place in critical areas, that means areas with gatherings of people.

The following RPAS are not subject to the provisions of the above mentioned regulation (Article 2, comma 3):

- *Balloons used for scientific observations or tethered balloons*
- *RPAS operating inside indoor space*

This last article explains why DigiSky decided to use tethered blimps for MONICA events. Since tethered blimp are allowed over critical areas they represent a valid solution for an aerial point of view. At least over Italian territories.

Italian Regulation over Tethered Balloons

As the reader should have understood from the previous chapter, tethered balloons or blimps are not subject to RPAS regulation, therefore it is much easier to fly them over critical scenarios. However it does not mean that the aeronautic authority permission to let them fly is not needed. Hereafter some extracts of the air traffic management memo (ATM - 05A, 23/07/2013) are commented. This memo regards events and special activity affecting air traffic management over Italian territories and defines procedures to require and emit a NOTAM (Notice to Air Man).

As an example, fireworks, night lasers and Chinese lanterns release are activities that do require the permission of aviation authority.

Figure 42: Chinese lanterns and fireworks are two examples of activity that do need aviation authority permission

Those activity cited at Art.3, comma 2g (raising of tethered balloon and blimps) can take place only if they do not enter the "airport respect surfaces" and they do not interfere with instrumental procedures for landing, abort landing, take off and circling. […] If the raising height of the airship is smaller than 40 meters from the surrounding ground than the NOTAM is not necessary.

From this article it is easy to derive why DigiSky fixes the maximum height from the ground to 40 meter. In this way in the Italian territory the regulation is respected involving less bureaucratic effort, also considering that obtaining a NOTAM may be a complex and long procedure.

Technical Parameters

DigiSky does not produce blimps and balloons on its own, therefore the first blimp DigiSky acquired was the result of a scouting between European producers. Moreover, the reader should keep in mind that the simple balloon design has been discarded by DigiSky due to the following considerations:

(1) Is not stable in windy conditions
(2) Does not allow a radio control (RC) configuration, therefore the payload should have a different design between RC and tethered version, lowering the reusability of the payload frame.

Figure 43: The DigiSky blimp prototype floating over the Aeritalia Airport, Turin (Ground Operators not visible in the picture are controlling the Blimp altitude through cables connected to Ground)

The prototype blimp has the following characteristics:

Table 4: Some of the blimp prototype technical parameters

Field	Condition	Value	Measurement unit
Length	inflated	5	meter
Width (max circumference)	inflated	1.8	meter
Weight	Deflated	4.5	kilograms
Volume	Inflated	9	Cubic Meters
Theoretical Payload	Sea Level	4.5	Kilograms
Real Overall Payload for partners	Turin Pilot	1	Kilogram

5.5.2. Infrastructure for blimps

The blimp itself requires a set of handling and deployment systems that will be described hereafter:

Inflating System – The blimp needs to be inflated with Helium FlyGas[8]. The average price in Italy for the gas is around 25 Euro/cubic meter before taxes, that is why a smaller airship may be preferred. On the market different kinds of Helium tanks are available, the following refers to an inside pressure of 200 bar:

- 27 liters (6 cubic meter)
- 40 liters (8 cubic meter)
- 50 liters (10 cubic meter)

8 The term FlyGas refers to Helium with a lower purity degree used in aerial application, while the super pure one is necessary in other applications, such as diving or in hospital procedures.

Therefore our prototype blimp could be inflated with a 50 liter tank. This in turn means that in a venue area, this tank (or more) must be allowed and carried around with an appropriate cart.

Deployment - Recover Area – This prototype blimp must be inflated/deflated in a clear area, protected from the ground dust and particles using a 4x6 meters cloth, to prevent an any accidental damage to the envelope.

Deflating System – The blimp requires a long time to deflate once the escape gas valve are opened, that is why a deflating system (vacuum) has to be connected to the valve to speed up the process. The vacuum should at least aspire draw out 1 cubic meter of gas in a minute. In order to power the vacuum, an appropriate power outlet must be reachable from the deflating-recover area.

