Exhaustion and reduced vigilance may also be seen via gait analysis. This is taken into account by a separate sub-model for "**Gait consistency**", weighed with 10 %.

Activity (and thus the previously calculated "**Activity index**") is a notable input factor for "Vigilance" [71].

Outputs

"Vigilance" is represented by a decimal number with 0 (not alert) and 10 (highly alert). The objective is to reach a Vigilance index (VI) of 10. Anxiety or other detrimental forms of vigilance are not considered. The VI is calculated every minute.

Summary and calculation formula

Quantity		Source	Range	Unit	Weight
t_{sleep}	Time asleep	Fitbit	0 / 100 / 400 / 500	[h]	20 %
t_{awake}	Time awake between sleep cycles		0 / 20 / 240 / 400		10 %
a	Arousal	Camera on face	0 / 0 / 1 / 1	[-]	40 %
GC	Gait consistency	Gait analysis	0 / 0 / 1 / 1	[-]	10 %
AI	Activity index	AI calculation	0 / 0 / 10 / 10	[-]	20 %

Table 13: Overview of input quantities for Vigilance index (VI); the quantities, their data source, validity ranges (for their interpretation, see Figure 24), units and impacts for the VI are displayed.

$$VI = 0.0050 \cdot t_{sleep} - 0.0045 \cdot t_{awake} + 1.0909 + 4 \cdot a + GC + 0.2000 \cdot AI$$

Time asleep | Time awake | Arousal | Gait consistency | Activity

4.4.4 Stress

Motivation

Lazarus defines: *"Stress arises when individuals perceive that they cannot adequately cope with the demands being made on them or with threats to their well-being"* [34]. Stress is commonplace in today's life; exceeding arousal is linked to cardiovascular disease, cancer, arthritis, and major depression [35]. Stress and an elevated "Resting heart rate" (measured by the fitness tracker) are linked [36] and a high resting heart rate itself correlates with many kinds of mortality [37][39].

Inputs

Stress is associated with sleep disorders in two ways: stress provokes sleep disturbances, and disturbed sleep provokes stress and increases risk e.g. for cardiovascular disease [40]. Both "Hours of sleep" (about 6-9 hours per night [40]) and "Sleep quality" have to be considered [33]. So **time asleep** and **time awake between sleep cycles**, measured by the Fitbit Charge HR, are taken into account with 10 % and 5 %, respectively.

In its negative form, **arousal** is a synonym for stress and is weighed with 20 %. In a similar manner, negative **emotional valence** (anxiety, sadness etc.) may be seen as a sign for stress (impact 10 %). Beyond the former two quantities, the camera measures **pain** on the patient's face. As another natural indicator it is considered to have a strong influence (20 %).

We define the **current heart rate** as another input for stress, as it can be seen as a short-term quantity indicating current stress situations. Although this quantity is measured redundantly (by the camera and the Charge HR), we expect its availability and accuracy to be limited and thus assign a rather low impact of 10%.

Stressed people tend to show faster **walking speeds** [72], which we take into account by an impact of 5 %. It is measured by gait analysis.

"**Activity**" reportedly decreases feelings of stress [73], while "**Vigilance**" might be seen as a synonym for stress. Both former calculated quantities are weighed with an impact of 10 %.

Outputs

"Stress" is represented by a decimal number with 0 (calm) and 10 (stressed). The objective is to reach a Stress index (SI) of 0. Cognitive or physical stimulation or other positive forms of stress are not considered. The SI is calculated every minute.

Summary and calculation formula

Quantity		Source	Range	Unit	Weight
t_{sleep}	Time asleep	Fitbit	0 / 100 / 400 / 500	[h]	10 %
t_{awake}	Time awake between sleep cycles		0 / 20 / 240 / 400		5 %
a	Arousal	Camera on face	0 / 0 / 1 / 1	[-]	20 %
emo	Emotional valence		-1 / -1 / 1 / 1	[-]	10 %
P	Pain		0 / 0 / 1 / 1	[-]	20 %
HR	Heart rate	Fitbit / Camera on face	0 / 120 / 180 / 200	[/min]	10 %
v_{walk}	Walking speed	Gait analysis	0 / 0 / 1 / 1	[m/s]	5 %
AI	Activity index	AI calculation	0 / 0 / 10 / 10	[-]	10 %
VI	Vigilance index	VI calculation	0 / 0 / 10 / 10	[-]	10 %

