ISSN 0280-5316
ISRN LUTFD2/TFRT--5843--SE

A Distributed Kalman Filter Algorithm for Self-localisation of Mobile Devices

Anne-Kathrin Hess

Department of Automatic Control
Lund University
August 2009

Lund University **Department of Automatic Control** **Box 118** **SE-221 00 Lund Sweden**	*Document name* MASTER THESIS *Date of issue* August 2009 *Document Number* ISRN LUTFD2/TFRT--5843--SE
Author(s) Anne-Kathrin Hess	*Supervisor* Prof. Rolf Findeisen. Inst. for Automation Eng. Lab. For Systems Theory and Control, Magdeburg Germany. Prof. Anders Rantzer Automatic Control, Lund. Ass. Prof. Anton Cervin Automatic Control, Lund (Examiner) *Sponsoring organization*

Title and subtitle

A Distributed Kalman Filter Algorithm for Self-localization of Mobile Devices.
(En distribuerad Kalmanfilteralgoritm för självlokalisering i rörliga object)

Abstract

In many applications involving mobile devices it is essential to track their position. Moreover, it is usually desired, to perform this localization in a distributed fashion without using a central processing unit. In this case, only distance measurements to reference nodes which are in range can be utilized. It is proposed in this thesis to also use distance measurements to other moving objects, to improve the position estimation.The self-localization task is addressed in this work by introducing a distributed Kalman Filter. This state observer estimates the position of the moving objects based on distance measurements to neighboring devices and reference nodes. Additionally it was investigated if the performance of this filter could be improve by adding a data fusion step to the filter.

In this case, every device additionally estimates the position of its neighbors. This generates multiple estimates for one object, which are afterwards fused using optimized weights. This allows the usage of more measurement information available in the network for the localization of one device. To compare the performance of the introduced algorithms, simulation results are given. A system with a static graph structure was investigated, as well as a system with a dynamic graph. It was found that the accuracy of the state estimation could be improved by introducing a data fusion step. Furthermore, it was seen that a higher average coupling among the nodes is necessary to ensure reliable performance when the graph is dynamic.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title 0280-5316		*ISBN*
Language English	*Number of pages* 70	*Recipient's notes*
Security classification		

Contents

List of Figures

List of Symbols

A_i	System matrix of node $\mathcal{N}_i$	5
$A_{[i]}$	System matrix of subsystem $\mathcal{N}_{[i]}$	6
$\mathbf{A}$	System matrix of network system $\mathcal{N}$	5
$\bar{\mathbf{A}}$	System matrix of extended system	17
C_i	Measurement matrix of node $\mathcal{N}_i$	6
$C_{[i]}$	reduced measurement matrix of node $\mathcal{N}_i$	6
$\bar{\bar{C}}_{[i]}$	Measurement matrix of subsystem $\mathcal{N}_{[i]}$	28
$\mathbf{C}$	Measurement matrix of network system $\mathcal{N}$	6
$\bar{\mathbf{C}}$	Measurement matrix of extended system	17
$\boldsymbol{d}^2$	Distance between two nodes	33
$\overline{\boldsymbol{d}}^2$	Distance between the steady state positions of two nodes	33
$e(k)$	Identity vector $G^{\xi}_{ji}(k)\mathbf{I}_{\bar{n}\times n_j}$	20
$\mathcal{G}^{\eta}$	Graph $\mathcal{G}^{\eta}(\mathcal{E}^{\eta}, \mathcal{N})$ (among mobile nodes)	4
$\mathcal{G}^{\mu}$	Graph $\mathcal{G}^{\mu}(\mathcal{E}^{\mu}, \mathcal{R})$ (mobile to reference nodes)	4
G^x_i	Selection matrix $x_{[i]} = G^x_i\mathbf{x}$	6
$\mathbf{G}^x$	Selection matrix $\mathbf{G}^x = [G^x_1 ... G^x_{\eta}]^T$	6
G^y_i	Selection matrix $y_{[i]} = G^y_i\mathbf{y}$	26
G^{ξ}_{ji}	Selection matrix $\hat{x}^{\text{loc}}_{[ji]} = G^{\xi}_{ji}\hat{\boldsymbol{\xi}}^{\mathbf{loc}}$	20
$\mathbf{G}^{\xi}$	Selection matrix $\mathbf{G}^{\xi} = [G^{\xi}_{11} ... G^{\xi}_{\eta\eta}]^T$	20
$\mathbf{I}_{n\times m}$	Identity matrix of dimension $n \times m$	13
K_i	Kalman gain of node $\mathcal{N}_i$ (dKF1)	12
$\mathbf{K}$	Kalman gain of network system $\mathcal{N}$ (dKF1)	12
$\bar{K}_i$	Kalman gain of node $\mathcal{N}_i$ (dKF2)	18
$\bar{\mathbf{K}}$	Kalman gain of dKF2 diag($\bar{\mathbf{K}}$) $= [\bar{K}_1 ... \bar{K}_{\eta}]$	20
$\bar{\bar{K}}_i$	Kalman gain of node $\mathcal{N}_i$ (dKF3)	24
$\bar{\bar{K}}_{[i]}$	Reduced Kalman gain of node $\mathcal{N}_i$ (dKF3)	26
$\bar{\bar{\mathbf{K}}}$	Kalman gain of network system $\mathcal{N}$ (dKF3)	24

$\mathcal{M}$	Set of reference nodes	3
$\mathcal{M}_{[i]}$	Set of reachable reference nodes of $\mathcal{N}_i$. $\mathcal{M}_{[i]} \subset \mathcal{M}$	4
$\mathcal{M}_i$	Reference node i, $\mathcal{M}_i \in \mathcal{M}$	4
m_i	Number of measurements of mobile node $\mathcal{N}_i$	6
$m_{[i]}$	Number of measurements of subsystem $\mathcal{N}_{[i]}$	26
m	Number of measurements in network system $\mathcal{N}$	6
$\mathcal{N}$	Set of mobile nodes	3
$\mathcal{N}_{[i]}$	Set of neighbors of $\mathcal{N}_i$, including itself.	4
$\mathcal{N}_i$	Mobile node $\mathcal{N}_i$	4
n_i	Number of states of mobile node $\mathcal{N}_i$	5
$n_{[i]}$	Number of states of subsystem $\mathcal{N}_{[i]}$	6
n	Number of states in network system $\mathcal{N}$	5
$\bar{n}$	Number of states in extended system	17
$\mathbf{0}_{n \times m}$	Zero matrix of dimension $n \times m$	13
$\mathbf{P}$	Covariance matrix of the estimation error $\tilde{\mathbf{x}}$	10
$P_{[i]}$	Covariance of the subsystem estimation error $\tilde{x}_{[i]}$	14
$\bar{\mathbf{P}}$	Covariance matrix $\bar{\mathbf{P}}$ of the estimation error $\tilde{\boldsymbol{\xi}}$	21
$\boldsymbol{p}$	Position vector of mobile nodes	32
$\overline{\boldsymbol{p}}$	Steady state position vector of mobile nodes	33
$\check{\boldsymbol{p}}$	Position vector of reference node	31
$\overline{\check{\boldsymbol{p}}}$	Steady state position vector of reference nodes	33
$\hat{\boldsymbol{p}}$	Position estimate	38
Q_i	Covariance matrix of the process noise w_i	5
$\mathbf{Q}$	Covariance matrix of the process noise $\mathbf{w}$	5
$\bar{\mathbf{Q}}$	Covariance matrix of the process noise $\bar{\mathbf{w}}$	17
$\mathcal{R}$	Set of all nodes $\mathcal{R} = \mathcal{N} \cup \mathcal{M}$	4
R_i	Covariance matrix of the measurement noise v_i	6
$R_{[i]}$	Covariance matrix of the measurement noise $v_{[i]}$	6
$\mathbf{R}$	Covariance matrix of the measurement noise $\mathbf{v}$	6
$\mathbb{R}$	Set of real numbers	5
U_i, V_i	Decomposition matrices for $\mathbf{K}$	12
$\bar{U}_i, \bar{V}_i$	Decomposition matrices for $\bar{\mathbf{K}}$	20
$\bar{\bar{U}}_i, \bar{\bar{V}}_i$	Decomposition matrices for $\bar{\bar{\mathbf{K}}}$	24
v_i	Measurement noise vector of node $\mathcal{N}_i$	6
$v_{[i]}$	Measurement noise vector of subsystem $\mathcal{N}_{[i]}$	6

$\mathbf{v}$	Measurement noise vector of network system	6
W_i	Weighting matrix to join estimates for $\mathcal{N}_i$ (dKF2)	18
$W_{[i]}$	Reduced weighting matrix for $\mathcal{N}_i$ (dKF2)	20
$\mathbf{W}$	Weighting matrix for network system $\mathcal{N}$ (dKF2)	20
$\bar{W}_i$	Weighting matrix to join estimates for $\mathcal{N}_i$ (dKF3)	24
$\bar{\mathbf{W}}$	Weighting matrix for network system $\mathcal{N}$ (dKF3)	24
w_i	Process noise vector of node $\mathcal{N}_i$	5
$w_{[i]}$	Process noise vector of subsystem $\mathcal{N}_{[i]}$	6
$\mathbf{w}$	Process noise vector of network system	5
$\bar{\mathbf{w}}$	Process noise vector of extended system	17
w_i^{lj}	Weight for estimate of $\mathcal{N}_l$ done by $\mathcal{N}_j$ and used by $\mathcal{N}_i$	18
x_i	State vector of node $\mathcal{N}_i$	5
$x_{[i]}$	State vector of subsystem $\mathcal{N}_{[i]}$	6
$\mathbf{x}$	State vector of network system $\mathcal{N}$	5
$\hat{x}_i$	State vector of node $\mathcal{N}_i$	10
$\hat{x}_{[i]}$	State vector of subsystem $\mathcal{N}_{[i]}$	12
$\hat{\mathbf{x}}$	State estimate of network system	12
$\hat{x}_{ji}^{\mathrm{loc}}$	State estimate of $\mathcal{N}_j$ calculated by $\mathcal{N}_i$	18
$\hat{x}_{[ji]}^{\mathrm{loc}}$	Vector with all state estimates of $\mathcal{N}_j$, communicated to $\mathcal{N}_i$	20
$\tilde{x}_i$	State estimation error of node $\mathcal{N}_i$	29
$\tilde{x}_{[i]}$	State estimation error of subsystem $\mathcal{N}_{[i]}$	14
$\tilde{\mathbf{x}}$	State estimation error of network system $\mathcal{N}$	10
$\overline{x}_i$	Steady state vector of node $\mathcal{N}_i$	5
$\overline{x}_{[i]}$	Steady state vector of subsystem $\mathcal{N}_{[i]}$	6
$\overline{\mathbf{x}}$	Steady state vector of network system $\mathcal{N}$	5
y_i	Measurement vector of node $\mathcal{N}_i$	5
$y_{[i]}$	Measurement vector of subsystem $\mathcal{N}_{[i]}$	6
$\mathbf{y}$	Measurement vector of network system $\mathcal{N}$	6
$\hat{y}_i$	Measurement prediction of node $\mathcal{N}_i$	10
$\hat{y}_{[i]}$	Measurement prediction of subsystem $\mathcal{N}_{[i]}$	29
$\hat{\mathbf{y}}$	Measurement prediction of network system $\mathcal{N}$	10
$\tilde{y}_i$	Measurement prediction error of node $\mathcal{N}_i$	12
$\tilde{y}_{[i]}$	Measurement prediction error of subsystem $\mathcal{N}_{[i]}$	26
$\tilde{\mathbf{y}}i$	Measurement prediction error of network system $\mathcal{N}$	24
$\overline{y}_{[i]}$	Measurement vector of subsystem $\mathcal{N}_{[i]}$ at steady state	29

ε	RMS error of position estimate	38
γ_i	Number of reachable nodes of $\mathcal{N}_i$, $\lvert\mathcal{M}_{[i]}\rvert = \gamma_i$	4
ζ_i	Number of neighbors of $\mathcal{N}_i$, including itself	4
η	Number of mobile nodes	3
μ	Number of reference nodes	3
ξ_i	Extended state vector of node $\mathcal{N}_i$	17
$\boldsymbol{\xi}$	State vector of extended system	17
$\hat{\xi}_i^{\mathrm{p}}$	State estimate of extended state vector, $p \in \{loc, reg\}$	18
$\hat{\xi}_{[i]}^{\mathrm{p}}$	Reduced state estimate of extended state vector	20
$\hat{\boldsymbol{\xi}}^{\mathbf{p}}$	State estimate of extended system $p \in \{loc, reg\}$	18
ρ	Communication Radius	4
τ	Timescale for robot movement	35
Φ	Reduced covariance matrix	22

Chapter 1

Introduction

Recent improvements in communication and calculation capabilities of mobile devices make it possible to implement advanced distributed control and observer strategies in multi agent systems. The usual advantages of distributed algorithms are the increase of efficiency, scalability and robustness. However, the distribution of familiar centralized control algorithms is usually not straightforward. It becomes especially challenging if strong communication constraints are assumed and no central processing unit is allowed.

In many applications of multi agent systems, localization of the mobile devices is essential. It is the objective of this thesis to design an algorithm, which estimates those locations in a distributed fashion. In particular a network system of mobile devices is considered in this work. Generally, the system dynamics are assumed to be time dependent and subject to stochastic disturbances.

The framework mostly used to estimate states of centralized stochastic systems is the Kalman Filter introduced by R. E. Kalman in 1960 [10]. In the literature different approaches for distributed Kalman Filters can be found. One method is to introduce a central unit which processes measurement information in a centralized fashion and distributes the state estimates back to the devices, e.g. [17]. Other papers only investigated systems which have significantly more states than available measurements. For these systems usually a Kalman Filter in information form [3] is used. This form has the advantage that calculations can be easily distributed among nodes, but a recalculation of the actual states is not trivial and thus usually not doable in a decentralized fashion, e.g. [6, 11]. None of these methods will be considered in this thesis, since neither a central processing unit nor a comparably small measurement vector are considered.

When it comes to localization of moving objects, GPS based position estimation is widely used. Nevertheless, this method is only applicable if the restrictions on the accuracy of the position estimate are low. Applications, where this assumption does not hold include for example the control of convoys on roads (e.g [7]). Using GPS for localization also fails in an indoor environment. Therefore most indoor localization algorithms use a sensor network instead which estimates the locations of the moving objects and communicates them back to the devices, e.g [18, 22]. This method is usually referred to as “active mobile architecture” because the mobile devices actively transmit signals to the sensors [18]. In [14] and [19] a distributed consensus filter was designed to track moving objects using such a sensor network. A distributed Kalman Filter using this measurement method is presented in [1, 2]. In this work a data fusion step is added to the Kalman Filter to increase the quality of the state estimates.

The drawback of an active mobile architecture is, that this method scales badly with an increasing number of mobile devices. Furthermore privacy concerns are introduced since every device is trackable by the infrastructure. This could be overcome by using a ”passive mobile architecture“ where static reference nodes actively emit signals and the mobile devices passively use those signals to measure their distance to the reference nodes independently [18]. An example for this architecture is the Cricket system [16].

If the network system gets sufficiently large, not every mobile node can communicate to every reference node. Consequently, the number of measurements available in both mobile architectures reduces. In this case the amount of measurements available can be increased by allowing mutual distance measurements and communication among neighboring mobile devices. This is in particular advantageous if interactions among moving objects are already used for other purposes, e.g. for collision avoidance. Then those measurements do not need to be generated additionally. This concept of using measurements to mobile nodes can also be found in algorithms for self-localization of nodes in networks, e.g. [8, 15]. These algorithms usually use communication and additional distance measurements among unlocalized nodes to improve performance. Nevertheless the positions of the nodes in the network are usually assumed to be known and fixed, preserving a static graph [20, 23].

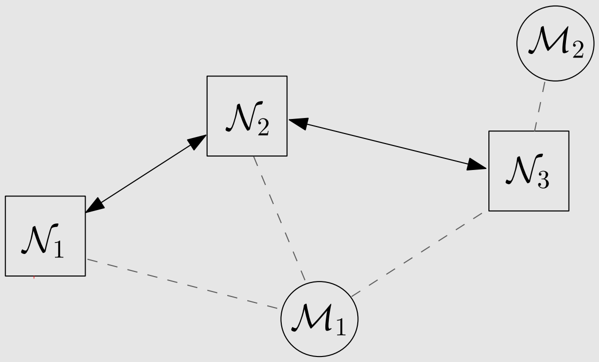
The measurement framework in this thesis extends the ”passive mobile architecture“ by allowing additional communication and measurements to mobile devices while in general the mobile devices are moving during operation. A theoretical framework to describe those systems will be introduced in Section 2. To clarify the notation a global Kalman Filter based on [10] is given in Section 3.1. By applying the assumed communication restrictions to this global filter a decentralized Kalman Filter is derived in Section 3.2. It was shown in [2] that the performance of a distributed Kalman Filter can be improved by adding a data fusion step when the active mobile architecture is used. Therefore it is investigated in Section 3.3.2 if the introduction of a data fusion step is also applicable to the communication framework under consideration. By modifying the algorithm given in [2] for the systems under consideration, only an algorithm for static mobile systems could be found. Therefore a third distributed Kalman Filter with weaker communication restrictions is introduced in Section 3.3.3 which also uses a data fusion step. This algorithm is also applicable to systems which have a dynamic graph. In Section 4 the performance of the introduced algorithms will be compared using a numerical example and MATLAB simulations.

Chapter 2

Problem Statement

The systems under consideration in this thesis consist of an arbitrary number η of mobile devices $\mathcal{N}$ and an arbitrary number μ of reference nodes (sensors) $\mathcal{M}$. All objects are locally distributed. While the reference nodes have a fixed position the mobile devices move randomly inside a certain area. The communication network of these systems is subject to physical constrains on the communication abilities of the nodes. The resulting communication topology will be introduced in Section 2.1. The objective of this work is to derive an algorithm where the whole system calculates an estimate of a distributed quantity by using only local measurements and information gathered through communication among devices. Even though the main focus of this thesis is position estimation the derived algorithms are also applicable to the estimation of any distributed state, e.g. temperature. Therefore only a general dynamical model of the system will be introduced in Section 2.2 and the model equations will be given. The derivation of the exact model equations for the application of self-localization can be found in Section 4.

Example: (Three Agent Network) *In order to give a more intuitive explanation of the theoretical results in this paper, this running example is introduced. The system consists of $\eta = 3$ mobile devices (squares) and $\mu = 2$ reference nodes (circles) as shown in the picture below. The black errors indicate communication among neighbors and the dashed gray lines indicate which reference nodes are reachable by the mobile nodes.*



2.1 Communication Topology

Graph theory is a convenient framework to study interactions of a finite set of elements. If the topology of interactions among these elements changes over time the graph is called dynamic. If this is not the case, a static graph is used [12, 23]. In this thesis we assume that only devices with a distance smaller than a maximal communication radius ρ can communicate, called "nearest neighboring scenario' in the literature, e.g. [12]. If not stated differently it will be assumed throughout this thesis that distance measurements can be taken to all nodes, to which communication is possible. Furthermore a dynamic graph is used, because in general all devices move randomly. The two graphs $\mathcal{G}_k^\eta = (\mathcal{N}, \mathcal{E}_k^\eta)$ and $\mathcal{G}_k^\mu = (\mathcal{M} \cup \mathcal{N}, \mathcal{E}_k^\mu)$ are introduced to model the interaction among the mobile nodes and between mobile and reference nodes respectively. The subscript k indicates the dependency of $\mathcal{G}$ on the discrete time k.

In $\mathcal{G}_k^\eta$ the edge $(\mathcal{N}_j, \mathcal{N}_i)$ is in $\mathcal{E}_k^\eta$ if and only if mobile node $\mathcal{N}_i$ and $\mathcal{N}_j$ can communicate at time k, giving the adjacency matrix $\mathcal{A}^\eta = [e_{ij}]$ with

$$e_{ij} = \begin{cases} 1 & \text{if} \quad d(\mathcal{N}_j, \mathcal{N}_i) \leq \rho \\ 0 & \text{otherwise,} \end{cases} \tag{2.1}$$

where $d(.,.)$ represents the euclidean distance between two nodes. If $\mathcal{N}_j$ and $\mathcal{N}_i$ can communicate, they are called neighbors. All neighbors of node $\mathcal{N}_i$ at time k are contained in the set $\mathcal{N}_{[i],k}$, including node i itself. Therefore the number of neighbors of node i, including itself, is given by[1]$|\mathcal{N}_{[i],k}| = \zeta_{i,k}$.

Graph $\mathcal{G}_k^\mu$ describes the communication between mobile nodes $\mathcal{N}$ and reference nodes $\mathcal{M}$. The edge $(\mathcal{M}_i, \mathcal{N}_i)$ is in $\mathcal{E}_k^\mu$ if and only if mobile node $\mathcal{N}_i$ can communicate to reference node $\mathcal{M}_i$. This can be summarized in matrix $\tilde{\mathcal{A}}^\mu = [e_{ij}]$, given by

$$e_{ij} = \begin{cases} 1 & \text{if} \quad d(\mathcal{M}_j, \mathcal{N}_i) \leq \rho \\ 0 & \text{otherwise,} \end{cases} \tag{2.2}$$

similar to the adjacency matrix (2.1). If mobile node $\mathcal{N}_i$ can communicate to reference node $\mathcal{M}_i$, node $\mathcal{M}_j$ is reachable by node $\mathcal{N}_i$. All reachable reference nodes $\mathcal{M}_j$ of node $\mathcal{N}_i$ at time k are contained in the set $\mathcal{M}_{[i],k}$. The number of reference nodes reachable by device $\mathcal{N}_i$ at time k is given by[1] $|\mathcal{M}_{[i],k}| = \gamma_{i,k}$.