Parking slot – When not used (for overnight events for example) the blimp, if helium inflated, has to be kept in a closed, indoor area, in order to prevent damage to its envelope or its ground connection from atmospheric agents or unintentional/intentional human damages.

Deployment System – Once inflated, ground operators should board the payload in the payload bay and proceed with the mission designed for the event. Each operator has to be connected to the blimp with a rope that cannot exceed the 38 meters of length and has to be instructed by DigiSky about the airship capabilities and handling manoeuvres.

5.5.3. Services enabled by blimps

The blimp has a payload area that enables the boarding of partner's sensors. Therefore, if cameras or microphones that meet the payload dimension, weight and power requirements are furnished, DigiSky will build a physical interface to board them on the payload area. Any connection to the MONICA cloud of these sensors has to be done at a device level and will not be provided by the blimp itself.

DigiSky will soon test the Image Stability achievable with a GoPro Hero 3 Mounted on a Gimbal. The data recorded by the camera will not be instantly broadcasted to ground, but instead will be recorded on a SD card and analysed once the mission will be concluded. This simple service, if demonstrated to be accurate enough, could help the event organizer to have images or video recorded from a 38 meters elevation to use for events promotion or other purposes (keep in mind that over large gatherings of people, even professional multi-copters with cameras would not be allowed).

5.5.4. Integration of blimps into the IoT middleware

The blimp will feature a Fly Data Log that has a WiFi port which could be used to connect to the IoT Middleware in order to provide information about the blimp location, altitude and other parameters.

5.5.5. Future work

DigiSky is leading the setup of the airship in two ways:

- The first one concerns the above mentioned tethered balloons which still requires an intensive test phase in order to comprehend the stability of the system with relation to the altitude, the wind condition and the payload boarded.
- The second one (which will be considered in a more advanced phase of the project) relies on the development of a RPAS balloon or multi-copter system that will be tested in the Arena that DigiSky promoted for the project. Since Aeronautical regulation over RPAS may be revisited by European Authorities and RPAS may be allowed in the future over critical areas, it could be useful to have an already tested version of these airships.

The main objectives for the future work with regards IoT middleware integration of the blimp is to:

- Investigate what data we can retrieve from the WiFi port.
- Investigate how the hosted devices will connect with MONICA.
- Enable the selected devices in the MONICA IoT middleware

6. Conclusion

In this report both the IoT wearables as well as the IoT enabled devices as they will be used in MONICA have been extensively discussed. The technical capabilities, infrastructure requirements and MONICA interoperability have been treated in detail. Besides operation and technical features, the usage of these devices in the context of the MONICA use cases has been discussed as well. A first, small scale, pilot of the so called staff wearable at the Kappa FuturFestival has been presented. A follow up of this deliverable will report on more experience with all these IoT devices at events taking place in 2018.

7. List of figures and tables

7.1. Figures

7.2. Tables

8. References

(H. Cox. 1964) H. Cox, “On the estimation of state variables and parameters for noisy dynamic systems,” IEEE Trans. Automat. Control, vol. 9, no. 1, pp. 5–12, Jan. 1964.

(W. Figel et al. 1969) W. G. Figel, N. H. Shepherd, and W. F. Trammel, Vehicle location by a signal attenuation method, IEEE Transactions on Vehicular Technology, vol. 18, no. 3, pp. 105–109, 1969.

(C. Cook et al. 1970) C. E. Cook and M. Bernfeld, Radar Signals: An Introduction to Theory and Applications, New York: Academic, 1970.

(S. Kirkpatrick et al. 1983) S. Kirkpatrick, C. D. Gelatt, Jr., M. P. Vecchi, “Optimisation by Simulated Annealing”, Science, New Series, Vol. 220, No. 4598. (May 13, 1983), pp. 671-680.

(H. Hashemi. 1993) H. Hashemi, The indoor radio propagation channel, In Proceedings of the IEEE, pp. 81(7):943-968, 1993.