Table 14: Overview of input quantities for Stress index (SI); the quantities, their data source, validity ranges (for their interpretation, see Figure 24), units and impacts for the SI are displayed.

$$SI = \underbrace{-0.0025 \cdot t_{sleep} + 1.2500}_{} + \underbrace{0.0023 \cdot t_{awake} - 0.0455}_{} + \underbrace{2.000 \cdot a}_{} + \underbrace{0.5000 \cdot v_{walk} + 2.000 \cdot P}_{}$$
$$\underbrace{- 0.5000 \cdot emo + 0.5000}_{} \underbrace{- 0.1000 \cdot AI + 1.0000}_{} + \underbrace{0.1000 \cdot VI + 0.0167 \cdot HR - 2.000}_{}$$

4.4.5 Physical indication

Motivation

"Physical indication" is defined as a short-term field and may (in the future, after intensive testing) trigger a warning to caregivers.

Inputs

We define high levels of "**Stress**" combined with "**Vigilance**" as worthy of indicating (impact 20 % and 15 %, respectively). **Arousal** and **pain** may also be triggers for an indication – since these numbers already highly influence stress, their effect is somewhat doubled. Their influences are estimated with 20 % and 25 %, respectively. Many quantities are measured by the camera; this is why an absolutely robust camera system is needed for a PI to function.

A **high current heart rate** as well as an **inconsistent gait** may also be a source of concern (impact 10 % each).

Outputs

"Physical indication" is represented by a decimal number between 0 (no indication) and 1 (physical indication). The objective is to reach a "Physical indication" (PI) of 0.

Summary and calculation formula

Quantity		Source	Range	Unit	Weight
SI	Stress index	SI calculation	0 / 0 / 10 / 10	[-]	20 %
VI	Vigilance index	VI calculation	0 / 0 / 10 / 10	[-]	15 %
a	Arousal	Camera on face	0 / 0 / 1 / 1	[-]	20 %
P	Pain		0 / 0 / 1 / 1	[-]	25 %
HR	Heart rate	Camera on face / Fitbit	0 / 140 / 200 / 200	[/min]	10 %
GC	Gait consistency	Gait analysis	0 / 0 / 1 / 1	[-]	10 %

Table 15: Overview of input quantities for Physical indication (PI); the quantities, their data source, validity ranges (for their interpretation, see Figure 24), units and impacts for the PI are displayed.

$$PI = 0.2000 \cdot SI + 0.1500 \cdot VI + 2.0000 \cdot a + 2.5000 \cdot P + 0.0167 \cdot HR - 2.3333 + GC$$

4.4.6 Emotional balance

Motivation

We define "Emotional balance" as a feeling of personal well-being without considering short-term "Physical indication".

Inputs

Poor "Sleep quality" was significantly correlated with increased physical health complaints and with increased feelings of tension, depression, anger, fatigue, and confusion" [33]. So **time**

asleep and **time awake between sleep cycles**, measured by the Fitbit Charge HR, are taken into account with 15 % and 10 %, respectively.

To have a broad (and thus robust) sub-model, we assume a relation of **gait consistency** with "Emotional valence" (impact 5 %).

Emotional valence is a direct measure for emotional balance and thus is assumed to be of high influence: 20 % impact.

Improvement of cognitive function in older adults [42] and reduction of depressive symptoms [43] serve as examples here. "**Stress**" is detrimental to "Emotional balance" [45], weighed with 20 %.

The positive influence of activity ("**Activity index**") on "Emotional balance" is widely accepted [41]; we assumed an impact of 20 %.

A certain degree of "**Vigilance**" is crucial to "Emotional balance" – 10 % impact.

Outputs

"Emotional balance" is represented by a decimal number with 0 (unstable) and 10 (relaxed). The objective is to reach an Emotional balance (EB) of 10. The EB is calculated every minute.