Example: (Three Agent Network) *The communication topology of the three agent network is assumed to be constant over time and is therefore modeled using the static graphs $\mathcal{G}^\eta = (\mathcal{N}, \mathcal{E}^\eta)$ and $\mathcal{G}^\mu = (\mathcal{M} \cup \mathcal{N}, \mathcal{E}^\mu)$ with*

$$\begin{aligned} \mathcal{N} &= \{\mathcal{N}_1, \mathcal{N}_2, \mathcal{N}_3\} \qquad \eta = 3 \\ \mathcal{M} &= \{\mathcal{M}_1, \mathcal{M}_2\} \qquad \mu = 2 \end{aligned}$$

[1] To simplify notation, the dependency of $\zeta_{i,k}$, $\gamma_{i,k}$, $\mathcal{N}_{[i],k}$ and $\mathcal{M}_{[i],k}$ on the discrete time k will be dropped from now on.

$$\begin{aligned}\mathcal{E}^{\eta} &= \{(\mathcal{N}_1,\mathcal{N}_1),(\mathcal{N}_2,\mathcal{N}_1),(\mathcal{N}_1,\mathcal{N}_2),(\mathcal{N}_2,\mathcal{N}_2),(\mathcal{N}_3,\mathcal{N}_2),(\mathcal{N}_3,\mathcal{N}_2),(\mathcal{N}_3,\mathcal{N}_3)\}\\ \mathcal{E}^{\mu} &= \{(\mathcal{M}_1,\mathcal{N}_1),(\mathcal{M}_1,\mathcal{N}_2),(\mathcal{M}_1,\mathcal{N}_3),(\mathcal{M}_2,\mathcal{N}_3)\}\end{aligned}$$

and the adjacency matrices

$$\mathcal{A}^{\eta} = \begin{pmatrix}1 & 1 & 0\\ 1 & 1 & 1\\ 0 & 1 & 1\end{pmatrix} \qquad \tilde{\mathcal{A}}^{\mu} = \begin{pmatrix}1 & 1 & 1\\ 0 & 0 & 1\end{pmatrix}.$$

The sets of neighbors and the sets of reachable nodes are defined as follows:

$$\begin{array}{llll} \mathcal{N}_{[1]} = \{\mathcal{N}_1,\mathcal{N}_2\} & \zeta_1 = 2, & \mathcal{M}_{[1]} = \{\mathcal{M}_1\} & \gamma_1 = 1,\\ \mathcal{N}_{[2]} = \{\mathcal{N}_1,\mathcal{N}_2,\mathcal{N}_3\} & \zeta_2 = 3, & \mathcal{M}_{[2]} = \{\mathcal{M}_1\} & \gamma_2 = 1,\\ \mathcal{N}_{[3]} = \{\mathcal{N}_2,\mathcal{N}_3\} & \zeta_3 = 2 & \mathcal{M}_{[3]} = \{\mathcal{M}_1,\mathcal{M}_2\} & \gamma_3 = 2. \end{array}$$

2.2 System Dynamics

The objective of this work is to estimate the states of the mobile devices $\mathcal{N}$ using information gathered through communication to other mobile nodes and to the sensor nodes $\mathcal{M}$. Therefore the system dynamics are only connected to the mobile devices $\mathcal{N}$ and their dynamics are furthermore assumed to be decoupled and linear. The system equation for one agent can therefore be described using

$$x_i(k) = A_i(k)x_i(k-1) + w_i(k) \quad \text{with } x_i(0) = x_i^0, \quad \forall i \in \mathcal{N} \tag{2.3}$$

where $x_i \in \mathbb{R}^{n_i}$ is the state vector and $w_i \in \mathbb{R}^{n_i}$ is the process noise of node $\mathcal{N}_i$. The noise w_i is modeled as a white zero mean Gaussian process w_i with the covariance matrix $Q_i = E[w_i {w_i}^T]$. The system equation for the overall system can be given in vector form

$$\mathbf{x}(k) = \mathbf{A}(k)\mathbf{x}(k-1) + \mathbf{w}(k) \qquad \text{with} \quad \mathbf{x}(0) = \mathbf{x}^0 \tag{2.4}$$

$$\begin{aligned}\text{using} \quad \mathbf{x} &= \begin{bmatrix}{x_1}^T & {x_2}^T & \dots & {x_\eta}^T\end{bmatrix}^T\\ \mathbf{w} &= \begin{bmatrix}{w_1}^T & {w_2}^T & \dots & {w_\eta}^T\end{bmatrix}^T\end{aligned}$$

and $\mathbf{A}$ being a block diagonal matrix diag$\{\mathbf{A}\} = \begin{bmatrix}A_1 & A_2 & \dots & A_\eta\end{bmatrix}$. In (2.4) $\mathbf{x} \in \mathbb{R}^n$, $n = \sum_{i=1}^{\eta} n_i$ is the state vector of the complete system and $\mathbf{w} \in \mathbb{R}^n$ is the process noise with covariance $\mathbf{Q} = E[\mathbf{w}\mathbf{w}^T]$.

In the considered setup the state estimation of one agent is based on measurements and communication with other agents and reference nodes. Therefore it must be assumed that in general the measurement equation of one agent is dependent on other agents states leading to a coupled measurement equation. The measurement dynamics of one agent $\mathcal{N}_i$

are therefore given by

$$y_i(k) = C_i(k)\mathbf{x}(k) + v_i(k) \quad \forall i \in \mathcal{N} \tag{2.5}$$

where $y_i \in \mathbb{R}^{m_{i,k}}$ is the measurement vector[2] and $v_i \in \mathbb{R}^{m_{i,k}}$ is the measurement noise of node $\mathcal{N}_i$. The noise v_i is modeled as a white zero mean Gaussian process v_i with the covariance matrix $R_i = E[v_i v_i{}^T]$. The assumption that communication is only possible to neighboring nodes was introduced in Section 2.1. This leads to zero columns $c_{:j}\ \forall j \notin \mathcal{N}_{[i]}$ in the local measurement matrices $C_i(k)$. By introducing the new variables

$$\begin{aligned} &x_{[i]}(k) = G_i^x(k)\mathbf{x}(k) \\ &A_{[i]}(k) = G_i^x(k)\mathbf{A}(k)G_i^x(k)^T \quad && C_{[i]}(k) = C_i(k)G_i^x(k)^T \\ &w_{[i]}(k) = G_i^x(k)\mathbf{w}(k) && v_{[i]}(k) = G_i^x(k)\mathbf{v}(k) \end{aligned} \tag{2.6}$$

with $G_i^x(k)^T G_i^x(k) = \mathbf{I}$, the measurement equation (2.5) can be rewritten in the following equivalent form

$$y_i(k) = C_{[i]}(k)x_{[i]}(k) + v_i(k) \tag{2.7}$$

where $x_{[i]} \in \mathbb{R}^{n_{[i],k}}$, $n_{[i],k} = \sum_{j\in\mathcal{N}_{[i]}} n_j$ is the state vector[3] of the subsystem $\mathcal{N}_{[i]}$, with $x_{[i]}(k) = A_{[i]}(k)x_{[i]}(k-1) + w_{[i]}(k)$. Furthermore the matrix $C_{[i]}(k) = C_i(k)G_i^x(k)^T$ is the reduced measurement matrix of node $\mathcal{N}_i$. The selection matrix G_i^x in (2.6) is given by the nonzero lines of

$$G_I^x = \begin{bmatrix} g^{1,i} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & g^{\eta,i} \end{bmatrix} \quad \text{with} \quad g^{j,i} = \begin{cases} \mathbf{I}_{n_j\times n_j} & \text{if } j \in \mathcal{N}_{[i]} \\ \mathbf{0}_{n_j\times n_j} & \text{otherwise} \end{cases} .$$

Using (2.5) the measurement equation for the complete system is given in vector form by

$$\mathbf{y}(k) = \mathbf{C}(k)\mathbf{x}(k) + \mathbf{v}(k) \tag{2.8}$$

$$\begin{aligned} \text{using} \quad \mathbf{y} &= \begin{bmatrix} y_1{}^T & y_2{}^T & \dots & y_\eta{}^T \end{bmatrix}^T \\ \mathbf{C} &= \begin{bmatrix} C_1{}^T & C_2{}^T & \dots & C_\eta{}^T \end{bmatrix}^T \\ \mathbf{v} &= \begin{bmatrix} v_1{}^T & v_2{}^T & \dots & v_\eta{}^T \end{bmatrix}^T. \end{aligned}$$

In (2.8) $\mathbf{y} \in \mathbb{R}^{m_k}$, $m_k = \sum_{i=1}^{\eta} m_{i,k}$ is the state vector[4] of the complete system and $\mathbf{v} \in \mathbb{R}^{m_k}$ is the measurement noise with covariance $\mathbf{R} = E[\mathbf{v}\mathbf{v}^T]$. Using (2.4), (2.8), (2.3) and (2.5) the model equations can be given

[2] The dependency of $m_{i,k}$ on k will be dropped.

[3] The dependency of $n_{[i],k}$ on k will be dropped.

[4] The dependency of m_k on k will be dropped.

Model equations

$$\text{local:} \quad \begin{aligned} x_i(k) &= A_i(k)x_i(k-1) + w_i(k) \quad \text{with} \quad x_i(0) = x_i^0 \\ y_i(k) &= C_{[i]}(k)x_{[i]}(k) + v_i(k) \end{aligned} \tag{2.9a}$$

$$\text{global:} \quad \begin{aligned} \mathbf{x}(k) &= \mathbf{A}(k)\mathbf{x}(k-1) + \mathbf{w}(k) \quad \text{with} \quad \mathbf{x}(0) = \mathbf{x}^0 \\ \mathbf{y}(k) &= \mathbf{C}(k)\mathbf{x}(k) + \mathbf{v}(k) \end{aligned} \tag{2.9b}$$

Example: (Three Agent Network) *All three agents in this example have a state vector of size* $n_i = 2 \ \forall i \in \mathcal{N}$ *and their system matrix is assumed to be the identity matrix* $A_i = \mathbf{I}_{2\times 2} \ \forall \ i \in \mathcal{N}$*. Furthermore the dimensions of the measurement are* $m = \{2, 3, 3\}$ *and the time independent measurement matrices* C_i *and their reduced analogous* $C_{[i]}$ *are given by*

$$C_1 = \begin{pmatrix} 16 & 12 & 0 & 0 & 0 & 0 \\ -16 & 4 & 16 & -4 & 0 & 0 \end{pmatrix} \quad C_{[1]} = \begin{pmatrix} 16 & 12 & 0 & 0 \\ -16 & 4 & 16 & -4 \end{pmatrix}$$

$$C_2 = \begin{pmatrix} -16 & 4 & 16 & -4 & 0 & 0 \\ 0 & 0 & 32 & -24 & 0 & 0 \\ 0 & 0 & -12 & -2 & 12 & 2 \end{pmatrix} \quad C_{[2]} = C_2$$

$$C_3 = \begin{pmatrix} 0 & 0 & -12 & -2 & 12 & 2 \\ 0 & 0 & 0 & 0 & 12 & 10 \\ 0 & 0 & 0 & 0 & -3 & 8 \end{pmatrix} \quad C_{[3]} = \begin{pmatrix} -12 & -2 & 12 & 2 \\ 0 & 0 & 12 & 10 \\ 0 & 0 & -3 & 8 \end{pmatrix}.$$

Analogously the reduced state vectors $x_{[i]}$ *are defined as*

$$x_{[1]} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T \quad x_{[2]} = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}^T \quad x_{[3]} = \begin{bmatrix} x_2 & x_3 \end{bmatrix}^T$$

and the global model is given by

$$\begin{pmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{pmatrix} = \begin{pmatrix} A_1 & 0 & 0 \\ 0 & A_2 & 0 \\ 0 & 0 & A_3 \end{pmatrix} \begin{pmatrix} x_1(k-1) \\ x_2(k-1) \\ x_3(k-1) \end{pmatrix} + \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix}$$

$$\begin{pmatrix} y_1(k) \\ y_2(k) \\ y_3(k) \end{pmatrix} = \begin{pmatrix} C_1 \\ C_2 \\ C_3 \end{pmatrix} \begin{pmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{pmatrix} + \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}.$$

Chapter 3

Kalman Filter Algorithms

In this chapter different Kalman Filter algorithms will be derived. As an introduction to the terminology of Kalman Filters, the standard Kalman Filter found by R. E. Kalman in 1960 [10] will be briefly described in Section 3.1. This algorithm calculates a global state estimate of the multi agent system using all available information. Conversely, the aim of a distributed filter is to parallelize the state estimation such that every node determines its state estimate independently. Due to the communication restrictions every node can only use a limited amount of information to calculate a state estimate. Based on those restrictions the first distributed algorithm is introduced as a non-optimal form of the global Kalman Filter in Section 3.2. In Section 3.3 the concept of distributed Kalman Filtering using joined estimates is introduced. This idea is taken from a model based data fusion algorithm [2] which will be shortly introduced in Section 3.3.1. Afterwards this method is applied to the systems under consideration in Section 3.3.2. Since only a filter for static network systems could be derived in Section 3.3.1 this fusion algorithm is modified in Section 3.3.3 to handle graph changes.

3.1 Global Kalman Filter

A Kalman Filter is a state observer invented by R. E. Kalman in 1960 [10] designed for systems of the form (2.9). The resulting estimate is unbiased and statistically optimal with respect to the covariance of the estimation error [13]. The classical Kalman Filter consist of two steps:

1. **Update**
   The state prediction $\hat{\mathbf{x}}(k|k-1)$ of the previous timestep is updated using the current measurement $\mathbf{y}(k)$. The normal Kalman Filter uses a linear function to calculate the updated state estimate with

$$\hat{\mathbf{x}}(k|k) = L_1(k)\hat{\mathbf{x}}(k|k-1) + L_2(k)\mathbf{y}(k). \tag{3.1a}$$

2. **Prediction**
   By using the system equation (2.9) and the updated state estimate $\hat{\mathbf{x}}(k|k)$, the state

estimate $\hat{\mathbf{x}}(k+1|k)$ of time k is predicted using

$$\hat{\mathbf{x}}(k+1|k) = \mathbf{A}(k)\hat{\mathbf{x}}(k|k). \tag{3.1b}$$

The state estimate $\hat{\mathbf{x}}(k|k)$ of the Kalman Filter must be unbiased and statistically optimal [13]. Therefore $L_1(k)$ and $L_2(k)$ must generate an unbiased estimate

$$E\left[\hat{\mathbf{x}}(k|k) - \mathbf{x}(k)\right] \overset{!}{=} 0 \Leftrightarrow E[\hat{\mathbf{x}}(k|k-1) - \mathbf{x}(k)] = 0 \tag{3.2a}$$

and minimize the covariance matrix $\mathbf{P}(k|k)$ of the estimation error, with

$$\mathbf{P}(k|k) = E\left[(\hat{\mathbf{x}}(k|k) - \mathbf{x}(k))(\hat{\mathbf{x}}(k|k) - \mathbf{x}(k))^{\mathrm{T}}\right].$$

In the case of a global Kalman Filter, the second criterion is equivalent to minimizing the trace of $\mathbf{P}(k|k)$ giving

$$L_1, L_2 = \underset{L_1,\ L_2}{\operatorname{argmin}}\ \mathrm{tr}(\mathbf{P}(k|k)). \tag{3.2b}$$

By using (3.1) in (3.2a) it can be determined that

$$L_1(k) = I - L_2(k)\mathbf{C}(k).$$

This leads to the following observation equations of the global Kalman Filter

$$\hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + \mathbf{K}(k)(\mathbf{y}(k) - \mathbf{C}\hat{\mathbf{x}}(k|k-1)) \tag{3.3a}$$

$$\hat{\mathbf{x}}(k+1|k) = \mathbf{A}\hat{\mathbf{x}}(k|k) \quad \text{with} \quad \hat{\mathbf{x}}(1|0) = \hat{\mathbf{x}}^0 \tag{3.3b}$$

where $\hat{\mathbf{y}}(k) = \mathbf{C}\hat{\mathbf{x}}(k)$ is the measurement prediction and $\mathbf{K}(k) = L_2(k)$ is the Kalman gain. This leads to a recursive state estimation, as visualized in Fig. 3.1. Using the model equations (2.9) and the estimation equations (3.3) the covariance matrices $\mathbf{P}$ of the estimation error $\tilde{\mathbf{x}}$ for the global model are given by

$$\begin{aligned}\mathbf{P}(k|k) &= E\left[\tilde{\mathbf{x}}(k|k)\tilde{\mathbf{x}}(k|k)^T\right] \\ &= \left[\mathbf{I} - \mathbf{K}(k)\mathbf{C}(k)\right]\mathbf{P}(k|k-1)\left[\mathbf{I} - \mathbf{C}(k)^T\mathbf{K}(k)^T\right] \\ &\quad + \mathbf{K}(k)\mathbf{R}(k)\mathbf{K}(k)^T\end{aligned} \tag{3.4a}$$

$$\begin{aligned}\mathbf{P}(k+1|k) &= E\left[\tilde{\mathbf{x}}(k+1|k)\tilde{\mathbf{x}}(k+1|k)^T\right] \\ &= \mathbf{A}(k)\mathbf{P}(k|k)\mathbf{A}(k) + \mathbf{Q}(k)\end{aligned} \tag{3.4b}$$

where $\tilde{\mathbf{x}}(k+1|k) = \mathbf{x}(k) - \hat{\mathbf{x}}(k+1|k)$ and $\tilde{\mathbf{x}}(k|k) = \mathbf{x}(k) - \hat{\mathbf{x}}(k|k)$ are the prediction and the estimation error, respectively. Since $\mathbf{K}(k)$ must fulfill (3.2b), it can be calculated using

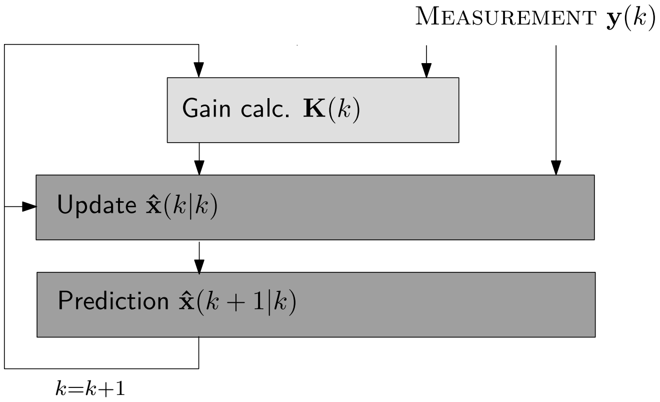

MEASUREMENT $\mathbf{y}(k)$
Gain calc. $\mathbf{K}(k)$
Update $\hat{\mathbf{x}}(k|k)$
Prediction $\hat{\mathbf{x}}(k+1|k)$
$k{=}k{+}1$

Fig. 3.1: Schematic representation of the global Kalman Filter algorithm. By using a global Kalman Filter, all measurements $\mathbf{y}(k)$ in the network are available for the state estimation. The gain calculation is performed online because a time dependent system is assumed.

the following minimization problem

$$\mathbf{K}(k) = \underset{\mathbf{K}(k)}{\operatorname{argmin}}\ \operatorname{tr}(\mathbf{P}(k|k)). \tag{3.5}$$

It is shown in Appendix A.2 that the conditions for an optimal $\mathbf{K}$ become

$$\mathbf{K}(k) = \mathbf{P}(k|k-1)\mathbf{C}(k)^{\mathrm{T}}\left[\mathbf{C}(k)\mathbf{P}(k|k-1)\mathbf{C}(k)^{\mathrm{T}} + \mathbf{R}(k)\right]^{-1}. \tag{3.6}$$

Using (3.3), (3.4) and (3.6) the global Kalman Filter algorithm can be summarized as follows:

global Kalman Filter:

$$\begin{aligned}
\mathbf{K}(k) &= \mathbf{P}(k|k-1)\mathbf{C}(k)^{\mathrm{T}}\left[\mathbf{C}(k)\mathbf{P}(k|k-1)\mathbf{C}(k)^{\mathrm{T}} + \mathbf{R}(k)\right]^{-1}\\
\hat{\mathbf{x}}(k|k) &= \hat{\mathbf{x}}(k|k-1) + \mathbf{K}(k)(\mathbf{y}(k) - \mathbf{C}(k)\hat{\mathbf{x}}(k|k-1))\\
\mathbf{P}(k|k) &= \left[\mathbf{I} - \mathbf{K}(k)\mathbf{C}(k)\right]\mathbf{P}(k|k-1)\left[\mathbf{I} - \mathbf{C}(k)^{T}\mathbf{K}(k)^{T}\right]\\
&\quad + \mathbf{K}(k)\mathbf{R}(k)\mathbf{K}(k)^{T}\\
\hat{\mathbf{x}}(k+1|k) &= \mathbf{A}(k)\hat{\mathbf{x}}(k|k) \quad \text{with} \quad \hat{\mathbf{x}}(1|0) = \hat{\mathbf{x}}^{0}\\
\mathbf{P}(k+1|k) &= \mathbf{A}(k)\mathbf{P}(k|k)\mathbf{A}(k) + \mathbf{Q}(k) \quad \text{with } \mathbf{P}(1|0) = \mathbf{P}^{0}
\end{aligned} \tag{3.7}$$

It can be seen in (3.1) that the Kalman gain $\mathbf{K}$ can be calculated offline, if the matrices $\mathbf{A}$ and $\mathbf{C}$ are time independent and assumptions about the covariances $\mathbf{R}$ and $\mathbf{Q}$ can be made a priori to the simulation.