(H.V.Poor. 1994) H. V. Poor, An Introduction to Signal Detection and Estimation, 2nd ed. New York: Springer-Verlag, 1994.

(A. Doucet. 1998) A. Doucet, On sequential Monte Carlo Methods for Bayesian Filtering, University of Cambridge, Tech. Rep. CB2 1PZ Cambridge,UK, 1998.

(M.S et al. 2002) M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking, IEEE Trans. on Signal Processing, vol. 50, no. 2, pp. 174–188, Feb. 2002.

(Patwari et al. 2003) N. Patwari, A. O. Hero, M. Perkins, N. S. Correal, and R. J. O'Dea. Relative location estimation in wireless sensor networks. Signal Processing, IEEE Transactions on, 51(8):2137-2148, 2003.

(Y. Qi et al. 2003) Y. Qi and H. Kobayashi. On relation among time delay and signal strength based geolocation methods. 2003.

(N. Patwari et al. 2003) N. Patwari, A. O. Hero, M. Perkins, N. S. Correal and R. J. O'Dea, Relative location estimation in wireless sensor networks, in IEEE Transactions on Signal Processing, vol. 51, no. 8, pp. 2137-2148, 2003.

(G. Welch et al. 2006) G. Welch and G. Bishop, An Introduction to the Kalman Filter, University of North Carolina at Chapel Hill, Tech. Rep. Chapel Hill, NC 27599-3175, 2006.

(S. Rao. 2007) S. Rao, Estimating the ZigBee transmission-range ISM band, EDN, vol. 52, no. 11, pp. 67–74, 2007.

(J. A. Ting et al. 2007) J. A. Ting, E. Theodorou and S. Schaal, A Kalman filter for robust outlier detection, 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, 2007, pp. 1514-1519.

(S. Gezici et al. 2008) S. Gezici, I. Guvenc and Z. Sahinoglu, On the Performance of Linear Least-Squares Estimation in Wireless Positioning Systems, 2008 IEEE International Conference on Communications, Beijing, 2008, pp. 4203-4208.

doi: 10.1109/ICC.2008.789

(Caceres et al. 2009) Mauricio A. Caceres, Francesco Sottile, and Maurizio A. Spirito, Adaptative Location Tracking by Kalman Filter in Wireless Sensor Networks, Proc. IEEE International Conferences On Wireless and Mobile Computing, Networking and Communications, pp. 123-128, 2009.

(F. Sottile et al. 2011) F. Sottile, H. Wymeersch, M. A. Caceres and M. A. Spirito, Hybrid GNSS-Terrestrial Cooperative Positioning Based on Particle Filter, 2011 IEEE Global Telecommunications Conference - GLOBECOM 2011, Houston, TX, USA, 2011, pp. 1-5.

(Chruszczyk et al. 2016) Chruszczyk. Ł, Zając. A. Comparison of Indoor/Outdoor, RSSI-Based Positioning Using 433, 868 or 2400 MHz ISM Bands. International Journal of Electronics and Telecommunications, 62(4), pp. 395-399. 2017.

(Chruszczyk. 2017) Chruszczyk, Ł. Statistical Analysis of Indoor RSSI Read-outs for 433 MHz, 868 MHz, 2.4 GHz and 5 GHz ISM Bands. International Journal of Electronics and Telecommunications, 63(1), pp. 33-38. 2017.

(Shmaliy et al. 2017) Y. S. Shmaliy, S. Zhao and C. K. Ahn, "Unbiased Finite Impulse Response Filtering: An Iterative Alternative to Kalman Filtering Ignoring Noise and Initial Conditions," in IEEE Control Systems, vol. 37, no. 5, pp. 70-89, Oct. 2017.

(Dexels et al 2017) Dexels, ISMB, Kappa FuturFestival UWB tracking test (6-8 July 2017), Internal report, 2017.

(VCA, 2017) VCA Technology Ltd, Company Website: Cameras with On-board Analytics, Online. URL http://www.vcatechnology.com/vca-cameras [Accessed 6 December 2017].