Summary and calculation formula

Quantity		Source	Range	Unit	Weight
t_{sleep}	Time asleep	Fitbit	0 / 100 / 400 / 500	[h]	15 %
t_{awake}	Time awake between sleep cycles		0 / 20 / 240 / 400	[h]	10 %
GC	Gait consistency	Gait analysis	0 / 0 / 1 / 1	[-]	5 %
emo	Emotional valence	Camera on face	-1 / 0 / 1 / 1	[-]	20 %
SI	Stress index	SI calculation	0 / 0 / 5 / 10	[-]	20 %
AI	Activity index	AI calculation	0 / 0 / 5 / 10	[-]	20 %
VI	Vigilance index	VI calculation	0 / 0 / 5 / 10	[-]	10 %

Table 16: Overview of input quantities for Emotional balance (EB); the quantities, their data source, validity ranges (for their interpretation, see Figure 24), units and impacts for the EB are displayed.

$$EB = 0.0050 \cdot t_{sleep} - 0.5000 - 0.0045 \cdot t_{awake} + 1.0909 + 0.5000 \cdot GC + 2.0000 \cdot emo - 0.4000 \cdot SI + 2.0000 + 0.4000 \cdot AI + 0.2000 \cdot VI$$

Chapter 5
Collaborative Localisation

Accurate, robust and reliable position estimation and tracking is essential the navigation of smart assistive devices such as the *FriWalk*. Compared with the smart walker developed in the project DALi (Devices for Assisted Living), a *FriWalk* is able not only to self-estimate its location using its own sensor data, but also to exploit the position information shared by other *FriWalks* detected in the same area. In the specific context of ACANTO, this approach, generally known in robotics as *collaborative*, *cooperative* or *synergic* localisation, can greatly increase localisation accuracy, reliability and robustness, as it will be shown in the rest of this Chapter. Of course, the availability of a joint, refined vision of the position of different *FriWalks* (also referred to as *agents* in the following) can potentially enhance the semantic interpretation of user status and activities based on the USM. Nonetheless, the flavour of the proposed collaborative localisation is just to increase the accuracy of the local estimates and not to share a common view of the *FriWalks* group as a single entity, i.e. even with the distributed approach each agent keeps track of its own position only and exploits the relative position measurements to reduce its uncertainty. This is mainly due to limitations in the communication bandwidth, limitations on the dimensionality of the estimation problem as well as to constraints on the scalability of the solution. Notice that simply sharing and collecting the position values estimated locally by multiple *FriWalks* using the existing communication infrastructure between them is generally sufficient to support the implementation of the USM-based monitoring services described in Chapter 4, hence this is the solution that is applied for ACANTO.

In this respect, an early study on collaborative localisation is presented in [53]. In [54] the authors envision a fully wireless synergic localisation system based on the potential ability of clusters of 4G mobile devices to measure their reciprocal distances through a hybrid time of arrival/angle of arrival technique. The case of collaborative localisation of wireless mobile platforms has been also described in [55], where the so-called parallel projection method was used to improve localisation accuracy in non-line-of-sight (NLOS) conditions. In [56] a Markov-based probabilistic method was proposed to refine each robot's belief about its own position as soon as other robots are detected. An alternative Markovian approach is adopted in [57]. In this case, first the egocentric measurement data are fused locally to create a Markov chain of robot pose estimates. Then, both inter-robot measurement data and state estimates are transferred to a central server, where localisation is refined by minimizing the mean square error of agents' positions. A spring model was used in [58] to reduce the pose estimation uncertainty associated with distance measures obtained using Wi-Fi and Bluetooth RSS data. An RGB-D camera was instead used to observe other agents and measure the relative position with respect to them [59].
A classic and well-established algorithm for collaborative localisation is the so-called interlaced extended Kalman filter (IEKF) [60]. The IEKF is inherently distributed, computationally acceptable and easy to implement. However, this solution, which was also adapted to the ACANTO case and tested as described in D3.1.1 [61], may suffer from some estimation accuracy problems when the rate of the relative position measurements between a *FriWalk* and its neighbours tends to prevail over absolute position and orientation measurements, which leads to non-negligible estimation cross-covariances between the estimates performed by different *FriWalks*. This issue, which was thoroughly investigated in [62], can be tackled by sharing and exchanging the cross-covariance matrices of all agents that are in the same area at a given time even if not all of them detect each other. As a result, a modified distributed algorithm for collaborative localisation was implemented and characterized, as described in the following.