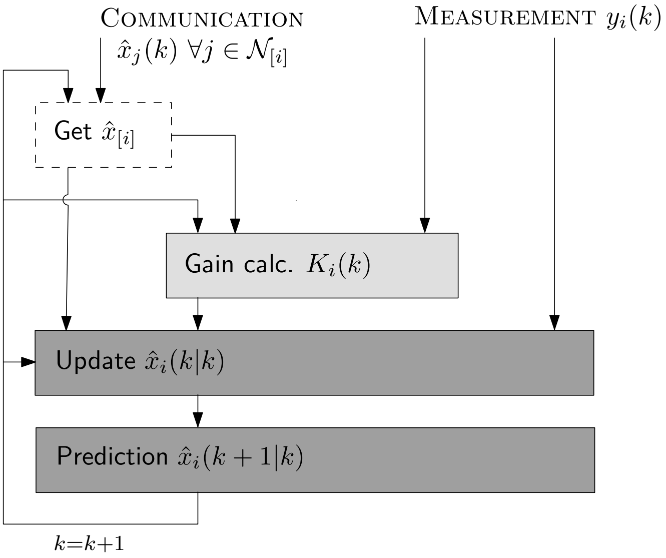


Fig. 3.2: Schematic representation of a distributed Kalman Filter using only local measurements $y_i(k)$. The communication of the state predictions $\hat{x}_i(k|k+1)$ is needed, since the local update equation (3.8) of the system is dependent on the subsystem state prediction $\hat{x}_{[i]}(k|k-1)$ of the former timestep. The local Kalman gain $K_i(k)$ must be calculated online, if the system dynamics are time dependent.

3.2 Distributed Kalman Filter using Local Measurements

This distributed Kalman Filter is based on the global filter given in Section 3.1. To guarantee that the algorithm is separable among nodes, the calculation of a local state estimate $\hat{x}_i$ must depend only on locally available information. In other words, every node has only access to its measurement vector y_i. The resulting algorithm is schematically shown in Fig. 3.2. It can be seen, that communication of the state prediction $\hat{x}_i(k+1|k)$ is allowed among neighbors. This is necessary since the local measurement equation (2.9) is dependent on the subsystem state estimate $x_{[i]}$. When using this algorithm, the observation equations for every single node $i \in \mathcal{N}$ become

$$\begin{aligned} \hat{x}_i(k|k) &= \hat{x}_i(k|k-1) \\ &\quad + K_i(k)\left(y_i(k) - C_{[i]}(k)\hat{x}_{[i]}(k|k-1)\right) \\ \hat{x}_i(k+1|k) &= A_i(k)\hat{x}_i(k|k) \quad \text{with} \quad \hat{x}_i(1|0) = \hat{x}_i^0. \end{aligned} \tag{3.8}$$

When (3.8) is written in matrix form

$$\begin{aligned} \hat{\mathbf{x}}(k|k) &= \hat{\mathbf{x}}(k|k-1) + \mathbf{K}(k)\left(\mathbf{y}(k) - \mathbf{C}(k)\hat{\mathbf{x}}(k|k-1)\right) \\ \hat{\mathbf{x}}(k+|k) &= \mathbf{A}(k)\hat{\mathbf{x}}(k|k) \quad \text{with} \quad \hat{\mathbf{x}}(1|0) = \hat{\mathbf{x}}^0 \end{aligned} \tag{3.9}$$

the Kalman Matrix $\mathbf{K}(k)$, diag$\{\mathbf{K}\} = \begin{bmatrix} K_1 & \dots & K_\eta \end{bmatrix}$ becomes block diagonal. To isolate the free parameters of $\mathbf{K}(k)$ for the optimization problem a matrix decomposition was used. This idea was inspired by [2]. Following this paper, the decoupling Kalman gain can be expressed by a sum

$$\mathbf{K}(k) = \sum_{i=1}^{\eta} U_i{}^T K_i(k) V_i(k) \tag{3.10}$$

$$\begin{aligned} \text{with} \qquad U_i &= \begin{bmatrix} \mathbf{0}_{n_i\times n_i(i-1)} & \mathbf{I}_{n_i\times n_i} & \mathbf{0}_{n_i\times n_i(\eta-i)} \end{bmatrix} \\ V_i(k) &= \begin{bmatrix} \mathbf{0}_{m_i\times\sum_{j=1}^{i-1} m_j} & \mathbf{I}_{m_i\times m_i} & \mathbf{0}_{m_i\times\sum_{j=i+1}^{\eta} m_j} \end{bmatrix} \end{aligned}$$

where the matrices $\mathbf{0}$ and $\mathbf{I}$ are the zero and the identity matrix respectively.

Example: (Three Agent Network) *For the three agent network introduced in Section 2 the Kalman Matrix $\mathbf{K}$ for the distributed algorithm is given by*

$$\mathbf{K} = \begin{bmatrix} [K_1] & 0 & 0 \\ 0 & [K_2] & 0 \\ 0 & 0 & [K_3] \end{bmatrix} \qquad \textit{with} \qquad \begin{aligned} dim(K_1) &= 2\times 2 \\ dim(K_2) &= 2\times 3 \\ dim(K_3) &= 2\times 3. \end{aligned}$$

The decomposition matrices U_i and V_i become

$$U_1 = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \end{bmatrix} \qquad V_1 = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \end{bmatrix}$$

$$U_2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \qquad V_2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$U_3 = \begin{bmatrix} 0 & \dots & 0 & 1 & 0 \\ 0 & \dots & 0 & 0 & 1 \end{bmatrix} \qquad V_3 = \begin{bmatrix} 0 & \dots & 0 & 1 & 0 & 0 \\ 0 & \dots & 0 & 0 & 1 & 0 \\ 0 & \dots & 0 & 0 & 0 & 1 \end{bmatrix}$$

The problem of calculating an optimal diagonal matrix is known in many applications, e.g. [2, 4]. Following [2], in this thesis the optimization criterion (3.2b) for the global Kalman Filter is used to calculate the Kalman gain (3.10), even though this does not lead to an optimal result. Therefore the following optimization problem

$$\begin{aligned} \mathbf{K}(k) &= \underset{\mathbf{K}(k)}{\operatorname{argmin}} \operatorname{tr}(\mathbf{P}(k|k)) \\ \text{s.t.} \quad \mathbf{K}(k) &= \sum_{i=1}^{\eta} U_i{}^T K_i(k) V_i(k) \end{aligned} \tag{3.11}$$

is given for the gain matrix. It is shown in Appendix A.3 that the following Kalman gains

$K_i \; \forall i \in \mathcal{N}$ solve this optimization problem

$$\begin{aligned} & K_i(k) = -\Upsilon_i(k)\Big(\Psi_i(k)\Big)^{-1} \\ & \text{with} \quad \Upsilon_i(k) = U_i\mathbf{P}(k|k-1)C_i(k)^T \\ & \qquad\quad \Psi_i(k) = C_i(k)\mathbf{P}(k|k-1)C_i(k)^T + R_i(k). \end{aligned} \tag{3.12}$$

Offline Gain Calculation

It is possible, that in a special application the graph of the network is static and the matrices $\mathbf{A}$ and $\mathbf{C}$ are time independent. If assumptions about the covariances $\mathbf{R}$ and $\mathbf{Q}$ can be made a priori to the experiment the Kalman gains K_i can then be calculated offline. In this case, the estimation error covariance $\mathbf{P}(k|k-1)$ in (3.12) can be calculated using (3.4b). Then the distributed Kalman filter (dKF) with offline gain calculation can be summarized as follows:

dKF using local measurements (offline gain calculation)

$$\begin{aligned} \text{offline:} \quad & K_i(k) = U_i\mathbf{P}(k|k-1)C_i^T\left[C_i\mathbf{P}(k|k-1)C_i^T + R_i\right]^{-1} \\ & \mathbf{P}(k|k) = \left[\mathbf{I} - \mathbf{K}(k)\mathbf{C}\right]\mathbf{P}(k|k-1)\left[\mathbf{I} - \mathbf{C}^T\mathbf{K}(k)^T\right] \\ & \qquad\quad + \mathbf{K}(k)\mathbf{R}\mathbf{K}(k)^T \\ & \mathbf{P}(k+1|k) = \mathbf{A}\mathbf{P}(k|k)\mathbf{A} + \mathbf{Q} \quad \text{with} \quad \mathbf{P}(1|0) = \mathbf{P}_0 \\ \text{online:} \quad & \hat{x}_i(k|k) = \hat{x}_i(k|k-1) + K_i(k)\left(y_i(k) - C_{[i]}\hat{x}_{[i]}(k|k-1)\right) \\ & \hat{x}_i(k+1|k) = A_i\hat{x}_i(k|k) \quad \text{with} \quad \hat{x}_i(1|0) = \hat{x}_i^0 \\ & \textsc{Communication}: \hat{x}_j(k) \; \forall j \in \mathcal{N}_{[i]} \end{aligned} \tag{3.13}$$

Online Calculation

Usually assumptions about a time independent graph do not hold. In this case Υ_i and Ψ_i need to be calculated online. This has to be done independently in every node without involving further communication. Using the same argumentation as in (2.7) we can rewrite (3.12) in

$$\begin{aligned} \Upsilon_i(k) &= p_{[i]}(k|k-1)C_{[i]}(k)^T \\ \Psi_i(k) &= C_{[i]}(k)P_{[i]}(k|k-1)C_{[i]}(k)^T + R_i(k) \end{aligned} \tag{3.14}$$

with $P_{[i]}(k|k-1) = G_i^x\mathbf{P}(k|k-1)(G_i^x)^T$ being the covariance of the subsystem estimation error $\tilde{x}_{[i]}(k|k-1)$ and $p_{[i]}(k|k-1)$ being the columns of $P_{[i]}(k|k-1)$ corresponding to $\tilde{x}_i$. Even though $C_{[i]}$ and R_i in (3.14) are known by node i, the calculation of $P_{[i]}(k|k-1)$ in (3.14) requires the Kalman gains $K_j \forall j \in \mathcal{N}_{[i]}$ and measurement matrices $C_j \; \forall j \in \mathcal{N}_{[i]}$.

Therefore $P_{[i]}(k|k-1)$ cannot be calculated analytically and is therefore approximated using

$$\begin{aligned} P_{[i]}(k|k-1) &= E\left[\tilde{\mathbf{x}}(k|k-1)\tilde{\mathbf{x}}(k|k-1)^T\right] \\ &\approx \frac{1}{k}\sum_{l=1}^{k}\left((\overline{x}_{[i]} - \hat{x}_{[i]}(l|l-1))(\overline{x}_{[i]} - \hat{x}_{[i]}(l|l-1))^T\right) \end{aligned} \tag{3.15}$$

where $\overline{x}_{[i]}$ is the mean value of $\hat{x}_{[i]}(k|k-1)$. This leads to a completely distributed recursive algorithm given by:

dKF using local measurements (offline gain calculation)

$$\begin{aligned} P_{[i]}(k|k-1) &= \frac{1}{k}\sum_{l=1}^{k}\left((\overline{x}_{[i]} - \hat{x}_{[i]}(l|l-1))(\overline{x}_{[i]} - \hat{x}_{[i]}(l|l-1))^T\right) \\ K_i(k) &= p_{[i]}(k|k-1)C_{[i]}(k)^T \\ &\quad \left[C_{[i]}(k)P_{[i]}(k|k-1)C_{[i]}(k)^T + R_i(k)\right]^{-1} \\ \hat{x}_i(k|k) &= \hat{x}_i(k|k-1) + K_i(k)\left(y_i(k) - C_{[i]}(k)\hat{x}_{[i]}(k|k-1)\right) \\ \hat{x}_i(k+1|k) &= A_i(k)\hat{x}_i(k|k) \quad \text{with} \quad \hat{x}_i(1|0) = \hat{x}_i^0 \\ &\textsc{Communication} : \hat{x}_j(k)\ \forall j \in \mathcal{N}_{[i]} \end{aligned} \tag{3.16}$$

3.3 Distributed Kalman Filter with Data Fusion

Using only local information for the state estimation result in a strongly restricted optimization problem as given in Section 3.2. Therefore state estimates can be improved by using more information about the global system. Consequently, extending the available information for the state estimation in every node is the objective for the Kalman Filter derived in this section. The idea for its realization is taken from the model based fusion algorithm given in [2] leading to an extension of the state vector. To clarify this concept it will be briefly introduced in Section 3.3.1. Afterwards this method is applied to networked systems with static graphs in Section 3.3.2. Finally, this fusion algorithm is modified in Section 3.3.3 for systems with a dynamic graph.

3.3.1 Data Fusion in Sensor Networks

In the model based data fusion algorithm derived in [2] the position of one mobile device is estimated by a network of σ locally distributed sensor nodes $\mathcal{S}$. The communication abilities of the sensors among each other are modeled using a static graph $\mathcal{G}^\sigma = (\mathcal{S}, \mathcal{E}^\sigma)$ where the edge (p, q) is in $\mathcal{E}^\sigma$ if and only if sensor p and q can exchange messages. Sensors to which sensor p can communicate are called neighbors and are contained in the set $\mathcal{S}_{[p]}$, including sensor $\mathcal{S}_p$ itself. It is assumed that the dynamics of the observed device are time invariant

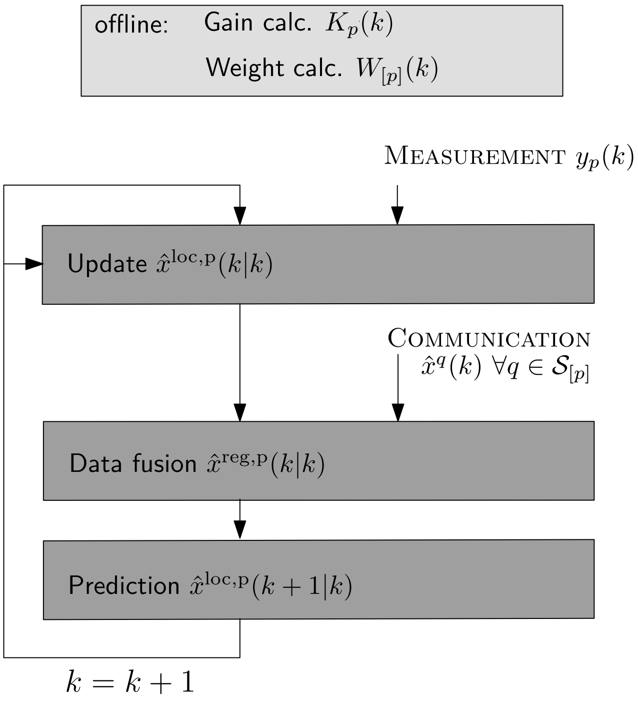

offline: Gain calc. $K_p(k)$
Weight calc. $W_{[p]}(k)$
Measurement $y_p(k)$
Update $\hat{x}^{\mathrm{loc,p}}(k|k)$
Communication
$\hat{x}^q(k)\ \forall q \in \mathcal{S}_{[p]}$
Data fusion $\hat{x}^{\mathrm{reg,p}}(k|k)$
Prediction $\hat{x}^{\mathrm{loc,p}}(k+1|k)$
$k = k + 1$

Fig. 3.3: Schematic representation of the data fusion algorithm for sensor networks introduced in [2]. The considered system consists of multiple sensors monitoring one target. Every sensor p estimates the position of the target $\hat{x}^{\mathrm{loc,p}}$ using local measurements y_p. All local position estimates are joined into a regional estimate $\hat{x}^{\mathrm{loc,p}}$ in the data fusion step using weights (see (3.18) for details). The gains K_p and weights W_p are calculated offline.

and the measurements are taken using an active mobile architecture. Similar to (2.3) and (2.5), the dynamics of the mobile device and the measurement equation of the sensor nodes $\mathcal{S}$ are given by the time invariant stochastic linear model

$$\begin{aligned} x(k+1) &= Ax(k) + w(k) \\ y_p(k) &= C_p x(k) + v(k) \quad \forall p \in \mathcal{S}. \end{aligned} \tag{3.17}$$

When state estimation is done in a system like this, usually every sensor has its individual Kalman filter using only local measurements y_p. This leads to a decoupled estimation, where every node uses an algorithm similar to Fig. 3.1. In [2] a communication step is introduced to couple the state estimation in such a system. This algorithm is schematically shown in Fig. 3.3. It can be seen in Fig. 3.3 that all local estimates $\hat{x}^{\mathrm{loc,p}}(k|k)$ of the state $x(k)$ are communicated among neighboring sensors and merged based on a weighted graph. This leads to the following observation equations for every node $p \in \mathcal{S}$:

$$\begin{aligned} \hat{x}^{\mathrm{loc,p}}(k|k) &= \hat{x}^{\mathrm{loc,p}}(k|k-1) + K_p\left(y_p - C_p\hat{x}^{\mathrm{loc,p}}\right) \\ \hat{x}^{\mathrm{reg,p}}(k|k) &= \sum_{q\in\mathcal{S}_{[p]}} w_{pq}\hat{x}^{\mathrm{loc,q}} \\ \hat{x}^{\mathrm{loc,p}}(k+1|k) &= A\hat{x}^{\mathrm{reg,p}}(k|k) \end{aligned} \tag{3.18}$$

where $\hat{x}^{\mathrm{loc,p}}$ is the local estimate of the state $x(k)$ of the mobile device calculated by sensor node p. The Kalman gains K_p $\forall p \in \mathcal{S}$ and the weights w_{pq} $\forall p, q \in \mathcal{S}$ are optimized offline in [2]. Please refer to [1] or [2] for details on this optimization procedure.

3.3.2 Data Fusion for Multi Agent Systems

In this section a distributed Kalman Filter is derived, which uses the concept of joined estimates for the system under consideration. The idea is, to let every node estimate the state of itself and its neighbors. If this is done, more estimates of one state variable are available and can be joined using the algorithm introduced in Section 3.3.1. In order to use this idea for the introduced systems, the given model needs to be extended. The new local state $\xi_i \in \mathbb{R}^{n_{[i]}}$ of every node is now given by the former subsystem state $x_{[i]}$ leading to the extended local system equation

$$\xi_i(k) = A_{[i]}(k)\xi_i(k-1) + w_{[i]} \quad \text{with } \xi_i(0) = \xi_i^0. \tag{3.19a}$$

Since the measurements available in the network are fixed, the new measurement equation is given by (2.7) using $\xi_i = x_{[i]}$:

$$y_i(k) = C_{[i]}(k)\xi_i(k) + v_i(k). \tag{3.19b}$$

Using (3.19) the extended global model can be given in matrix form with

$$\begin{aligned} \boldsymbol{\xi}(k) &= \bar{\mathbf{A}}(k)\boldsymbol{\xi}(k-1) + \bar{\mathbf{w}} \quad \text{with } \boldsymbol{\xi}(0) = \boldsymbol{\xi}^0 \\ \mathbf{y} &= \bar{\mathbf{C}}(k)\boldsymbol{\xi}(k) + \mathbf{v}(k) \end{aligned} \tag{3.20}$$

using

$$\bar{\mathbf{w}} = \begin{bmatrix} {w_{[1]}}^T & {w_{[2]}}^T & \dots & {w_{[\eta]}}^T \end{bmatrix}^T$$

$$\bar{\mathbf{A}}(k) = \begin{bmatrix} A_{[1]} & & 0 \\ & \ddots & \\ 0 & & A_{[\zeta_i]} \end{bmatrix} \qquad \bar{\mathbf{C}}(k) = \begin{bmatrix} C_{[1]} & & 0 \\ & \ddots & \\ 0 & & C_{[\zeta_i]} \end{bmatrix}.$$

In (3.20) $\boldsymbol{\xi} \in \mathbb{R}^{\bar{n}}$, $\bar{n}_k = \sum_{i=1}^{\eta} n_{[i]}$ is the state vector[1] of the extended system and $\bar{\mathbf{w}} \in \mathbb{R}^{\bar{n}}$ is the process noise with covariance $\bar{\mathbf{Q}} = E[\bar{\mathbf{w}}\bar{\mathbf{w}}^T]$.

Example: (Three Agent Network) *For the three agent network introduced in Section 2 the extended model equations become*

$$\begin{pmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \\ \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{bmatrix} \\ \begin{bmatrix} x_2(k) \\ x_3(k) \end{bmatrix} \end{pmatrix} = \begin{pmatrix} A_{[1]} & 0 & 0 \\ 0 & A_{[2]} & 0 \\ 0 & 0 & A_{[3]} \end{pmatrix} \begin{pmatrix} \begin{bmatrix} x_1(k-1) \\ x_2(k-1) \end{bmatrix} \\ \begin{bmatrix} x_1(k-1) \\ x_2(k-1) \\ x_3(k-1) \end{bmatrix} \\ \begin{bmatrix} x_2(k-1) \\ x_3(k-1) \end{bmatrix} \end{pmatrix} \begin{pmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \\ \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \\ \begin{bmatrix} w_2 \\ w_3 \end{bmatrix} \end{pmatrix}$$

[1] To simplify notation, the dependency of $\bar{n}_k$ on the discrete time k will be dropped from now on.

$$\begin{pmatrix} y_1(k)\\ y_2(k)\\ y_3(k)\end{pmatrix} = \begin{pmatrix} C_{[1]} & 0 & 0\\ 0 & C_{[2]} & 0\\ 0 & 0 & C_{[3]}\end{pmatrix} \begin{pmatrix} \begin{bmatrix} x_1(k)\\ x_2(k)\end{bmatrix}\\ \begin{bmatrix} x_1(k)\\ x_2(k)\\ x_3(k)\end{bmatrix}\\ \begin{bmatrix} x_2(k)\\ x_3(k)\end{bmatrix}\end{pmatrix} + \begin{pmatrix} v_1\\ v_2\\ v_3\end{pmatrix}$$

When the system given in (3.20) is used for state estimation, a distributed Kalman Filter could be derived analogously to Section 3.2, giving the following local estimation equations:

$$\begin{aligned} &\hat{\xi}_i(k|k) = \hat{\xi}_i^{\text{loc}}(k|k-1) + \bar{K}_i(k)\left(y_i(k) - C_{[i]}(k)\hat{\xi}_i^{\text{loc}}(k|k-1)\right)\\ &\hat{\xi}_i^{\text{loc}}(k+1|k) = A_{[i]}(k)\hat{\xi}_i^{\text{loc}}(k|k) \quad \text{with } \hat{\xi}_i^{\text{loc}}(1|0) = \hat{x}_{[i]}^0 \end{aligned} \tag{3.21}$$

where the local state estimate $\hat{\xi}_i^{\text{loc}}(k|k)$ of node $\mathcal{N}_i$ is a column vector containing the state estimates $\hat{x}_{ji}^{\text{loc}}$ $\forall j \in \mathcal{N}_{[i]}$ of its neighbors $\mathcal{N}_j$. When looking at (3.21) one can see, that this Kalman Filter would be completely decoupled among different nodes. Therefore a data fusion step, analogously to the Kalman Filter introduced in [2], is added to increase the performance. The resulting algorithm has the same structure as the one shown in Fig. 3.3 but uses a generally time dependent system model. The new algorithm is visualized in Fig. 3.4. It can be seen in Fig. 3.4 that the local state estimates $\hat{\xi}_i^{\text{loc}}(k|k)$ are communicated among neighbors and joined to a regional state estimate $\hat{\xi}_i^{\text{reg}}(k|k)$ using weights, similar to the fusion algorithm in Section 3.3.1. When a data fusion step is introduced in (3.21) the following local estimation equations can be derived

$$\begin{aligned} &\hat{\xi}_i^{\text{loc}}(k|k) = \hat{\xi}_i^{\text{loc}}(k|k-1) + \bar{K}_i(k)\left(y_i(k) - C_{[i]}(k)\hat{\xi}_i^{\text{loc}}(k|k-1)\right)\\ &\hat{\xi}_i^{\text{reg}}(k|k) = W_i(k)\hat{\boldsymbol{\xi}}^{\mathbf{loc}}(k|k)\\ &\hat{\xi}_i^{\text{loc}}(k+1|k) = A_{[i]}(k)\hat{\xi}_i^{\text{reg}}(k|k) \quad \text{with } \hat{\xi}_i^{\text{loc}}(1|0) = \hat{x}_{[i]}^0 \end{aligned} \tag{3.22}$$

where $\hat{\boldsymbol{\xi}}^{\mathbf{loc}}(k|k) = \begin{bmatrix}\hat{\xi}_1^{\text{loc}T} & \hat{\xi}_2^{\text{loc}T} & \dots & \hat{\xi}_\eta^{\text{loc}T}\end{bmatrix}^T$ is a column vector containing all local state estimates. The weighting matrix W_i in (3.22) contains the the sub matrices $W_{ji} \subset W_i$ if $j \in \mathcal{N}_{[i]}$, one under the other. The single weights W_{ji} are given by

$$W_{ji} = \begin{bmatrix}\Omega_i^1 & \dots & \Omega_i^\eta\end{bmatrix} \tag{3.23}$$

$$\begin{aligned} &\text{using} \quad \Omega_i^j = \begin{bmatrix}\Omega_i^{1j} & \dots & \Omega_i^{\zeta_j j}\end{bmatrix}\\ &\text{with} \quad \Omega_i^{lj} = \begin{cases} w_i^{lj} & \text{if } \mathcal{N}_{[i]}(j) = \mathcal{N}_{[j]}(l)\\ \mathbf{0}_{n_p \times n_p} & \text{otherwise} \end{cases} \end{aligned}$$

where $p = \mathcal{N}_{[j]}(l)$ describes the l^{th} element of the set $\mathcal{N}_{[j]}$. In (3.23) w_i^{lj} is the weight for the state estimate of $\mathcal{N}_l$ calculated by $\mathcal{N}_j$ and used by $\mathcal{N}_i$. In order to get an unbiased estimate,

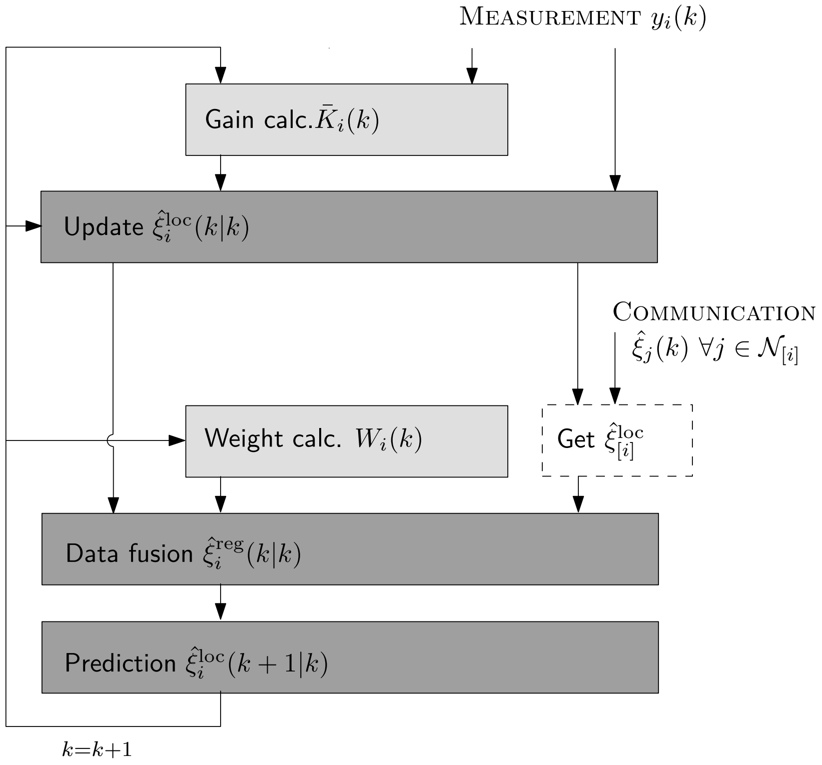

MEASUREMENT $y_i(k)$
Gain calc.$\bar{K}_i(k)$
Update $\hat{\xi}_i^{\text{loc}}(k|k)$
COMMUNICATION
$\hat{\xi}_j(k)\ \forall j \in \mathcal{N}_{[i]}$
Weight calc. $W_i(k)$
Get $\hat{\xi}_{[i]}^{\text{loc}}$
Data fusion $\hat{\xi}_i^{\text{reg}}(k|k)$
Prediction $\hat{\xi}_i^{\text{loc}}(k+1|k)$
$k{=}k{+}1$

Fig. 3.4: Schematic representation of the distributed Kalman Filter using data fusion for the systems under consideration. To generate multiple estimates of one position, every node estimates its own position and the position of its neighbors. This is modeled using the extended state variables $\hat{\xi}_i^{\text{loc}}$ (3.19). All position estimates are communicated and joined using weights. This generates regional estimates $\hat{\xi}_i^{\text{reg}}$ which are used for prediction (see (3.22) for details).

the estimation equation has to fulfill (3.2a), which leads to the following constrain for W_{ji}

$$W_{ji}(k)\mathbf{I}_{\bar{n}\times n_j} = \mathbf{I}_{n_j\times n_j}. \tag{3.24}$$

Example: (Three Agent Network) *In the three agent network the vector $\hat{\boldsymbol{\xi}}^{\mathbf{loc}}$ of local estimates and the weighting matrix $\mathbf{W}$ have the following form*

$$\hat{\boldsymbol{\xi}}^{\mathbf{loc}} = \left(\begin{bmatrix}\hat{x}_{11}^{\text{loc}} & \hat{x}_{21}^{\text{loc}}\end{bmatrix} \begin{bmatrix}\hat{x}_{12}^{\text{loc}} & \hat{x}_{22}^{\text{loc}} & \hat{x}_{32}^{\text{loc}}\end{bmatrix} \begin{bmatrix}\hat{x}_{23}^{\text{loc}} & \hat{x}_{33}^{\text{loc}}\end{bmatrix}\right)^T$$

$$\mathbf{W} = \begin{pmatrix} \begin{bmatrix} W_{11} \\ W_{21} \end{bmatrix} \\ \\ \begin{bmatrix} W_{12} \\ W_{22} \\ W_{32} \end{bmatrix} \\ \\ \begin{bmatrix} W_{23} \\ W_{33} \end{bmatrix} \end{pmatrix} = \begin{pmatrix} w_1^{11} & \mathbf{0} & w_1^{12} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & w_1^{21} & \mathbf{0} & w_1^{22} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \\ w_2^{11} & \mathbf{0} & w_2^{12} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & w_2^{21} & \mathbf{0} & w_2^{22} & \mathbf{0} & w_2^{23} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & w_2^{32} & \mathbf{0} & w_2^{33} \\ \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & w_3^{22} & \mathbf{0} & w_3^{23} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & w_3^{32} & \mathbf{0} & w_3^{33} \end{pmatrix}$$

It can be seen in (3.23) that the weighting matrices W_{ji} includes many zero columns. Therefore (3.22) can be rewritten for every estimate $\hat{x}_{ji}^{\mathrm{loc}} \subset \hat{\xi}_i^{\mathrm{reg}}\ \forall j \in \mathcal{N}_{[i]}$ in

$$\begin{aligned} \hat{x}_{ji}^{\mathrm{loc}}(k|k) &= W_{ji}(k) G_{ji}^{\xi}(k)^T G_{ji}^{\xi}(k) \hat{\boldsymbol{\xi}}^{\mathbf{loc}}(k|k) \\ &= W_{[ji]}(k) \hat{x}_{[ji]}^{\mathrm{loc}}(k|k) \end{aligned} \tag{3.25}$$

where $\hat{x}_{[ji]}^{\mathrm{loc}}$ contains all state estimates of node $\mathcal{N}_j$ which where communicated to node $\mathcal{N}_i$ by its neighbors $\mathcal{N}_j\ \forall j \in \mathcal{N}_{[i]}$. The selection matrix G_{ji}^{ξ} is given by

$$G_{ji}^{\xi} = \begin{bmatrix} H_i^{1j} & \quad H_i^{\eta j} \end{bmatrix} \tag{3.26}$$

$$\text{using} \quad H_i^{lj} = \begin{bmatrix} h_{1,1} & & h_{1,q_m} \\ & \ddots & \\ h_{p_m,1} & & h_{p_m,q_m} \end{bmatrix}$$

$$\text{with} \quad h_{p,q} = \begin{cases} \mathbf{I}_{n_J \times n_J} & \text{if } {\scriptstyle (\mathcal{N}_{[L]}(p)=L) \cap (\mathcal{N}_{[L]}(q)=J)} \\ & \quad {\scriptstyle L=\mathcal{N}_{[i]}(l),\ J=\mathcal{N}_{[i]}(j)} \\ \mathbf{0}_{n_J \times n_J} & \text{otherwise} \end{cases}$$

with p_m and q_m being the number of common neighbors $\mathcal{N}_{[i]} \cap \mathcal{N}_{[J]}$ and $\mathcal{N}_{[i]} \cap \mathcal{N}_{[L]}$, respectively. Using $G_{ji}^{\xi}(k)$ the constrain (3.24) can be rewritten in

$$W_{[ji]}(k) e(k) = \mathbf{I}_{n_j \times n_j} \qquad \text{with} \quad e(k) = G_{ji}^{\xi}(k) \mathbf{I}_{\bar{n} \times n_j} \tag{3.27}$$

Using (3.25) and introducing the column vector $\hat{\xi}_{[i]}^{\mathrm{loc}}$ with $\hat{x}_{[ji]}^{\mathrm{loc}} \in \hat{\xi}_{[i]}^{\mathrm{loc}}$ if $j \in \mathcal{N}_{[i]}$, the local estimation equations can be written, equivalent to (3.22), as

$$\begin{aligned} &\hat{\xi}_i^{\mathrm{loc}}(k|k) = \hat{\xi}_i^{\mathrm{loc}}(k|k-1) + \bar{K}_i(k) \left(y_i(k) - C_{[i]}(k) \hat{\xi}_i^{\mathrm{loc}}(k|k-1) \right) \\ &\hat{\xi}_i^{\mathrm{reg}}(k|k) = W_{[i]}(k) \hat{\xi}_{[i]}^{\mathrm{loc}}(k|k) \\ &\hat{\xi}_i^{\mathrm{loc}}(k+1|k) = A_{[i]}(k) \hat{\xi}_i^{\mathrm{reg}}(k|k) \quad \text{with } \hat{\xi}_i^{\mathrm{loc}}(1|0) = \hat{x}_{[i]}^0 \end{aligned} \tag{3.28}$$

where $W_{[i]}$ is a block diagonal matrix containing $W_{[ji]} \in W_{[i]}$ if $j \in \mathcal{N}_{[i]}$. The schematic representation of (3.28) is shown in Fig. 3.4. When (3.28) is written in matrix form

$$\begin{aligned} \hat{\boldsymbol{\xi}}^{\mathbf{loc}}(k|k) &= \hat{\boldsymbol{\xi}}^{\mathbf{reg}}(k|k-1) + \bar{\mathbf{K}}(k)(\mathbf{y} - \bar{\mathbf{C}}(k) \hat{\boldsymbol{\xi}}^{\mathbf{reg}}(k|k-1)) \\ \hat{\boldsymbol{\xi}}^{\mathbf{reg}}(k|k) &= \mathbf{W}(k) \hat{\boldsymbol{\xi}}^{\mathbf{loc}}(k|k) \\ \hat{\boldsymbol{\xi}}^{\mathbf{reg}}(k+1|k) &= \bar{\mathbf{A}}(k) \hat{\boldsymbol{\xi}}^{\mathbf{reg}}(k|k) \quad \text{with } \hat{\boldsymbol{\xi}}^{\mathbf{loc}}(1|0) = \hat{\mathbf{x}}^0 \end{aligned} \tag{3.29}$$

with $\mathbf{W} = \begin{bmatrix} {W_1}^T & \dots & {W_\eta}^T \end{bmatrix}^T$. The Kalman Matrix $\bar{\mathbf{K}}(k)$ becomes block diagonal in (3.29) with diag$\{\bar{\mathbf{K}}\} = \begin{bmatrix} \bar{K}_1 & \dots & \bar{K}_\eta \end{bmatrix}$.

Example: (Three Agent Network) *Using the selection matrices G_{ji}^{ξ} the zeros in $\mathbf{W}$ can be eliminated giving the following reduced state vectors and weighting matrices for the single systems:*

$$\begin{aligned}
\hat{x}_{[11]}^{\text{loc}} &= \begin{pmatrix}\hat{x}_{11}^{\text{loc}} & \hat{x}_{12}^{\text{loc}}\end{pmatrix}^T & W_{[11]} &= \begin{pmatrix}w_1^{11} & w_1^{12}\end{pmatrix} \\
\hat{x}_{[21]}^{\text{loc}} &= \begin{pmatrix}\hat{x}_{21}^{\text{loc}} & \hat{x}_{22}^{\text{loc}}\end{pmatrix}^T & W_{[21]} &= \begin{pmatrix}w_1^{21} & w_1^{22}\end{pmatrix} \\
\hat{x}_{[12]}^{\text{loc}} &= \begin{pmatrix}\hat{x}_{11}^{\text{loc}} & \hat{x}_{12}^{\text{loc}}\end{pmatrix}^T & W_{[11]} &= \begin{pmatrix}w_2^{11} & w_2^{12}\end{pmatrix} \\
\hat{\xi}_{[22]}^{\text{loc}} &= \begin{pmatrix}\hat{x}_{21}^{\text{loc}} & \hat{x}_{22}^{\text{loc}} & \hat{x}_{23}^{\text{loc}}\end{pmatrix}^T & W_{[22]} &= \begin{pmatrix}w_2^{21} & w_2^{22} & w_2^{23}\end{pmatrix} \\
\hat{\xi}_{[32]}^{\text{loc}} &= \begin{pmatrix}\hat{x}_{22}^{\text{loc}} & \hat{x}_{23}^{\text{loc}}\end{pmatrix}^T & W_{[32]} &= \begin{pmatrix}w_2^{22} & w_2^{23}\end{pmatrix} \\
\hat{\xi}_{[23]}^{\text{loc}} &= \begin{pmatrix}\hat{x}_{22}^{\text{loc}} & \hat{x}_{23}^{\text{loc}}\end{pmatrix}^T & W_{[23]} &= \begin{pmatrix}w_3^{22} & w_3^{23}\end{pmatrix} \\
\hat{\xi}_{[33]}^{\text{loc}} &= \begin{pmatrix}\hat{x}_{32}^{\text{loc}} & \hat{x}_{33}^{\text{loc}}\end{pmatrix}^T & W_{[33]} &= \begin{pmatrix}w_3^{32} & w_3^{33}\end{pmatrix}.
\end{aligned}$$

Using the extended model equations (3.20) and the estimation equations (3.22) the covariance matrices $\bar{\mathbf{P}}$ of the estimation error $\tilde{\boldsymbol{\xi}}$ are given by

$$\begin{aligned}
\bar{\mathbf{P}}(k|k) = {} & \mathbf{W}(k)\left[\mathbf{I} - \bar{\mathbf{K}}(k)\bar{\mathbf{C}}(k)\right]\bar{\mathbf{P}}(k|k-1) \\
& \left[\mathbf{I} - \bar{\mathbf{C}}(k)^T\bar{\mathbf{K}}(k)^T\right]\mathbf{W}(k)^T \\
& + \mathbf{W}(k)\bar{\mathbf{K}}(k)\mathbf{R}(k)\bar{\mathbf{K}}(k)^T\mathbf{W}(k)^T \\
\bar{\mathbf{P}}(k+|k) = {} & \bar{\mathbf{A}}(k)\bar{\mathbf{P}}(k|k)\bar{\mathbf{A}}(k) + \bar{\mathbf{Q}}(k).
\end{aligned} \tag{3.30}$$

To isolate the free parameters of $\bar{\mathbf{K}}(k)$ for the optimization problem (3.2b) again a matrix decomposition was used.

$$\bar{\mathbf{K}}(k) = \sum_{i=1}^{\eta} \bar{U}_i(k)^T \bar{K}_i(k) \bar{V}_i(k) \tag{3.31}$$

$$\begin{aligned}
\text{with} \quad \bar{U}_i(k) &= \left[\mathbf{0}_{n_{[i]}\times n_{[i]}(i-1)} \quad \mathbf{I}_{n_{[i]}\times n_{[i]}} \quad \mathbf{0}_{n_{[i]}\times n_{[i]}(\eta-i)}\right] \\
\bar{V}_i(k) &= \left[\mathbf{0}_{m_i\times\sum_{j=1}^{i-1} m_j} \quad \mathbf{I}_{m_i\times m_i} \quad \mathbf{0}_{m_i\times\sum_{j=i+1}^{\eta} m_j}\right]
\end{aligned}$$

where the matrices $\mathbf{0}$ and $\mathbf{I}$ are the zero and the identity matrix respectively. Using the the optimality constrain (3.24) on the weights and matrix decomposition (3.31) in (3.2b) one gets the following optimization problem

$$\begin{aligned}
[\bar{\mathbf{K}}(k),\ \mathbf{W}(k)] &= \underset{\bar{\mathbf{K}}(k)}{\operatorname{argmin}}\ \operatorname{tr}(\bar{\mathbf{P}}(k|k)) \\
\text{s.t.} \quad \bar{\mathbf{K}}(k) &= \sum_{i=1}^{\eta} \bar{U}_i(k)^T \bar{K}_i(k) \bar{V}_i(k) \\
\mathbf{W}(k)\mathbf{I}_{\bar{n}\times n_i} &= \mathbf{I}_{n\times n_i}.
\end{aligned} \tag{3.32}$$