5.1 Sensing technologies and algorithm description

In order to solve the problem of collaborative localisation of a team of *FriWalks*, the following assumptions will be considered in the following:

1. N agents can move freely in the same room. The dynamic of each *FriWalk* does not depend on any other agent, the only constraint to motion being collision avoidance.

2. The position of each agent i (with i = 1,..., N) at time kT_s (T_s being the sampling period used to discretize the system) is represented by vector $p_k^i = \left[x_k^i, y_k^i, \theta_k^i\right]^T$, where x_k^i and y_k^i are the agent's planar coordinates in the chosen reference frame, while θ_k^i is the angle between the longitudinal axis of the robot and the x−axis X_W of the reference frame. In practice, the kinematic state of a *FriWalk* may include two additional state variables μ_k^i, δ_k^i which represent the systematic relative linear and angular velocity offsets due to imperfect odometry. Further details on this aspect will be reported in Section 5.1.1.
3. Each agent is able to estimate its own state autonomously (namely without the help of other agents) by fusing odometry data with absolute position and heading measures obtained from QR codes used as landmarks. Such landmarks are placed on the floor at a distance d from one another over a regular *triangular* grid in such a way that at most one of them can be detected by a plain monocular camera directed towards ground.
4. Besides the sensors used by each robot for its own local state estimation, every agent is equipped with a front RGB-D camera able to recognize and to measure the relative position between the robot's camera and any other agents located within its detection range.
5. All agents can share the information about their state and the respective covariance matrices through radio transceivers ensuring high-rate and low-latency communication.

In the following, $\langle W \rangle = (X_W, Y_W, Z_W)$ will denote the fixed reference frame in which the team of *FriWalks* is supposed to be localized. In practice, $\langle W \rangle$ can be set on the map of the environment where the FriWalks are deployed.

5.1.1 Models description

As briefly explained above, three types of sensing technologies are used for *FriWalk* localisation. Two encoders mounted on the rear wheels of each agent are used to estimate position and heading variations from a supposedly known initial location. Visual odometry has not been included (for now) in the FriWalk to speed-up the times of development of the prototype. A standard camera sporadically detects the QR codes stuck on the floor at known locations to update each agent's state about its own absolute position and orientation in the reference frame $\langle W \rangle$. Finally, a front RGB-D camera is used to detect other possible *FriWalks* in the surroundings and to estimate the relative position from them. These measures, along with the state collected from such agents and the corresponding covariance matrices, can be used to achieve collaborative localisation, as explained in Section 5.1.2.

Odometry: Odometry relies on two AMT102 encoders installed on the rear wheels of each agent. In the current implementation the *FriWalk* wheels radius is r= 0.15 m and the rear wheel axle length is t=0.685 m. The encoders data can be regarded as inputs to an augmented unicycle process model. The discretized version of this model for agent i (with $i = 1,\ldots,N$) is defined as follows

$$s_{k+1}^i = f\left(s_k^i, \widehat{\Delta\Phi_k^i}\right) = s_k^i + f_\phi\left(s_k^i\right)\widehat{\Delta\Phi_k^i} \tag{5.1}$$

where

$$f_\phi(s) = \begin{bmatrix} \frac{r}{2}(1+\mu)\cos\theta & \frac{r}{2}(1+\mu)\cos\theta \\ \frac{r}{2}(1+\mu)\sin\theta & \frac{r}{2}(1+\mu)\sin\theta \\ \frac{r}{t}(1+\delta) & -\frac{r}{t}(1+\delta) \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{5.2}$$