Example: (Three Agent Network) *For the three agent network the Kalman Matrix* $\bar{\mathbf{K}}$ *becomes*

$$\bar{\mathbf{K}} = \begin{bmatrix} \begin{bmatrix} \bar{K}_{11} \\ \bar{K}_{21} \end{bmatrix} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \begin{bmatrix} \bar{K}_{12} \\ \bar{K}_{22} \\ \bar{K}_{32} \end{bmatrix} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \begin{bmatrix} \bar{K}_{23} \\ \bar{K}_{33} \end{bmatrix} \end{bmatrix} \quad \textit{with} \quad \begin{aligned} \bar{K}_1 &= \begin{bmatrix} \bar{K}_{11} \\ \bar{K}_{21} \end{bmatrix}, \ dim(\bar{K}_1) = 5 \times 2 \\ \bar{K}_2 &= \begin{bmatrix} \bar{K}_{12} \\ \bar{K}_{22} \\ \bar{K}_{32} \end{bmatrix}, \ dim(\bar{K}_2) = 7 \times 3 \\ \bar{K}_3 &= \begin{bmatrix} \bar{K}_{23} \\ \bar{K}_{33} \end{bmatrix}, \ dim(\bar{K}_3) = 5 \times 3. \end{aligned}$$

It is shown in [2] that an integrated solution of this problem with respect to both constrains is not possible. Furthermore the optimality constrain on $\mathbf{W}$ in (3.32) can be divided in one constrain for every sub matrix as given in (3.27). Therefore (3.32) can be separated in

$$\begin{aligned} \bar{\mathbf{K}}(k) &= \underset{\bar{\mathbf{K}}(k)}{\operatorname{argmin}} \operatorname{tr}(\bar{\mathbf{P}}(k|k)) \\ \text{s.t.} \quad \bar{\mathbf{K}}(k) &= \sum_{i=1}^{\eta} \bar{U}_i(k)^T \bar{K}_i(k) \bar{V}_i(k) \\ \mathbf{W}(k) &= \mathbf{W}(k-1) \end{aligned} \tag{3.33}$$

and $\forall i \in \mathcal{N}_{[i]}$

$$\begin{aligned} W_{[ji]}(k) &= \underset{W_{[ji]}(k)}{\operatorname{argmin}} \operatorname{tr}\left(W_{[ji]}(k) \Phi_{ji}(k) W_{[ji]}(k)^T \right) \\ \text{s.t.} \quad W_{[ji]}(k) e(k) &= \mathbf{I}_{n_j \times n_j} \end{aligned} \tag{3.34}$$

with

$$\begin{aligned} e(k) &= G^{\xi}_{ji}(k) \mathbf{I}_{\bar{n} \times n_j} \\ \Phi_{ji}(k) &= G^{\xi}_{ji}(k) \left[\mathbf{I} - \bar{\mathbf{K}}(k) \bar{\mathbf{C}}(k) \right] \bar{\mathbf{P}}(k|k-1) \\ & \quad \left[\mathbf{I} - \bar{\mathbf{C}}(k)^T \bar{\mathbf{K}}(k)^T \right] G^{\xi}_{ji}(k)^T \\ & \quad + \ G^{\xi}_{ji}(k) \bar{\mathbf{K}}(k) \mathbf{R}(k) \bar{\mathbf{K}}(k)^T G^{\xi}_{ji}(k)^T. \end{aligned}$$

It is shown in Appendix A.4 that the optimal solution to the optimization problem (3.33) is given by[2]

$$\begin{bmatrix} \operatorname{vec}[\bar{K}_1] \\ \vdots \\ \operatorname{vec}[\bar{K}_\eta] \end{bmatrix} = - \begin{bmatrix} \bar{\Upsilon}_{11}{}^T \otimes \bar{\Xi}_{11} & \dots & \bar{\Upsilon}_{1\eta}{}^T \otimes \bar{\Xi}_{1\eta} \\ \vdots & \ddots & \vdots \\ \bar{\Upsilon}_{\eta 1}{}^T \otimes \bar{\Xi}_{\eta 1} & \dots & \bar{\Upsilon}_{\eta\eta}{}^T \otimes \bar{\Xi}_{\eta\eta} \end{bmatrix}^{-1} \begin{bmatrix} \bar{\Psi}_1 \\ \vdots \\ \bar{\Psi}_\eta \end{bmatrix} \tag{3.35}$$

[2] All variables $\bar{K}$, $\bar{\Upsilon}$, $\bar{\Xi}$, $\bar{\Psi}$ are dependent on the discrete time k in (3.35)

with

$$\begin{aligned}
\bar{\Psi}_{ij}(k) &= -\mathrm{vec}[\bar{U}_i \mathbf{W}^T \mathbf{W} \bar{\mathbf{P}}(k|k-1) \bar{\mathbf{C}} i^T \bar{V}_i^T] \\
\bar{\Xi}_{ij}(k) &= \bar{U}_i(k) \mathbf{W}(k-1)^T \mathbf{W}(k-1) \bar{U}_j(k)^T \\
\bar{\Upsilon}_{ij}(k) &= \bar{V}_j \bar{\mathbf{C}} \bar{\mathbf{P}}(k|k-1) \bar{\mathbf{C}}^T \bar{V}_i^T + \bar{V}_j \mathbf{R} \bar{V}_i^T.
\end{aligned}$$

The solution to the optimization problem (3.34), also derived in Appendix A.4, becomes

$$W_{[ji]}(k) = \delta(k)^T \left(\mathbf{I} - \Gamma_1^0(k)(\Gamma_1^0(k))^T\right) \tag{3.36}$$

$$\begin{aligned}
\text{with} \quad \delta(k) &= (\Phi_{ji}(k))^{-1} e(k) \left(e(k)^T (\Phi_{ji}(k))^{-1} e(k)\right)^{-1} \\
\Gamma^0(k) &= \mathrm{Null}\left(\begin{bmatrix} \Phi_{ji}(k) & e(k) \\ e(k)^T & \mathbf{0}_{n_i \times n_i} \end{bmatrix}\right) \\
\Gamma_1^0(k) &= \Gamma^0[1:n_{[i]},:](k)
\end{aligned}$$

where $\mathrm{Null}(.)$ denotes a matrix containing the vectors spanning its null space and $.[1:n,:]$ denotes the first n lines of a matrix.

Offline Gain Calculation

It can be seen in (3.35) and (3.36) that the calculation of $\bar{K}_i$ and $W_{[i]}$ is highly coupled between nodes and that even information from outside the subsystem $\mathcal{N}_{[i]}$ are needed to calculate $\hat{\xi}_i^{\mathrm{loc}}$. Therefore this approach can only be used for special applications where the graph of the network is static, the matrices $\bar{\mathbf{A}}$ and $\bar{\mathbf{C}}$ are time independent and assumptions about the covariances $\mathbf{R}$ and $\bar{\mathbf{Q}}$ can be made a priori. Then $\bar{K}_i$ and $W_{[i]}$ can be calculated offline and provided to the mobile devices before they start operating. This leads to the following algorithm:

dKF with data fusion (offline gain calculation)

$$\text{offline:} \quad \bar{\mathbf{K}}(k) = -\begin{bmatrix} \bar{\Upsilon}_{11}{}^T \otimes \bar{\Xi}_{11} & \dots & \bar{\Upsilon}_{1\eta}{}^T \otimes \bar{\Xi}_{1\eta} \\ \vdots & \ddots & \vdots \\ \bar{\Upsilon}_{\eta 1}{}^T \otimes \bar{\Xi}_{\eta 1} & \dots & \bar{\Upsilon}_{\eta\eta}{}^T \otimes \bar{\Xi}_{\eta\eta} \end{bmatrix}^{-1} \begin{bmatrix} \bar{\Psi}_1 \\ \vdots \\ \bar{\Psi}_\eta \end{bmatrix}$$

$$\begin{aligned}
W_{[ji]}(k) &= \delta(k)^T \left(\mathbf{I} - \Gamma_1^0(k)(\Gamma_1^0(k))^T\right) \\
\bar{\mathbf{P}}(k|k) &= \mathbf{W}(k) \left[\mathbf{I} - \bar{\mathbf{K}}(k)\bar{\mathbf{C}}\right] \mathbf{P}(k|k-1) \\
&\quad \left[\mathbf{I} - \bar{\mathbf{C}}^T \bar{\mathbf{K}}(k)^T\right] \mathbf{W}(k)^T \\
&\quad + \mathbf{W}(k)\bar{\mathbf{K}}(k)\mathbf{R}\mathbf{K}(k)^T \mathbf{W}(k)^T \\
\bar{\mathbf{P}}(k+1|k) &= \bar{\mathbf{A}}\bar{\mathbf{P}}(k|k)\bar{\mathbf{A}} + \bar{\mathbf{Q}} \quad \text{with } \bar{\mathbf{P}}(1|0) = \bar{\mathbf{P}}_0
\end{aligned} \tag{3.37}$$

$$\text{online:} \quad \hat{\xi}_i^{\mathrm{loc}}(k|k) = \hat{\xi}_i^{\mathrm{reg}}(k|k-1) + \bar{K}_i(k)\left(y_i(k) - C_{[i]}\hat{\xi}_i^{\mathrm{reg}}(k|k-1)\right)$$

$$\begin{aligned}
&\textsc{Communication}: \hat{\xi}_j(k)\ \forall j \in \mathcal{N}_{[i]}\\
&\hat{\xi}_i^{\mathrm{reg}}(k|k) = W_i(k)\hat{\boldsymbol{\xi}}^{\mathbf{loc}}(k|k)\\
&\hat{\xi}_i^{\mathrm{reg}}(k+1|k) = A_{[i]}\hat{\xi}_i^{\mathrm{reg}}(k|k) \quad \text{with } \hat{\xi}_i^{\mathrm{reg}}(1|0) = \hat{x}_{[i]}^0
\end{aligned}$$

Online Gain Calculation

To calculate $\bar{K}_i$ and W_i online, (3.35) and (3.36) need to be separable but the attempt to show this property failed. It will be shown in the next section how the whole problem can be rewritten to get an algorithm that can be calculated online.

3.3.3 Data Fusion for Dynamic Multi Agent Systems

In this section the algorithm introduced in Section 3.3.2 will be modified to make online gain calculation possible. The idea is to introduce a second communication step, similar to the distributed Kalman filter introduced in Section 3.2. This leads to a new algorithm, schematically shown in Fig. 3.5. It can be seen in Fig. 3.5 that the joined state estimates $\hat{x}_i(k+1|k)$ are now communicated to neighboring nodes. Therefore every node only needs to calculate a joined estimate of its own state $\hat{x}_i(k|k)$ and not of the complete subsystem $\hat{\xi}_i^{\mathrm{reg}}(k|k)$ as in Fig. 3.4. Using this framework the observation equations (3.22) for every single node $\mathcal{N}_i \in \mathcal{N}$ change to

$$\begin{aligned}
\hat{\xi}_i^{\mathrm{loc}}(k|k) &= \hat{\xi}_i^{\mathrm{loc}}(k|k-1) + \bar{K}_i(k)\left(y_i(k) - C_{[i]}(k)\hat{\xi}_i^{\mathrm{loc}}(k|k-1)\right)\\
&= \hat{x}_{[i]}(k|k-1) + \bar{K}_i(k)\tilde{y}_i(k)\\
\hat{x}_i(k|k) &= \bar{W}_i(k)\hat{\boldsymbol{\xi}}^{\mathbf{loc}}(k|k)\\
\hat{x}_i(k+1|k) &= A_i(k)\hat{x}_i(k|k) \quad \text{with } \hat{x}_i(1|0) = \hat{x}_i^0
\end{aligned} \tag{3.38}$$

where $\bar{W}_i$ is given by the lines of W_i that correspond to $\hat{x}_{ii}^{\mathrm{loc}}$ in $\hat{\xi}_i^{\mathrm{loc}}(k|k)$. When (3.38) is written in matrix, the following estimation equations are obtained

$$\begin{aligned}
\hat{\boldsymbol{\xi}}^{\mathbf{loc}}(k|k) &= \mathbf{G}^x(k)\hat{\mathbf{x}}(k|k-1) + \bar{\mathbf{K}}(k)\tilde{\mathbf{y}}(k)\\
\hat{\mathbf{x}}(k|k) &= \bar{\mathbf{W}}(k)\hat{\boldsymbol{\xi}}^{\mathbf{loc}}(k|k)\\
\hat{\mathbf{x}}(k+1|k) &= \mathbf{A}(k)\mathbf{x}(k|k) \quad \text{with } \hat{\mathbf{x}}(1|0) = \hat{\mathbf{x}}^0
\end{aligned} \tag{3.39}$$

with $\bar{\mathbf{W}} = \left[\bar{W}_1{}^T \ \ \dots \ \ \bar{W}_\eta{}^T\right]^T$. By substituting the first subequation of (3.39) into the second subequation, the estimation equation (3.39) can be rewritten in

$$\begin{aligned}
\hat{\mathbf{x}}(k|k) &= \bar{\mathbf{W}}(k)\mathbf{G}^x(k)\hat{\mathbf{x}}(k|k-1) + \bar{\mathbf{W}}(k)\bar{\mathbf{K}}(k)\tilde{\mathbf{y}}(k)\\
&= \hat{\mathbf{x}}(k|k-1) + \bar{\bar{\mathbf{K}}}(k)\tilde{\mathbf{y}}(k)\\
\hat{\mathbf{x}}(k+1|k) &= \mathbf{A}(k)\hat{\mathbf{x}}(k|k) \quad \text{with } \hat{\mathbf{x}}(1|0) = \hat{\mathbf{x}}^0
\end{aligned} \tag{3.40}$$

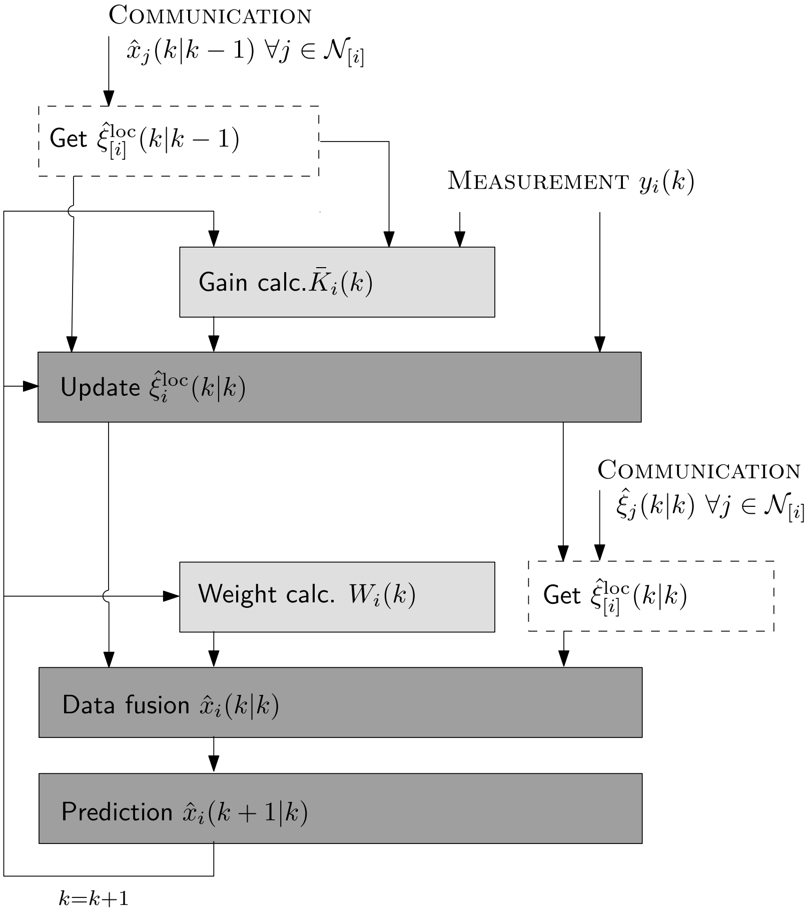

COMMUNICATION
$\hat{x}_j(k|k-1)\ \forall j \in \mathcal{N}_{[i]}$
Get $\hat{\xi}^{\mathrm{loc}}_{[i]}(k|k-1)$
MEASUREMENT $y_i(k)$
Gain calc.$\bar{K}_i(k)$
Update $\hat{\xi}^{\mathrm{loc}}_i(k|k)$
COMMUNICATION
$\hat{\xi}_j(k|k)\ \forall j \in \mathcal{N}_{[i]}$
Weight calc. $W_i(k)$
Get $\hat{\xi}^{\mathrm{loc}}_{[i]}(k|k)$
Data fusion $\hat{x}_i(k|k)$
Prediction $\hat{x}_i(k+1|k)$
$k{=}k{+}1$

Fig. 3.5: Schematic representation of the extended distributed Kalman Filter using data fusion. To simplify the filter given in Fig. 3.4, it is extended by an additional communication step. Consequently only the local state $x_i(k)$ has to be predicted by every node (3.38). To incorporate data fusion, still the extended state $\hat{\xi}^{\mathrm{loc}}_i(k|k)$ needs to be estimated.

$$\begin{aligned} \text{with} \quad & \bar{\bar{\mathbf{K}}}(k) = \bar{\mathbf{W}}(k)\bar{\mathbf{K}}(k) = \left[\bar{\bar{K}}_1(k)^T \quad \dots \quad \bar{\bar{K}}_\eta(k)^T\right]^T \\ \text{and} \quad & \bar{\mathbf{W}}(k)\mathbf{G}^x(k) = \mathbf{I}, \quad \text{due to (3.24).} \end{aligned}$$

Example: (Three Agent Network) *Using the weighting matrix* $\mathbf{W}$ *for this example, introduced in Section 3.3.2, the new weighting matrix* $\bar{\mathbf{W}}$ *becomes*

$$\bar{\mathbf{W}} = \begin{pmatrix} w^{11} & \mathbf{0} & w^{12} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & w^{21} & \mathbf{0} & w^{22} & \mathbf{0} & w^{23} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & w^{32} & \mathbf{0} & w^{33} \end{pmatrix}$$

Given the constrain that communication is only allowed between neighbors the local estimation equations (3.38) change to

$$\begin{aligned}\tilde{y}_i(k) &= y_i(k) - C_{[i]}\hat{x}_{[i]}(k|k-1)\\ \hat{x}_i(k|k) &= \hat{x}_i(k|k-1) + \bar{\bar{K}}_{[i]}(k)\tilde{y}_{[i]}(k)\\ \hat{x}_i(k-1|k) &= A_i(k)\hat{x}_i(k|k) \quad \text{with } \hat{x}_i(1|0) = \hat{x}_i^0\end{aligned} \tag{3.41}$$

$$\begin{aligned}\text{using} \quad \bar{\bar{K}}_{[i]}(k) &= \bar{\bar{K}}_i(k)G_i^y(k)^T\\ \tilde{y}_{[i]}(k) &= G_i^y(k)\tilde{\mathbf{y}}(k) \quad \text{with } \tilde{y}_j(k) \subset \tilde{y}_{[i]}(k) \text{ if } j \in \mathcal{N}_{[i]}.\end{aligned}$$

The selection matrix G_i^y in (3.41) is given by the nonzero lines of

$$G_I^y = \begin{bmatrix} g^{1,i} & & \mathbf{0}\\ & \ddots & \\ \mathbf{0} & & g^{\eta,i}\end{bmatrix} \quad \text{with } g^{j,i} = \begin{cases}\mathbf{I}_{m_{[j]}\times m_{[j]}} & \text{if } j \in \mathcal{N}_{[i]}\\ \mathbf{0}_{m_{[j]}\times m_{[j]}} & \text{otherwise}\end{cases} \quad .$$

The new algorithm given in (3.40) and (3.41) is visualized in Fig. 3.6. It can be seen in Fig. 3.6, that by introducing an additional communication step the observation algorithm reduces to a form that is similar to the first distributed Kalman Filter introduced in Section 3.2. Nevertheless (3.22) uses the measurement information of the whole subsystem $\mathcal{N}_{[i]}$ whereas (3.9) only uses local measurements. This is because in Fig. 3.6 the measurement prediction error $\tilde{y}_i$ is communicated among neighbors to calculate $\tilde{y}_{[i]}$ a priori to the update step.

Similar to (3.4) the covariance matrices $\mathbf{P}$ of the estimation error $\tilde{\mathbf{x}}$ are given by

$$\begin{aligned}\mathbf{P}(k|k) = &\left[\mathbf{I} - \bar{\bar{\mathbf{K}}}(k)\mathbf{C}(k)\right]\mathbf{P}(k|k-1)\left[\mathbf{I} - \mathbf{C}(k)^T\bar{\bar{\mathbf{K}}}(k)^T\right]\\ &+ \bar{\bar{\mathbf{K}}}(k)\mathbf{R}(k)\bar{\bar{\mathbf{K}}}(k)^T\\ \mathbf{P}(k+1|k) = &\ \mathbf{A}(k)\mathbf{P}(k|k)\mathbf{A}(k) + \mathbf{Q}(k).\end{aligned} \tag{3.42a}$$

Example: (Three Agent Network) *The Kalman matrix $\bar{\bar{\mathbf{K}}}$ of the three agent network has the following form*

$$\bar{\bar{\mathbf{K}}} = \begin{bmatrix} [\bar{\bar{K}}_{11} & \bar{\bar{K}}_{21}] & 0\\ [\bar{\bar{K}}_{12} & \bar{\bar{K}}_{22} & \bar{\bar{K}}_{32}]\\ 0 & [\bar{\bar{K}}_{23} & \bar{\bar{K}}_{33}]\end{bmatrix} \quad \textit{with} \quad \begin{aligned}\bar{\bar{K}}_1 &= \begin{bmatrix}\bar{\bar{K}}_{11} & \bar{\bar{K}}_{21}\end{bmatrix} & dim = 2\times 5\\ \bar{\bar{K}}_2 &= \begin{bmatrix}\bar{\bar{K}}_{12} & \bar{\bar{K}}_{22} & \bar{\bar{K}}_{32}\end{bmatrix} & dim = 2\times 8\\ \bar{\bar{K}}_3 &= \begin{bmatrix}\bar{\bar{K}}_{23} & \bar{\bar{K}}_{33}\end{bmatrix} & dim = 2\times 6.\end{aligned}$$

To isolate the free parameters of $\bar{\bar{\mathbf{K}}}(k)$ for the optimization problem (3.2b) again a matrix decomposition can be used

$$\bar{\bar{\mathbf{K}}}(k) = \sum_{i=1}^{\eta}\bar{\bar{U}}_i{}^T\bar{\bar{K}}_{[i]}(k)\bar{\bar{V}}_i(k) \tag{3.43}$$

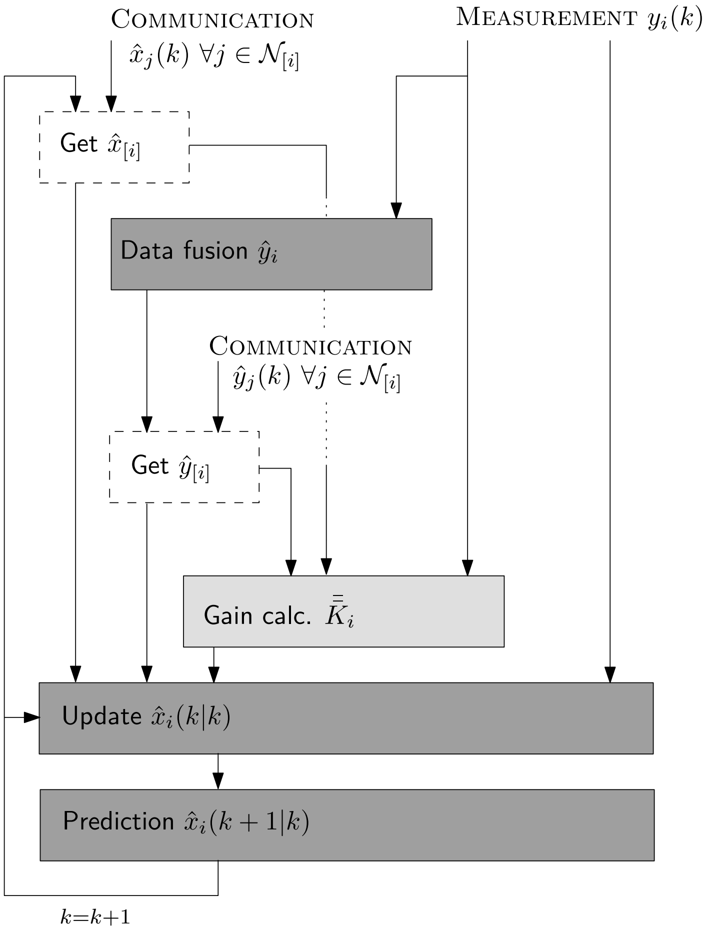

COMMUNICATION
$\hat{x}_j(k)\ \forall j \in \mathcal{N}_{[i]}$
MEASUREMENT $y_i(k)$
Get $\hat{x}_{[i]}$
Data fusion $\hat{y}_i$
COMMUNICATION
$\hat{y}_j(k)\ \forall j \in \mathcal{N}_{[i]}$
Get $\hat{y}_{[i]}$
Gain calc. $\bar{\bar{K}}_i$
Update $\hat{x}_i(k|k)$
Prediction $\hat{x}_i(k+1|k)$
$k{=}k{+}1$

Fig. 3.6: Schematic representation of the rearranged distributed Kalman Filter using data fusion and an additional communication step. It was derived in (3.40) that the Kalman Filter given in Fig. 3.5 can be rearranged as given in this figure. By using this filter, only the local state estimates $\hat{x}_i$ need to be calculated by every node (3.41). The data fusion is realized by additionally communicating the measurement error $\tilde{y}_i$ among neighbors.

$$\begin{aligned} \text{with} \quad & \bar{\bar{U}}_i = \begin{bmatrix} \mathbf{0}_{n_i \times n_i(i-1)} & \mathbf{I}_{n_i \times n_i} & \mathbf{0}_{n_i \times n_i(\eta - i)} \end{bmatrix} \\ & \bar{\bar{V}}_i(k) = G_i^y(k). \end{aligned}$$

The minimization problem (3.2b) is then given by

$$\begin{aligned} \bar{\bar{\mathbf{K}}}(k) &= \underset{\bar{\bar{\mathbf{K}}}(k)}{\operatorname{argmin}}\ \operatorname{tr}(\mathbf{P}(k|k)) \\ \text{s.t.} \quad \bar{\bar{\mathbf{K}}}(k) &= \sum_{i=1}^{\eta} \bar{\bar{U}}_i^T \bar{\bar{K}}_i(k) \bar{\bar{V}}_i(k) \end{aligned} \tag{3.44}$$

and can be solved similar to Section 3.2 leading to the optimal Kalman gains $\bar{\bar{K}}_i$

$$\bar{\bar{K}}_i(k) = -\bar{\bar{\Upsilon}}_i(k)\Big(\bar{\bar{\Psi}}_i(k)\Big)^{-1} \tag{3.45}$$

$$\begin{aligned}\text{with}\quad \bar{\bar{\Upsilon}}_i(k) &= \bar{\bar{U}}_i\mathbf{P}(k|k-1)\bar{\bar{C}}_{[i]}(k)^T\\ \bar{\bar{\Psi}}_i(k) &= \bar{\bar{C}}_{[i]}(k)\mathbf{P}(k|k-1)\bar{\bar{C}}_{[i]}(k)^T + R_{[i]}(k)\end{aligned}$$

where $\bar{\bar{C}}_{[i]} = G_i^y\mathbf{C}$ is the measurement matrix of the subsystem $\mathcal{N}_{[i]}$.

Offline Gain Calculation

If the graph of the network is static, the matrices $\bar{\mathbf{A}}$ and $\bar{\mathbf{C}}$ are time independent and assumptions about the covariances $\mathbf{R}$ and $\bar{\mathbf{Q}}$ can be made a priori, the Kalman gain can calculated offline using (3.45) with (3.42) directly. The algorithm is then given by

Extended dKF with data fusion (offline gain calculation)

$$\begin{aligned}\text{offline:}\quad &\bar{\bar{K}}_i(k) = \bar{\bar{U}}_i\mathbf{P}(k|k-1)\bar{\bar{C}}_{[i]}{}^T\left[\bar{\bar{C}}_{[i]}\mathbf{P}(k|k-1)\bar{\bar{C}}_{[i]}{}^T + R_{[i]}\right]^{-1}\\ &\mathbf{P}(k|k) = \left[\mathbf{I} - \bar{\bar{\mathbf{K}}}(k)\mathbf{C}\right]\mathbf{P}(k|k-1)\left[\mathbf{I} - \mathbf{C}^T\bar{\bar{\mathbf{K}}}(k)^T\right]\\ &\qquad\qquad + \bar{\bar{\mathbf{K}}}(k)\mathbf{R}\bar{\bar{\mathbf{K}}}(k)^T\\ &\mathbf{P}(k+|k) = \mathbf{A}\mathbf{P}(k|k)\mathbf{A} + \mathbf{Q} \quad \text{with} \quad \mathbf{P}(1|0) = \mathbf{P}^0\\ \\ \text{online:}\quad &\tilde{y}_i(k|k-1) = y_i(k) - C_{[i]}\hat{x}_{[i]}(k|k-1)\\ &\textsc{Communication} : \hat{y}_j(k)\ \forall j \in \mathcal{N}_{[i]}\\ &\hat{x}_i(k|k) = \hat{x}_i(k|k-1) + \bar{\bar{K}}_{[i]}(k)\tilde{y}_{[i]}(k)\\ &\hat{x}_i(k|k-1) = A_i\hat{x}_i(k-1|k-1) \quad \text{with} \quad \hat{x}_i(1|0) = \hat{x}_i^0\\ &\textsc{Communication} : \hat{x}_j(k)\ \forall j \in \mathcal{N}_{[i]}\end{aligned} \tag{3.46}$$

Online Gain Calculation

The structure of (3.45) is already decoupled and very similar to (3.12). Nevertheless the optimal Kalman gain (3.45) cannot be simplified similarly since $\bar{\bar{C}}_{[i]} \neq C_{[i]}$ and $C_j \subset \bar{\bar{C}}_{[i]}\ \forall j \in \mathcal{N}_{[i]}$. Therefore information even beyond those available in $\mathcal{N}_{[i]}$ are necessary to calculate $\bar{\bar{\Upsilon}}_i(k)$ and $\bar{\bar{\Psi}}_i(k)$ independently. That is why $\bar{\bar{\Upsilon}}_i(k)$ and $\bar{\bar{\Psi}}_i(k)$ need to be approximated using measurements. It can be found in the literature, e.g. [13], that the following equalities hold for a global Kalman Filter

$$\begin{aligned} E\left[\tilde{x}_i(k|k-1)\tilde{\mathbf{y}}(k)^T\right] &= \mathbf{P}(k|k-1)\mathbf{C}(k) \\ E\left[\tilde{\mathbf{y}}(k)\tilde{\mathbf{y}}(k)^T\right] &= \mathbf{C}(k)\mathbf{P}(k|k-1)\mathbf{C}(k)^T + \mathbf{R}(k). \end{aligned} \tag{3.47}$$

Therefore $\bar{\bar{\bar{\Upsilon}}}_i(k)$ and $\bar{\bar{\bar{\Psi}}}_i(k)$ can be approximated using

$$\begin{aligned} \bar{\bar{\bar{\Upsilon}}}_i(k) &= E\left[\tilde{x}_i(k|k-1)\tilde{y}_{[i]}(k)^T\right] \\ &\approx \frac{1}{k}\sum_{l=1}^{k}\Big((\overline{x}_i - \hat{x}_i(l|l-1))(\overline{y}_{[i]} - \hat{y}_{[i]}(l|l-1))^T\Big) \end{aligned} \tag{3.48a}$$

$$\bar{\bar{\bar{\Psi}}}_i(k) = E\left[\tilde{y}_{[i]}(k)\tilde{y}_{[i]}(k)^T\right]$$

$$\approx \frac{1}{k}\sum_{l=1}^{k}\Big((\overline{y}_{[i]} - \hat{y}_{[i]}(l|l-1))(\overline{y}_{[i]} - \hat{y}_{[i]}(l|l-1))^T\Big) \tag{3.48b}$$

with the initial conditions

$$\begin{aligned} \bar{\bar{\bar{\Upsilon}}}_i(0) &= \bar{\bar{\bar{U}}}_i\mathbf{P}^0\bar{\bar{\bar{C}}}_{[i]}(0) \\ \bar{\bar{\bar{\Psi}}}_i(0) &= \bar{\bar{\bar{C}}}_{[i]}(0)\mathbf{P}^0\bar{\bar{\bar{C}}}_{[i]}(0)^T + R_{[i]}(0). \end{aligned}$$

In (3.48) $\overline{x}_i$ and $\overline{y}_{[i]}$ are the mean values of $\hat{x}_i(k|k-1)$ and $\hat{y}_{[i]}(k)$, respectively. This leads to the fully distributed algorithm

Extended dKF with data fusion (online gain calculation)

$$\begin{aligned} &\tilde{y}_i(k) = y_i(k) - C_{[i]}\hat{x}_{[i]}(k|k-1) \\ &\textsc{Communication} : \hat{y}_j(k)\ \forall j \in \mathcal{N}_{[i]} \\ &\bar{\bar{\bar{\Upsilon}}}_i(k) = \frac{1}{k}\sum_{l=1}^{k}\Big((\overline{x}_i - \hat{x}_i(l|l-1))(\overline{y}_{[i]} - \hat{y}_{[i]}(l|l-1))^T\Big) \\ &\bar{\bar{\bar{\Psi}}}_i(k) = \frac{1}{k}\sum_{l=1}^{k}\Big((\overline{y}_{[i]} - \hat{y}_{[i]}(l|l-1))(\overline{y}_{[i]} - \hat{y}_{[i]}(l|l-1))^T\Big) \\ &\bar{\bar{\bar{K}}}_i(k) = -\bar{\bar{\bar{\Upsilon}}}_i(k)\big(\bar{\bar{\bar{\Psi}}}_i(k)\big)^{-1} \\ &\hat{x}_i(k|k) = \hat{x}_i(k|k-1) + \bar{\bar{\bar{K}}}_{[i]}(k)\tilde{y}_{[i]}(k) \\ &\hat{x}_i(k+1|k) = A_i\hat{x}_i(k|k) \quad \text{with} \quad \hat{x}_i(1|0) = \hat{x}_i^0 \\ &\textsc{Communication} : \hat{x}_j(k)\ \forall j \in \mathcal{N}_{[i]} \end{aligned} \tag{3.49}$$

Chapter 4

Numerical Example
Self-localization of Mobile Devices

To compare the performance of the distributed Kalman Filters, simulations were done in MATLAB. The investigated system of mobile agents estimates the position of every agent in a decentralized fashion. Communication and distance measurements are possible among neighboring mobile nodes and to reachable reference nodes. The set up for this numerical example and its system dynamics are given in detail in Section 4.1. Afterwards the specifications used in MATLAB are given and the obtained simulation results are described in Section 4.3.

4.1 System Dynamics

The derivation of the Kalman Filter algorithms in Section 3 was based on the communication topology and the general linear model of an arbitrary multi-agent system, both introduced in Section 2. These general system properties will be specified in this section for a system of robots who estimate their position using a distributed Kalman Filter algorithm. All assumptions made in Section 3 also hold for this system. To argue the validity of the generated results, the chosen parameter will be furthermore discussed in Section 4.2.

4.1.1 Network Properties

The investigated system consists of $\mu = 6$ reference nodes $\mathcal{M}$ and $\eta = 8$ mobile devices $\mathcal{N}$. Those are spread on a 2D surface as shown in Fig. 4.1. The positions $\boldsymbol{p} = [p^{\mathrm{x}}\ p^{\mathrm{y}}]^T$ of the reference nodes are chosen as given below

$$\begin{aligned} \check{\boldsymbol{p}}_1 &= \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \check{\boldsymbol{p}}_2 = \begin{bmatrix} 0 \\ 4 \end{bmatrix} \quad \check{\boldsymbol{p}}_3 = \begin{bmatrix} 0 \\ 8 \end{bmatrix} \\ \check{\boldsymbol{p}}_4 &= \begin{bmatrix} 4 \\ 0 \end{bmatrix} \quad \check{\boldsymbol{p}}_5 = \begin{bmatrix} 4 \\ 4 \end{bmatrix} \quad \check{\boldsymbol{p}}_6 = \begin{bmatrix} 4 \\ 8 \end{bmatrix} \end{aligned} \tag{4.1}$$

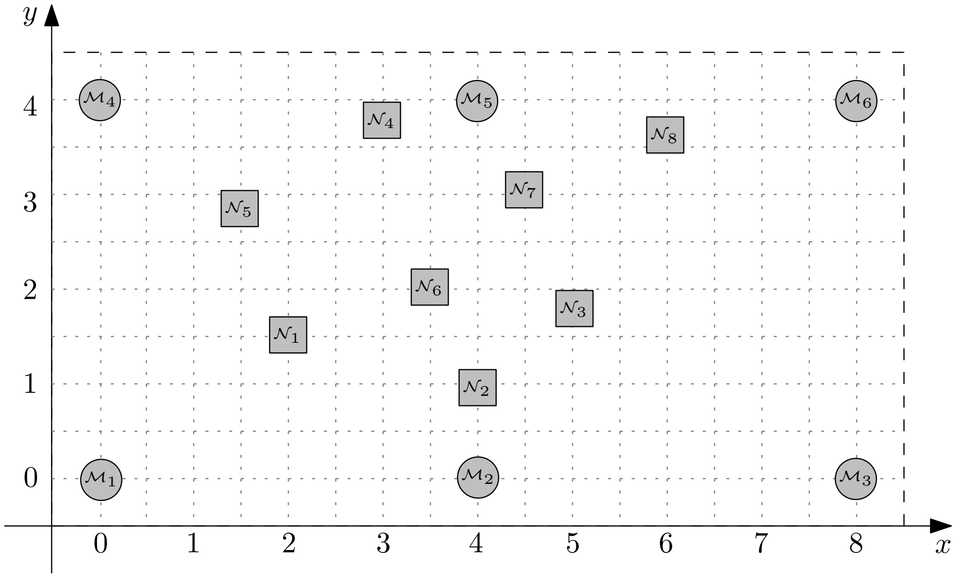


Fig. 4.1: Initial positions of mobile (circles) and reference (squares) nodes. This setup was used for all simulations given in this section.

and the initial positions of the mobile devices are given by

$$\begin{aligned} \boldsymbol{p}_1^0 &= \begin{bmatrix} 2.0 \\ 1.5 \end{bmatrix} & \boldsymbol{p}_2^0 &= \begin{bmatrix} 4.0 \\ 1.0 \end{bmatrix} & \boldsymbol{p}_3^0 &= \begin{bmatrix} 5.5 \\ 1.8 \end{bmatrix} & \boldsymbol{p}_4^0 &= \begin{bmatrix} 3.0 \\ 3.8 \end{bmatrix} \\ \boldsymbol{p}_5^0 &= \begin{bmatrix} 1.5 \\ 2.8 \end{bmatrix} & \boldsymbol{p}_6^0 &= \begin{bmatrix} 3.5 \\ 2.0 \end{bmatrix} & \boldsymbol{p}_7^0 &= \begin{bmatrix} 3.5 \\ 3.2 \end{bmatrix} & \boldsymbol{p}_8^0 &= \begin{bmatrix} 6.0 \\ 3.7 \end{bmatrix} . \end{aligned} \tag{4.2}$$

The communication topology is modeled using graph theory as introduced in Section 2.1. Since all properties of the graphs $\mathcal{G}^\eta$ and $\mathcal{G}^\mu$ are extractable from the adjacency matrix $\mathcal{A}^\eta$ and its equivalent $\tilde{\mathcal{A}}^\mu$ for $\mathcal{G}^\mu$, only those will be given for the investigated system. Nevertheless the mobile devices are moving in the 2D plane. That is why both matrices are subject to changes and are therefore only given for the initial state. Using a communication radius of $\rho = 2.5\mathrm{m}$ the initial matrices $\mathcal{A}^\eta$ and $\tilde{\mathcal{A}}^\mu$ become

$$\mathcal{A}_0^\eta = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \quad \tilde{\mathcal{A}}_0^\mu = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} . \tag{4.3}$$

4.1.2 Model Equations

In this example the positions of the moving devices $\mathcal{N}$ are estimated. Every node $i \in \mathcal{R}$ has a position $\boldsymbol{p}_i = [p_i^{\mathrm{x}} \; p_i^{\mathrm{y}}]^T$ in the 2D plane. The reference nodes $\mathcal{M}$ are static thus their position does not change over time. Conversely the mobile nodes $\mathcal{N}$ are moving in the 2D plane. The dynamics connected to the node positions can be described by

$$\begin{aligned} \boldsymbol{p}_i(k+1) &= \boldsymbol{p}_i(k) + \Delta\boldsymbol{p}_i(k) \quad \text{with } \boldsymbol{p}_i(0) = \boldsymbol{p}_i^0 && \forall i \in \mathcal{N} && (4.4a) \\ \check{\boldsymbol{p}}_i(k+1) &= \check{\boldsymbol{p}}_i(k) \quad \text{with } \check{\boldsymbol{p}}_i() = \check{\boldsymbol{p}}_i && \forall i \in \mathcal{M} && (4.4b) \end{aligned}$$

where $\Delta\boldsymbol{p}_i$ is the position shift of node i. To estimate the positions of the mobile agents, distance measurements are used. The distances $\boldsymbol{d}$ between the mobile nodes $\mathcal{N}$ can be calculated using

$$\boldsymbol{d}_{ij}^2 = \left(\boldsymbol{p}_i - \boldsymbol{p}_j\right)^T \left(\boldsymbol{p}_i - \boldsymbol{p}_j\right) \quad \text{with } i, j \in \mathcal{N}. \tag{4.5}$$