$\widehat{\Delta\Phi_k^i} = \left[\Delta\Phi^i{}_{r,k} + \epsilon_{r,k}^i, \Delta\Phi^i{}_{l,k} + \epsilon_{l,k}^i\right]^T$ is the input vector containing the displacements measured by the right and left wheels encoders ($\Delta\Phi^i{}_{r,k}$ and $\Delta\Phi^i{}_{l,k}$ being the real displacements at time kT_s $\epsilon_{r,k}^i$ and $\epsilon_{l,k}^i$ being the respective random uncertainty contributions) and $s_k^i = [x_k^i, y_k^i, \theta_k^i, \mu_k^i, \delta_k^i]$ is vector of the state variables of the ith agent. This vector includes the coordinates (x_k^i, y_k^i) of

the rear axle mid-point chosen as a reference for position, the heading angle $\theta_k^i$with respect to X_W and finally the systematic relative offsets μ_k^i, δ_k^i affecting the linear and angular velocities, respectively, of the robot in the chosen reference point. It is worth noticing that μ_k^i, δ_k^i are not present in a standard unicycle model. However, the 5-state variable model is to be preferred to the classic one whenever the drift caused by encoders is significant and its mean value has to be properly estimated and compensated.

In order to evaluate if such drift terms are indeed relevant, as well as to estimate the variance of encoders data, several experiments were performed in the laboratories of the University of Trento. In particular, the actual position (namely the ground truth) of a *FriWalk* in an open room was measured with sub-centimetre accuracy using a motion capture system Optitrack PRIME 13[8] equipped with 14 calibrated cameras. The *FriWalk* was moved repeatedly along straight-line, rectangular and finally random trajectories. The mean value and the standard deviation of encoders data were finally estimated from the histograms of the differences between the wheel angular displacements measured by the encoders in every sampling period and the corresponding values collected by the motion capture system. The trends of the mean value and of the standard deviation as a function of the actual angular displacement in one sampling period are shown in Figure 25. The results of a linear least-squares fitting show that i) the differences between left and right wheels are negligible, and ii) both mean and standard deviation values grow quite linearly with wheels angular speed. Therefore, the use of a 5-state variables model is perfectly justified in the case at hand.

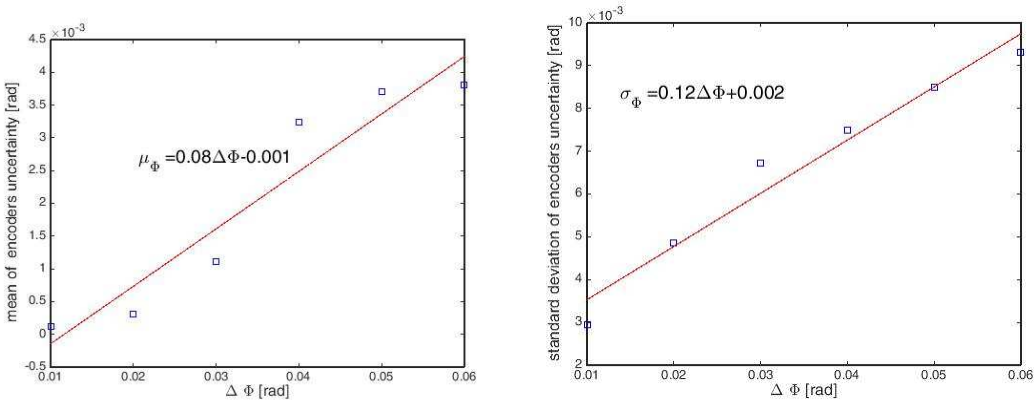

Figure 25: Mean value (on the left) and standard deviation (on the right) of encoders uncertainty.

Front camera and QR code deployment: As shortly explained at the beginning of this Section, the role of the front camera is to detect one of the QR codes stuck on the floor, as shown in Figure 26. To this purpose, the camera has to be slightly oriented towards the ground in order to ensure a front reading range between about 1 m and 2 m.

As known, a QR code is a square image containing an encoded binary matrix, which can store different kinds of data (e.g. numerical, alphanumerical or bytes). In the case considered, a QR code stores only an integer number q, which is univocally associated with its planar coordinates (x_q, y_q) and its orientation angle θ_q in in the reference frame $\langle W \rangle$. This approach is very flexible, since the table associating each QR code number to (x_q, y_q, θ_q) can be easily changed and adapted to different environments, without reprinting the QR codes. Also, in this way just low-density numeric-only codes can be used. The choice of using low-density QR codes increases the ability to detect them at larger distances. In the case considered, version-1 QR codes with just 21×21 black-and-white cells and a type L (i.e. low-level) Reed-Solomon error correction coding

[8] http://optitrack.com/products/prime-13/

(ECC) are adopted. All QR codes were generated by an online tool[9] according to the ISO/IEC Standard 18004:2006 and were printed on regular A4 paper sheets with a resolution of 600 dpi.