To get a linear measurement equation, (4.5) is linearized around the steady state positions $\overline{\boldsymbol{p}}_i$ and $\overline{\boldsymbol{p}}_j$ using a first order Taylor approximation, leading to

$$\boldsymbol{d}_{ji}^2 = \overline{\boldsymbol{d}}_{ji}^2 + 2\left(\overline{\boldsymbol{p}}_i - \overline{\boldsymbol{p}}_j\right)^T (\boldsymbol{p}_i - \overline{\boldsymbol{p}}_i) - 2\left(\overline{\boldsymbol{p}}_i - \overline{\boldsymbol{p}}_j\right)^T \left(\boldsymbol{p}_j - \overline{\boldsymbol{p}}_j\right) \tag{4.6}$$

with $\overline{\boldsymbol{d}}_{ji}^2 = \boldsymbol{d}_{ji}^2(\overline{\boldsymbol{p}}_i, \overline{\boldsymbol{p}}_j)$ being the distance between the steady state positions of both nodes. The distances between the reference nodes and the mobile nodes can be calculated analogously. By introducing the system variables for the mobile nodes $\mathcal{N}_i \in \mathcal{N}$

$$\begin{aligned} x_i(k) &= \boldsymbol{p}_i(k) - \overline{\boldsymbol{p}}_i, && \forall i \in \mathcal{N} \\ y_{ji}(k) &= \boldsymbol{d}_{ji}^2(k) - \overline{\boldsymbol{d}}_{ji}^2, && \forall i, j \in \mathcal{N} \\ C_{ji} &= 2\left(\overline{\boldsymbol{p}}_i - \overline{\boldsymbol{p}}_j\right)^T, && \forall i, j \in \mathcal{N} \end{aligned} \tag{4.7a}$$

and the variables for the measurements from mobile to reference nodes $\mathcal{M}_i \in \mathcal{M}$

$$\begin{aligned} \check{y}_{ji}(k) &= \boldsymbol{d}_{ji}^2(k) - \overline{\boldsymbol{d}}_{ji}^2, && \forall i \in \mathcal{N},\ j \in \mathcal{M} \\ \check{C}_{ji} &= 2\left(\overline{\boldsymbol{p}}_i - \overline{\boldsymbol{p}}_j\right)^T, && \forall i \in \mathcal{N},\ j \in \mathcal{M}. \end{aligned} \tag{4.7b}$$

the following measurement equations for one measurement y_{ji} can be derived

$$\begin{aligned} y_{ji}(k) &= C_{ji}\left(x_i(k) - x_j(k)\right) && \text{if } i, j \in \mathcal{N} \\ \check{y}_{ji}(k) &= \check{C}_{ji} x_i(k) && \text{if } i \in \mathcal{N},\ j \in \mathcal{M}. \end{aligned} \tag{4.8}$$

By assuming white noise v_i on the distance measurements, the measurement equation for one mobile node $\mathcal{N}_i$ can be given in vector form analogously to (2.5) by

$$y_i(k) = C_i \mathbf{x}(k) + v_i(k). \tag{4.9}$$

with $\mathbf{x} = [x_1^T \; \dots \; x_\eta^T]^T$. The measurement matrix y_i in (4.9) is given by the nonzero lines of

$$\begin{aligned}\boldsymbol{y_I} = &\Big[y_{1,i} \quad \dots \quad y_{i-1,i} \quad \big[\check{y}_{1,i} \quad \dots \quad \check{y}_{\mu,i}\big] \quad y_{i+1,i} \quad \dots \quad y_{\eta,i}\Big]^T \\ &\text{with} \quad y_{j,i} = \quad \text{if } (j,i) \notin \mathcal{E}^\eta \quad i,j \in \mathcal{N} \\ &\text{and} \quad \check{y}_{j,i} = \quad \text{if } (j,i) \notin \mathcal{E}^\mu \quad i \in \mathcal{N},\; j \in \mathcal{M}\end{aligned} \tag{4.10}$$

The measurement matrix C_i in (4.9) is given by the nonzero lines of $\boldsymbol{C_I}$, where $\boldsymbol{C_I}$ is defined as

$$\boldsymbol{C_I} = \begin{pmatrix} -C_{1,i} & 0 & \dots & 0 & C_{1,i} & 0 & \dots & & 0 \\ 0 & \ddots & & & \vdots & & & & \\ & & & -C_{i-1,i} & C_{i-1,i} & & & & \\ \vdots & & & \begin{matrix} 0 \\ \\ \vdots \end{matrix} & \begin{bmatrix} \check{C}_{1,i} \\ \vdots \\ \check{C}_{\mu,i} \end{bmatrix} & \begin{matrix} \vdots \\ \\ 0 \end{matrix} & & & \vdots \\ & & & & C_{i+1,i} & -C_{i+1,i} & & & \\ & & & & \vdots & & & \ddots & 0 \\ 0 & \dots & & 0 & C_{\eta,i} & 0 & \dots & 0 & -C_{\eta,i} \end{pmatrix}$$

$$\begin{aligned} &\text{with} \quad C_{j,i} = \mathbf{0} \text{ if } (j,i) \notin \mathcal{E}^\eta,\; i,j \in \mathcal{N} \\ &\text{and} \quad \check{C}_{j,i} = \mathbf{0} \text{ if } (j,i) \notin \mathcal{E}^\mu,\; i \in \mathcal{N},\; j \in \mathcal{M} \end{aligned} \tag{4.11}$$

Example: (Measurement equation for mobile node $\mathcal{N}_1$) *Using the definitions in* (4.11) *and* (4.10) *the measurement equation for node $\mathcal{N}_1$ is given by*

$$\begin{bmatrix} \check{y}_{1,1} \\ \check{y}_{2,1} \\ y_{2,1} \\ y_{5,1} \\ y_{6,1} \end{bmatrix} = \begin{pmatrix} \check{C}_{1,1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \check{C}_{2,1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ C_{2,1} & -C_{2,1} & 0 & 0 & 0 & 0 & 0 & 0 \\ C_{5,1} & 0 & 0 & 0 & -C_{5,1} & 0 & 0 & 0 \\ C_{6,1} & 0 & 0 & 0 & 0 & -C_{6,1} & 0 & 0 \end{pmatrix} \mathbf{x}(k) + \mathbf{v}(k). \tag{4.12}$$

By removing the nonzero columns of C_1 one gets the reduced measurement matrix $C_{[1]}$. Using $C_{[1]}$ leads to the following, equivalent measurement equation:

$$\begin{bmatrix} \check{y}_{1,1} \\ \check{y}_{2,1} \\ y_{2,1} \\ y_{5,1} \\ y_{6,1} \end{bmatrix} = \begin{pmatrix} \check{C}_{1,1} & 0 & 0 & 0 \\ \check{C}_{2,1} & 0 & 0 & 0 \\ C_{2,1} & -C_{2,1} & 0 & 0 \\ C_{5,1} & 0 & -C_{5,1} & 0 \\ C_{6,1} & 0 & 0 & -C_{6,1} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_5 \\ x_6 \end{pmatrix} + \begin{pmatrix} v_1 \\ v_2 \\ v_5 \\ v_6 \end{pmatrix}. \tag{4.13}$$

Static Graph

As mentioned in Section 3 there are applications where the graph of the system does not change. This is also true for the position estimation of mobile agents, as discussed in this chapter. One possibility to obtain a static graph is to use a strategy to preserve the graph while moving the agents, as given for example in [9]. In this thesis a simpler set up is investigated, where the movement of the robots is restricted to a small area around their steady state position. It is furthermore assumed that the robots move significantly slower than the sampling frequency. Therefore the position shift $\Delta\boldsymbol{p}_i$ in (4.4) is very small and can be approximated by white process noise w. Using these assumptions (4.4) and (4.9) can be used to derive the following system equations for one node

$$\begin{aligned} x_i(k) &= A_i x_i(k-1) + w_i(k) \quad \text{with} \quad x_i(0) = \mathbf{I}_{2\times1} \quad \forall i \in \mathcal{N} \\ y_i(k) &= C_{[i]} x_{[i]}(k) + v_i(k) \end{aligned} \tag{4.14}$$

where A_i is the identity matrix $\mathbf{I}_{2\times2}$. The measurement matrix $C_{[i]}$ contains the nonzero columns of C_i and $x_{[i]}$ is the corresponding state vector, both defined in (2.7). Following (2.5), the matrix form of (4.14) is given by

$$\begin{aligned} \mathbf{x}(k) &= \mathbf{A}\mathbf{x}(k-1) + \mathbf{w}(k) \quad \text{with} \quad \mathbf{x}(0) = \mathbf{0}_{16\times1} \\ \mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k) + \mathbf{v}(k). \end{aligned} \tag{4.15}$$

Dynamic Graph

If the movement of the robots is not restricted, the graph of the system becomes dynamic. Furthermore, if the mobile nodes are moving in the 2D plane, the linearization of the model around a fixed steady state is not applicable. But since it is assumed that the dynamics of the positions are significantly slower than the sampling time, e.g. the mobile devices have a fairly low velocity, the time scale τ of the robot movement and the time scale k of the position estimation can be separated. To model the global behavior of the system a switching scheme is introduced, where the steady state position $\overline{\boldsymbol{p}}$ is updated after a certain time T_τ,

$$\overline{\boldsymbol{p}}(\tau) = \frac{1}{K} \sum_{k=k_{\tau-1}}^{k_\tau-1} \left(\hat{\mathbf{x}}(\tau-1,k) + \overline{\boldsymbol{p}}(\tau-1)\right) \quad \forall k \in [k_{\tau-1}, k_\tau). \tag{4.16}$$

By introducing those additional dynamics, the linearization (4.6) has to be redone in every time step τ. Therefore the system variables (4.17) become time dependent

$$\begin{aligned} x_i(\tau,k) &= \boldsymbol{p}_i(k) - \overline{\boldsymbol{p}}_i(\tau), & k &\in [k_\tau, k_{\tau+1}), \quad \forall i \in \mathcal{N}, \\ y_{ji}(\tau,k) &= \boldsymbol{d}_{ji}^2(k) - \overline{\boldsymbol{d}}_{ji}^2(\tau), & k &\in [k_\tau, k_{\tau+1}), \quad \forall i,j \in \mathcal{N}, \\ C_{ji}(\tau) &= 2\left(\overline{\boldsymbol{p}}_i(\tau) - \overline{\boldsymbol{p}}_j(\tau)\right)^T, & k &\in [k_\tau, k_{\tau+1}), \quad \forall i,j \in \mathcal{N}. \end{aligned} \tag{4.17a}$$

The variables including reference nodes become

$$\begin{aligned}
\check{y}_{ji}(\tau,k) &= \boldsymbol{d}_{ji}^2(k) - \overline{\boldsymbol{d}}_{ji}^2(\tau), && k \in [k_\tau, k_{\tau+1}), \\
& && \forall i \in \mathcal{N},\ j \in \mathcal{M}, \\
\check{C}_{ji}(\tau) &= 2\left(\overline{\boldsymbol{p}}_i(\tau) - \overline{\boldsymbol{p}}_j(\tau)\right)^T, && k \in [k_\tau, k_{\tau+1}), \\
& && \forall i \in \mathcal{N},\ j \in \mathcal{M}.
\end{aligned} \tag{4.17b}$$

Using these new variables and the general system dynamics given in (4.4) and (4.9) a switching model for the system of moving robots can be given:

$$\begin{aligned}
\mathbf{x}(\tau,k) &= \mathbf{A}\mathbf{x}(\tau,k-1) + \mathbf{w}(k), && k \in [k_\tau, k_{\tau+1}) \\
\mathbf{y}(\tau,k) &= \mathbf{C}(\tau)\mathbf{x}(\tau,k) + \mathbf{v}(k), && k \in [k_\tau, k_{\tau+1})
\end{aligned} \tag{4.18}$$

$$\text{with} \quad \begin{aligned}
\mathbf{x}(\tau,k_\tau-1) &= \mathbf{x}(\tau-1,k_\tau-1) + \Delta\overline{\boldsymbol{p}}(\tau), \\
\Delta\overline{\boldsymbol{p}}(0) &= \overline{\boldsymbol{p}}(\tau-1) - \overline{\boldsymbol{p}}(\tau) = \mathbf{0}_{16\times1}, \\
\mathbf{x}(0) &= \mathbf{0}_{16\times1},
\end{aligned}$$

where $\Delta\overline{\boldsymbol{p}}(\tau)$ is the position shift at time τ. The local system equations for every single robot can be derived analogously.

4.2 Parameter Selection

While selecting the parameters the main focus was to make reasonable assumptions which would lead to meaningful simulation results. The parameter choices will be argued in this section and are summarized in Table 4.1. It was already mentioned in Section 4.1.1 that the investigated system consists of $\eta = 8$ mobile nodes and $\mu = 6$ reference nodes. Their positions are given in (4.2) and (4.1), respectively. The communication topology of the system is given in (4.3).

The fusion algorithm introduced in Section 3.3.1 was implemented in real robots and experimental results where published in [1]. Since the experiments in [1] use ultrasound for the distance measurements, this setup was also assumed in this thesis. Following the parameters used in [21], the communication radius was set to $\rho = 2.5$m and the measurement noise on those distance measurements was set to $\sigma_d = 0.002$m^2. It is furthermore assumed that the robots move with an average velocity of $v = 0.1$m/s and that a collision avoidance strategy is implemented to preserve a minimum distance $d_{min} = 0.2$m among robots.

Using those specifications random trajectories of the robots where generated and mutual measurements where simulated. The sampling rate was set to $f_k = 100$Hz to compromise between measurement accuracy and computational costs. The switching frequency for the model was set to $f_\tau = 10$Hz. To preserve sensitivity of the online gain calculation on changing conditions, the number of measurements used for this purpose was restricted to $n_{save} = 300$.

It is crucial for the Kalman Filter algorithm to perform well, to make reasonable assumptions

Table 4.1: Parameters used for the simulation of both scenarios.

Parameter	Value	Description
Network properties		
μ	6	number of reference nodes
η	8	number of mobile nodes
ρ	2.5 m^2	communication radius
Robot behavior		
d_{min}	0.2 m	minimum distance between nodes
σ_d	0.002 m^2	noise for distance measurements
v	$0.1\ \frac{\text{m}}{\text{s}}$	average velocity of the robots
Simulation parameters		
f_k	100 Hz	sampling rate
f_τ	10 Hz	switching frequency
n_{save}	300	Number of stored measurements for online calculation
Kalman Filter parameters		
$\mathbf{Q}$	diag(0.001 m^2)	covariance matrix of process noise
$\mathbf{R}$	diag(0.05 m^2)	covariance matrix of meas. noise
$\hat{\mathbf{x}}(0)$	$\mathbf{0}_{16\times1}$	initial conditions for state vector
$\mathbf{P}(0)$	diag(0.001m^2)	initial conditions for covariance matrix

about the initial conditions. In the simulations, the initial positions of the mobile nodes (see (4.2)) where used as initial steady state positions $\overline{\boldsymbol{p}}(\tau = 0, k = 0)$. Therefore the initial conditions for the state estimate are zero. Giving a sampling rate of $f_k = 100$Hz and a velocity of $v = 0.1$m/s the covariance of the process noise becomes $Q_i =$ diag$(0.001$m$^2)$. Analogously the initial covariance of the estimation error $\mathbf{P}$ is set to $\mathbf{P}(0) =$ diag$(0.001$m$^2)$. Since the model equations are linearized around a steady state position, the output variable $\mathbf{y}$ does not relate directly to the distance measurement. By simulating measurements $\mathbf{y}_{meas}$ and $\mathbf{y}_{real}$ with and without measurement noise, respectively, the covariance matrix $\mathbf{R}$ was calculated. By using the approximation

$$\mathbf{R} = E\left[\mathbf{v}\mathbf{v}^T\right] \approx \frac{1}{k_{end}} \sum_{k=1}^{k_{end}} \left(\mathbf{v}(k)\mathbf{v}(k)^T\right)$$
$$\text{with} \quad \mathbf{v}(k) = \mathbf{y}_{meas}(k) - \mathbf{y}_{real}(k)$$

a covariance matrix of $\mathbf{R} =$ diag$(0.05$ m$^2)$ was obtained. Following the usual convention, a diagonal matrix was used. To simplify the algorithm it was assumed, that $\mathbf{R}$ is not a function of the distance nor the time.

4.3 Simulation Results

In this section the performance of the introduced Kalman Filter algorithms is investigated using MATLAB simulations of two different scenarios. Firstly the mobile nodes are only allowed to move in a small area around their steady state position. This leads to a system with a static graph. In this set up, the algorithms using offline gain calculation can be used and compared. Secondly, the restrictions on the movement of the robots where dropped, allowing the agents to move in the 2D plane. This leads to a dynamic graph. Therefore online gain calculation is used for the second scenario. To refer to the different algorithms in an easy fashion, abbreviations are introduced in Table 4.2.

Table 4.2: Abbreviations for the simulated Kalman Filter Algorithms. The equation specifying each algorithm is given in the last column.

Abbrev.	Algorithm name	Defining equation
gKF	global Kalman Filter	p. 11, (3.1)
dKF1	distributed Kalman Filter using global measurements	offline: p. 14, (3.2) online: p. 15, (3.2)
dKF2	distributed Kalman Filter with data fusion	offline: p. 23, (3.3.2)
dKF3	extended distributed Kalman Filter with data fusion for systems with dynamic graph	offline: p. 28, (3.46) online: p. 29, (3.3.3)

4.3.1 Static Graph

To generate a scenario where a static graph is preserved throughout the simulation, the movement of the robots was restricted to an area of radius $r = 0.4$m around their steady state positions (see (4.2)). The randomly generated trajectories for the mobile nodes, using the robot parameters specified in Table 4.1, are shown in Fig. 4.2. Using those trajectories simulations where done with all three distributed algorithms, given in Table 4.2. The results for the global Kalman Filter (gKF) are also given for comparison. At first the algorithms where tested using offline gain calculation. Afterwards also online gain calculation was used for the static graph to see the impact of the approximations made on the performance of dKF1 and dKF3. Finally also the switched model (4.18) was used to verify its applicability to the given problem. The simulation results are shown in Fig. 4.3. In this graphic the average RMS error $\overline{\varepsilon}$

$$\begin{aligned} &\overline{\varepsilon} = \frac{1}{k_{end}} \sum_{k=1}^{k_{end}} \sqrt{(\boldsymbol{p}_k - \hat{\boldsymbol{p}}(k))(\boldsymbol{p}(k) - \hat{\boldsymbol{p}}(k))^T} \\ &\text{with} \quad \varepsilon = \sqrt{(\boldsymbol{p}_k - \hat{\boldsymbol{p}}(k))(\boldsymbol{p}(k) - \hat{\boldsymbol{p}}(k))^T} \end{aligned} \tag{4.19}$$

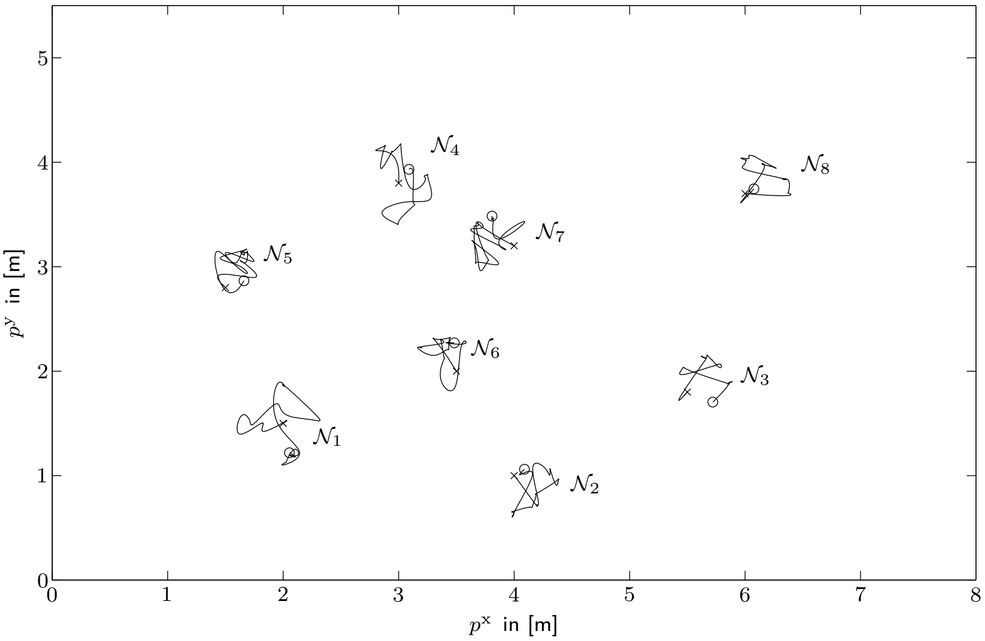


Fig. 4.2: Randomly generated trajectories of mobile robots $\mathcal{N}$ in the 2D-plane. The movement was restricted to an area of radius $r = 0.4$m around the steady state positions of the agents. This leads to a static graph and linear dynamics of the system.