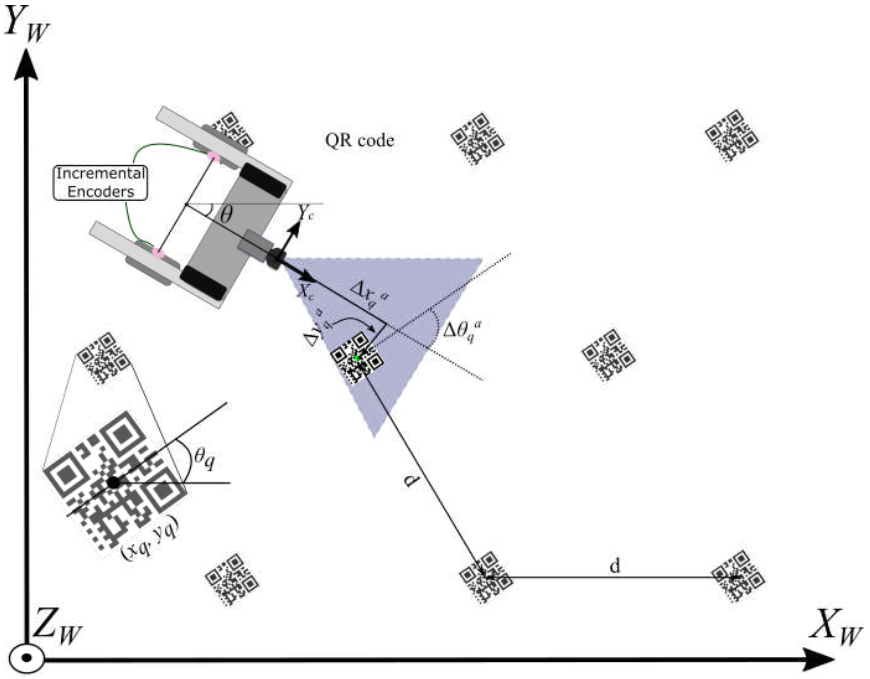

Figure 26: QR code detection system based on a front camera. QR codes are deployed over a regular triangular grid.

As a rule of thumb, QR code size should be one order of magnitude smaller than the scanning range. Also, their size should be proportional to a data density factor given by the ratio between the number of columns (or rows) of the chosen QR code type (i.e. 21 in the case considered) and the number of columns (or rows) of a standard version-2 code (i.e. 25). Since the average scanning range of the chosen camera is about 1.8 m, the QR code size was set equal to 15×15 cm. QR code detection relies on the open-source Zbar[10] library. QR code landmark recognition is instead implemented in C++ using the primitives of the OpenCV[11] library. Both applications run on the Intel Nuc Mini-PC on board of the *FriWalk*.

In general, the QR code placement strategy strongly depends on the geometry of the chosen environment. If no specific constraints exist, a reasonable approach is to deploy the QR codes over a regular grid. In theory, only three periodic, monohedral and regular tiling patterns can be designed over the plane, i.e. equilateral triangles, squares and hexagons. Among them, the triangular one is reasonably easy to deploy. Moreover, an optimal placement strategy has been found in this case [63]. Such a strategy is described in Section 5.2.

Suppose that at time kT_s agent a detects QR code q. Consider that in general $a \neq i$ since not necessarily all *FriWalks* detect a QRcode at any time. With reference to Figure 26, let Δx_q^a be and, Δy_q^a the measured distances between the camera and the detected QR code in the camera frame. Also, let $\Delta\theta_q^a$ be the measured angle difference between the camera optical axis and θ_q. The values of such quantities can be measured using standard image processing algorithms, e.g. based on homography [64]. If the points of the landmark to detect are coplanar (like in the case of the QR codes) and if the dimensions of such codes are known a-priori, the homography-based

[9] http://goqr.me/
[10] http://zbar.sourceforge.net/
[11] http://opencv.org/

techniques ensure a robust estimation of Δx_q^a, Δy_q^a and $\Delta\theta_q^a$ regardless of the actual position and orientation of the camera. Moreover, once such data are available the absolute position and heading of the *FriWalk* a can be determined by using the following output model, i.e.

$$y_k^a = h^a(s_k) = \begin{bmatrix} (x_q - x_k^a)\cos\theta_k^a + (y_q - y_k^a)\sin\theta_k^a \\ -(x_q - x_k^a)\sin\theta_k^a + (y_q - y_k^a)\cos\theta_k^a \\ \theta_q - \theta_k^a \end{bmatrix} \tag{5.3}$$

If y_k^a is measured, then (5.3) can be rewritten as $\hat{y}_k^a = h^a(s_k) + \zeta_k^a$, where random vector ζ_k^a models measurement uncertainty .

RGB-D camera: The RGB-D camera installed in front of a *FriWalk* to detect other agents in the environment and to estimate their relative position is an ASUS Xtion Pro 3D system. As soon as another *FriWalk b* enters into the field of view (FoV) of the RGB-D camera (with a reading range of about 3.5 m), the embedded image processing algorithm first detects the new agent and then it computes the *x*-*y* position offsets Δx^{ab} and Δy^{ab} between agent *b* and the principal point of the camera on *FriWalk a*, as shown in Figure 27. As a result, the relative measurement model can be defined as follows, i.e.

$$y_k^{ab} = h^{ab}(s_k) = \begin{bmatrix} (x_k^b - x_k^a)\cos\theta_k^a + (y_k^b - y_k^a)\sin\theta_k^a \\ -(x_k^b - x_k^a)\sin\theta_k^a + (y_k^b - y_k^a)\cos\theta_k^a \end{bmatrix} \tag{5.4}$$

Similarly to (5.3), if y_k^{ab} is measured, then (5.4) can be rewritten as $\hat{y}_k^{ab} = h^{ab}(s_k) + \zeta_k^{ab}$, where random vector ζ_k^{ab} models measurement uncertainty .

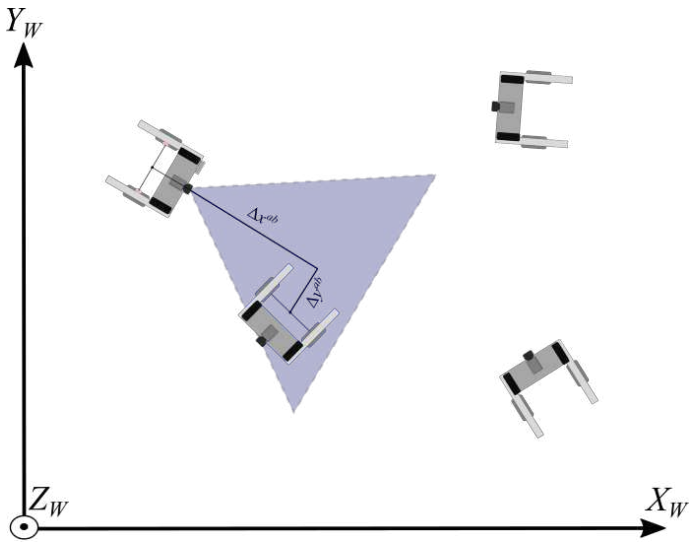

Figure 27: Position offset measurements between *FriWalks a* and *b*. Such measurements are performed with an ASUS Xtion Pro 3D system and are essential to perform collaborative localisation.

5.1.2 Cooperative Localisation Algorithm

Consider a system consisting of *N* independent *FriWalks* each one described by the process model (5.1) and by two intermittent measurement models, i.e. based on (5.3) and (5.4) depending on whether a QR code or another agent are detected at time kT_s, respectively. Using such models, a centralized EKF with intermittent observation can be easily defined. However, a centralized cooperative localisation approach requires a central master device receiving the sensor data from all agents, computing the location of each *FriWalk* and sending back the estimated results to all agents. Unfortunately, this implementation is quite impractical and hardly scalable. So implementing a distributed version of this algorithm is of primary importance.

After a thorough mathematical analysis, we reached the conclusion that this operation is possible but requires a complex formulation. A major and sneaky issue which arises in a possible distributed implementation and that may greatly influence estimator accuracy is the non-negligible effect of the cross-covariance between the estimated states of any pair of agents which perform some measurement at time kT_s. This problem does not exist in a centralized EKF, since the filter inherently takes into account the cross covariance of the states of different agents. To preserve such a consistency for the EKF implementation and distribute the computation of the collaborative localisation algorithm among the set of agents, [62] proposes to track the predicted values of the cross covariance matrices between the agents *i* and *j* in each agent *i* and *j* separately, and then update their values in one shot as soon as *i* measure the relative position to *j* or vice versa, namely as soon as one detects the other. This is mainly related to the way in which the cross covariance matrices are predicted. To clarify this point, let us consider the covariance matrix evolution of agent *i*. In this case, the predicted value is computed using its dynamic only. When the cross covariance between the agents *i* and *j* is considered, its predicted value is instead computed as using both the dynamic evolutions of the agents *i* and *j*. The interesting thing is that the dynamic of the *i*—th agent accumulates all on one side, while the dynamic of the *j*—th agent on the other. Hence, there is no cross products between the different dynamics and, hence, each agent can predict its own side of the cross covariance independently and perform the merge only when needed, i.e. when one detects the other. This way, the communication bandwidth is allocated only if strictly needed.

The basic steps of the algorithm are still basically two, i.e. *prediction* and *update*, and are summarized below:

- **Prediction step**

  At every sample time kT_s each agent *i*=1,...,*N* predicts its own state as well as the corresponding covariance matrix as follows, i.e.

$$\hat{s}_{k+1}^{i+} = f(\hat{s}_k^i, \widehat{\Delta\Phi}_k^i)$$
$$P_{k+1}^{i+} = F_k^i P_k^i {F_k^i}^T + G_k^i Q_k^i {G_k^i}^T \qquad (5.5)$$

  where F_k^i is the Jacobian of $f(\cdot)$ as defined in (5.1) with respect to the state variables of agent *i* at time kT_s, G_k^i is the Jacobian of $f(\cdot)$ with respect to the input quantities (namely the encoder increments), P_k^i is the state covariance matrix of the *i*th agent and Q_k^i is the diagonal covariance matrix of input data, since the sensor data of left and right wheels can be reasonably assumed to be uncorrelated.

  In addition, agent *i* computes also the state transition matrix

$$\Psi_{k+1}^i = F_k^i \Psi_k^i \text{ with } \Psi_0^i = I_5 \qquad (5.6)$$

  This matrix is needed in the following update step any time a measurement is performed.

- **Update step**

  In the update step three cases may occur on each agent: 1) a QR code is detected; 2) another agent is detected; 3) no measurements are performed.

  1. If an agent *a* detects a QR code, by using (5.3) it follows that

$$z_{k+1} = y_{k+1}^a - h^a(\hat{s}_{k+1}^{a+})$$
$$M_{k+1}^a = H_{k+1}^a P_{k+1}^{a+} {H_{k+1}^a}^T + R_{k+1}^a \qquad (5.7)$$
$$\Gamma_{k+1}^a = (\Psi_{k+1}^a)^{-1} P_{k+1}^{a+} {H_{k+1}^a}^T$$

     where H_k^a is the Jacobian of $h^a(\cdot)$ in (5.3) with respect to the state variables of agent *a* and computed at $\hat{s}_k^{a+}$, while R_{k+1}^a is the covariance matrix of the camera-based measurements of position and heading obtained from QR codes. Once the results of (5.7) are ready, agent *a* broadcasts to all the other agents the following data, i.e. $(a, z_{k+1}, M_{k+1}^a, \Gamma_{k+1}^a, {\Psi_{k+1}^a}^T {H_{k+1}^a}^T)$. Each agent *i*=1,...,*N* first computes *N*-1 matrices

$$\Gamma_{k+1}^j = \Pi_{ja_k}^i {\Psi_{k+1}^a}^T {H_{k+1}^a}^T \text{ for } j = \{1, \dots, N\} \backslash \{a\} \qquad (5.8)$$