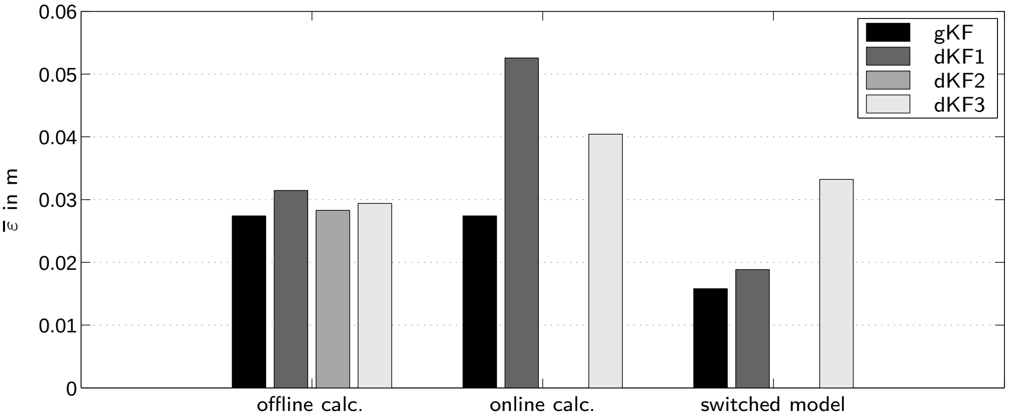


Fig. 4.3: Comparison of the average RMS error $\overline{\varepsilon}$ of the position estimation for the static graph scenario, visualized in Fig. 4.2. Simulations where done using online gain calculation (online calc.) and offline gain calculation (offline calc.). In the last column the performance of the algorithms is shown when the gains are calculated online and the switched model (4.18) is used (switched model).

of the position estimation error is plotted for the different algorithms using the respective scenario as specified underneath the bars. The position estimate in (4.19) is calculated using (4.7a) with $\hat{\boldsymbol{p}}(k) = \hat{\mathbf{x}}(k) + \overline{\boldsymbol{p}}(k)$.

When comparing only the offline and online gain calculation scenarios in Figure 4.3, it can be seen that the most precise position estimates were obtained when offline gain calculation was used. This result was expected since in this case more available information can be used and less approximations have to be utilized to calculate the Kalman matrix. When looking at the results of the first scenario it can also be seen, that the performance of `dKF2` gets very close to the performance of the `gKF`. This proves again, that the data fusion concept does improve the accuracy of the estimation. The extended data fusion algorithm `dKF3`, which can also be calculated online, performs similar than the `dKF2` algorithm. Nevertheless it is also obvious that the improvement gained, in comparison to the simpler algorithm `dKF1` is not significant.

When comparing the performance of the single algorithms in different scenarios a different behavior of `dKF1` and `dKF3` can be seen. The error of `dKF1` nearly doubles when the gain is calculated online and can be reduced even far below the error of the offline scenario, when the switched model was used. This is due to the fact, that using a more realistic model by redoing the linearization, the predicted state estimates gets more accurate. Since the state prediction is used to calculate the covariance matrix online (see (3.15)) the impact of these adjustments is quite large. A comparable behavior regarding the first and the last scenario can be seen in the results obtained by using the `gKF`.

By looking at the estimation error of algorithm `dKF3` a different behavior can be seen. The error is also increased when using online gain calculation due to the necessary approximations. But due to more available measurements, the impact of those approximations is lower than in the `dKF1` case. Furthermore, it can be seen, that introducing the switched model does not improve the performance of `dKF3` as much as seen with the `gKF` and `dKF1` algorithm.

4.3.2 Dynamic Graph

In this second scenario the movement of the robots is not restricted to a certain area. It is assumed that the robots have an implemented collision avoidance strategy to have a minimum distance of $d_{min} = 0.2$m between each other at all times. Fig. 4.4 shows the randomly generated trajectories using those assumptions and the simulation parameters specified in Table 4.1. Due to the movement of the robots the graph changes during the simulations. New measurements are available if one robot gets closer to another one, and others are lost due to an increased distance. This introduces switches in the behavior of the state estimation, since the number of available measurements changes over time. Those switches cause difficulties in the calculation of the covariance matrices of the prediction error.

The covariance matrices are calculated from measurements using (3.15) and (3.48) for algorithm `dKF1` and `dKF3`, respectively. This approximation uses stored measurements from previous timesteps as well as current ones. Since the graph is dynamic, the number of stored values for every available distance measurement differes. If a new node is entering the communication horizon, no stored values are available for this measurement. This can lead to undesired effects since the covariance calculation needs some time to converge (compare

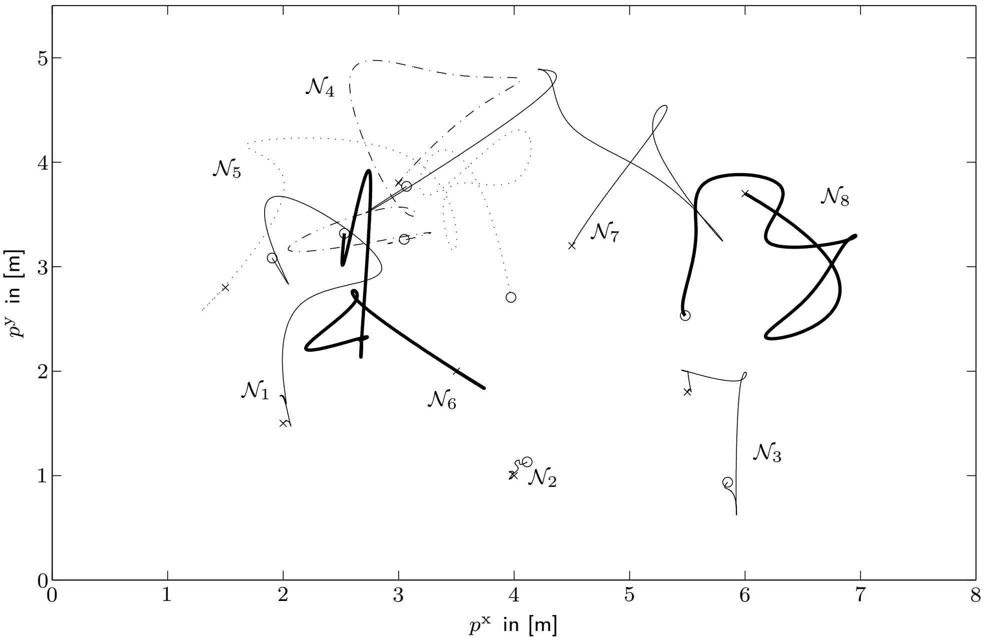


Fig. 4.4: Randomly generated trajectories of mobile robots $\mathcal{N}$ in the 2D-plane. By loosening the restrictions on the movement of the mobile devices, the graph of the system becomes dynamic, and the switched model (4.18) must be used. The movement was restricted to always ensure a minimum distance of $d_{min} = 0.2$m between nodes. The robots move with an average velocity of $v = 0.1\frac{\text{m}}{\text{s}}$.

Fig. B.2 in Appendix B). Therefore a hysteresis was introduced, where a certain number of measurements is gathered before the measurement is used for gain calculation. The size of the hysteresis was set to $n_{wait} = 100$ and $n_{wait} = 300$ for the dKF1 and dKF3 algorithm, respectively. This method ensures that the covariance converges before it is actually used for gain calculation. It additionally reduces the number of graph changes by not using measurements to neighbors which are in communication distance only for a very short time.

To compare the investigated algorithms, the RMS error of the position estimates

$$\begin{aligned}\overline{\varepsilon} &= \frac{1}{k_{end}} \sum_{\tau=0}^{\tau_{end}} \sum_{k=k_\tau}^{k_{\tau+1}-1} \sqrt{(\boldsymbol{p}(\tau,k) - \hat{\boldsymbol{p}}(\tau,k))(\boldsymbol{p}(\tau,k) - \hat{\boldsymbol{p}}(\tau,k))^T} \\ \text{with} \quad \varepsilon &= \sqrt{(\boldsymbol{p}(\tau,k) - \hat{\boldsymbol{p}}(\tau,k))(\boldsymbol{p}(\tau,k) - \hat{\boldsymbol{p}}(\tau,k))^T}\end{aligned} \tag{4.20}$$

was calculated, similar to (4.19). The resulting errors for the different algorithms are shown in Fig. 4.5. Since the performance of the algorithms was not satisfying for a communication horizon of $\rho = 2.5$m it was investigated if an increased communication horizon of $\rho = 3.5$m decreases the error. This expectation proved to be true, leading to a significantly reduced error for both algorithms when a larger horizon is used.

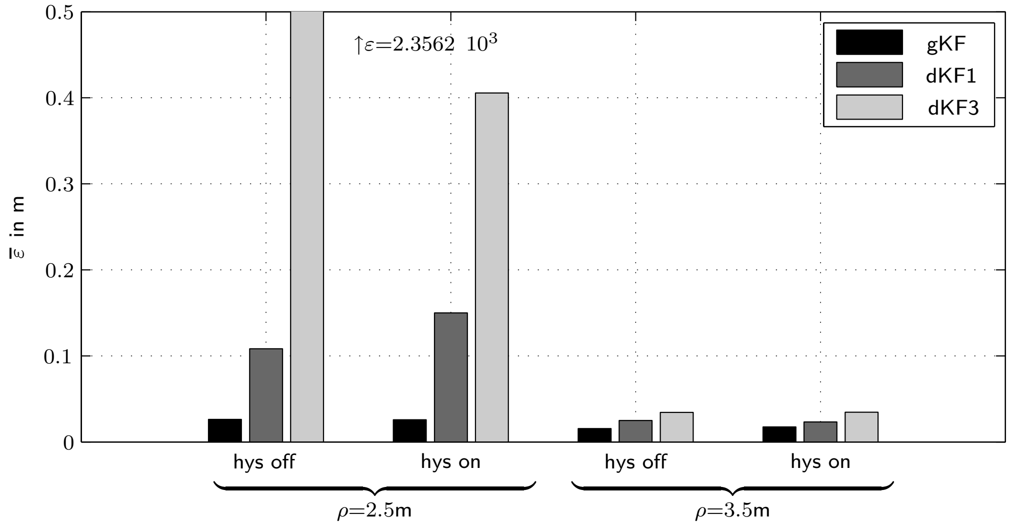


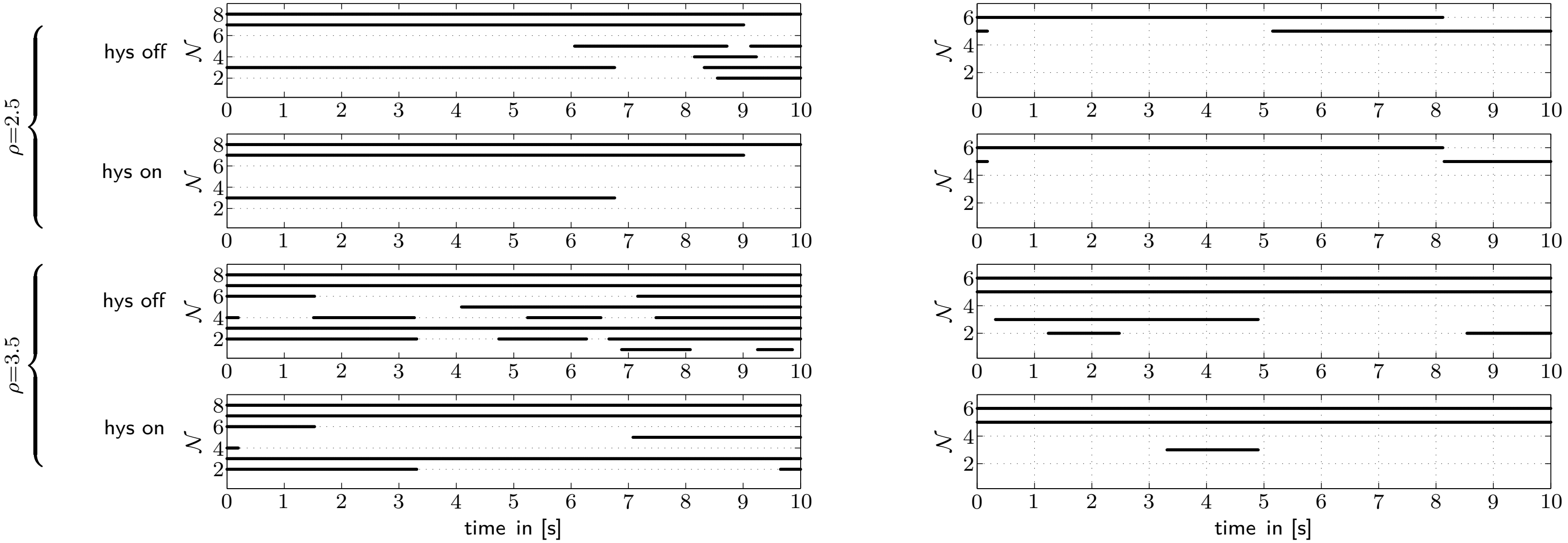
Fig. 4.5: Comparison of the average RMS error $\bar{\varepsilon}$ for the dynamic graph scenario, visualized in Fig. 4.4. For both communication radiuses $\rho = 2.5$m and $\rho = 3.5$m simulations were done with the hysteresis for the graph changes turned on (hys on) and off (hys off). The size of the hysteresis was set to $n_{wait} = 100$ and $n_{wait} = 300$ for the dKF1 and dKF3 algorithm, respectively.

The devolution of the error for one single node is shown in Fig. 4.7 where the error is plotted over time for node $\mathcal{N}_8$. The corresponding graph switches for Fig. 4.5 are visualized in Fig. 4.6. By comparing the graph switches for the different scenarios in Fig. 4.6, one can see, that by introducing a hysteresis on the used measurements, the number of graph changes could be reduced. But on the other hand this also reduces the average number of neighbors and reachable nodes significantly, as it can be seen in Table 4.3. This can also have negative effects on the performance of the filter.

Looking at Fig. 4.5 again, it can be seen that the introduction of the hysteresis has quite large effects when only a small communication horizon is used. Therefore it can be concluded that graph switches are more crucial if the system is not well coupled. In this case situations

Table 4.3: Comparison of the average coupling $\bar{\zeta}$ between mobile nodes and the average number $\bar{\gamma}$ of reachable reference nodes for different simulated scenarios. For completeness also the average coupling in the static graph is given.

graph	scenario	$\bar{\zeta}$	$\bar{\gamma}$
static	$\rho = 2.5$	3.8	1.3
dynamic	$\rho = 2.5$, hys off	3.3	1.1
	$\rho = 2.5$, hys on	2.8	1.0
	$\rho = 3.5$, hys off	5.4	2.5
	$\rho = 3.5$, hys on	4.8	2.3



Neighboring nodes $\mathcal{N}_{[8]}$ of node $\mathcal{N}_8$

Reachable reference nodes $\mathcal{M}_{[8]}$ of node $\mathcal{N}_8$

Fig. 4.6: Visualization of the graph changes in $\mathcal{G}^\eta$ and $\mathcal{G}^\mu$ concerning $\mathcal{N}_8$ for all four simulated online scenarios. Node $\mathcal{N}_8$ can communicate to note $\mathcal{N}_i$ at time k if there is a black dot at (k, i). Results are similar for other nodes.

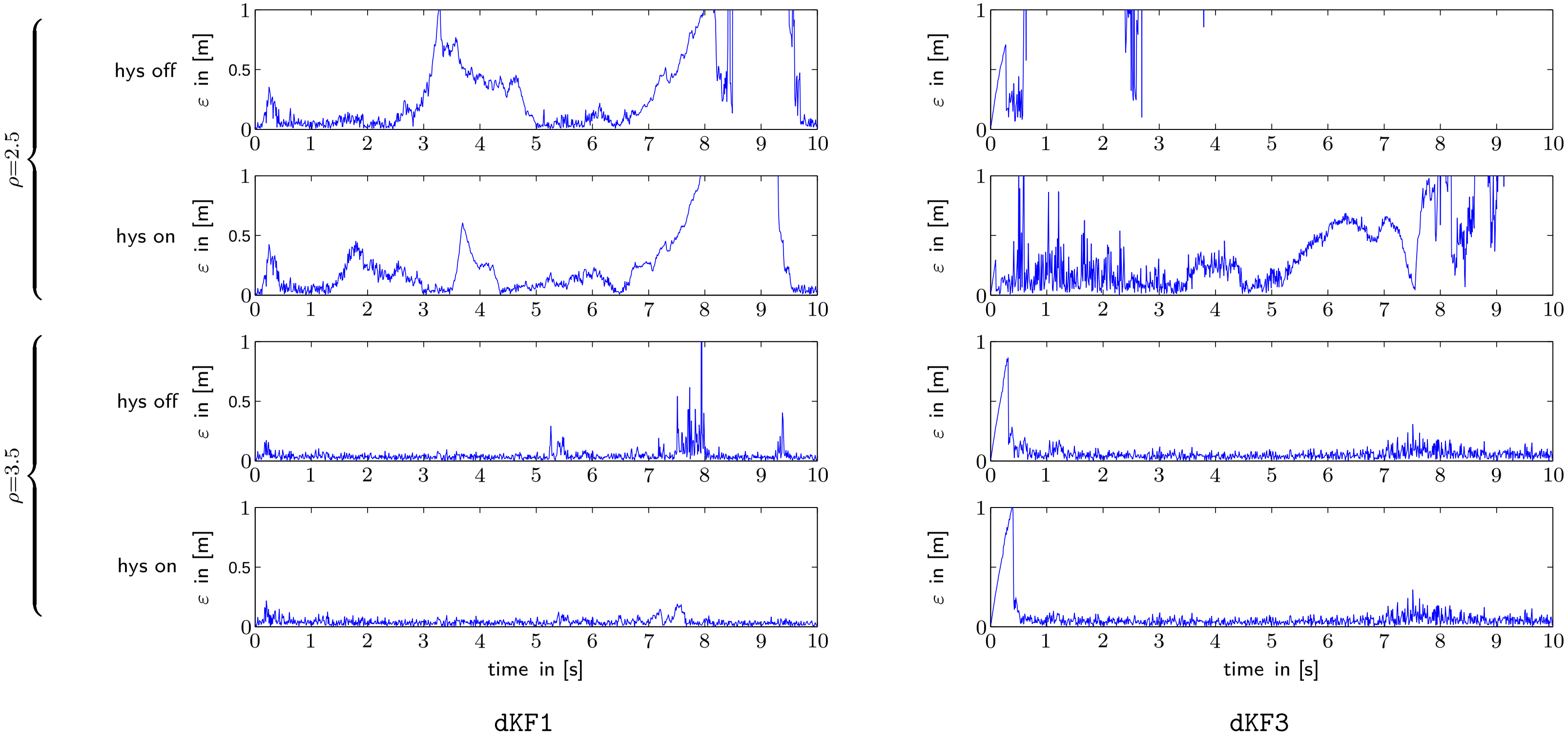


Fig. 4.7: Comparison of the RMS error ε of node $\mathcal{N}_8$ over time. All four scenarios visualized in Fig. 4.5 are given from top to bottom. From left to right, the performance of dKF1 and dKF3 can be compared. To make changes more obvious the y-axis was scaled. For the unscaled version please refer to Fig. B.1 in Appendix B.

can occur where the coupling for some nodes gets too low to obtain good estimation results. Furthermore, by using a switched model, this can result in a fast blowup of the error. Fortunately it was observed that the error reduces again when new measurements become available. This behavior can be seen in the unscaled version of Fig. 4.7 which can be found in Appendix B, Fig. B.1. It was furthermore observed that the dKF1 algorithm is more robust to graph changes than the dKF3 algorithm. This might be due to the fact that the online gain calculation for dKF3 depends fully on available measurement whereas in dKF1 only the covariance of the state prediction is approximated.

By comparing the estimation errors obtained by using $\rho = 3.5$m with the errors shown for the third scenario in Fig. 4.3 one can see that they are in the same range. Therefore it seems necessary to have a higher coupling when graph changes occur in the network to ensure reliable performance. The second interesting observation is that the overall performance of the simple algorithm dKF1 is better then the performance of the dKF3 algorithm when using a dynamic graph. But on the other hand it can be seen in Fig. 4.3 that the estimation error for dKF1 is also decreased below the error for dKF3 when only the switched model is used even if no graph changes occur. Therefore it is assumed, that the accuracy of the state prediction gained by using the switched model is more advantageous for the simple Kalman Filter, than the use of additional measurements is for the data fusion filter. It should be object of further work to investigate the performance of dKF1 and dKF3 when a fully linear model is used.

Chapter 5

Conclusion

In this thesis different distributed Kalman Filter algorithms where derived for both static and dynamic network systems. The objective for all algorithms was to estimate the position of mobile nodes in a distributed fashion. Distance measurements and communication were restricted to only take place among neighbors.

Firstly, a distributed filter was designed, which is based on a normal global Kalman Filter algorithm. It was obtained by restricting the optimization problem for the Kalman gain what leads to a distributed filter that only uses local measurements. Afterwards, the concept of a data fusion Kalman Filter was applied to the network system. A straight forward derivation of this filter could only be found for systems with a static graph structure. Therefore an additional communication step was introduced to derive a distributed data fusion Kalman Filter that is usable for both static and dynamic network systems.

For all investigated algorithms, the analytical calculation of the gain matrix is strongly coupled among different nodes. If the graph of the investigated system is static and the model is time independent, the gain calculation can be done offline. But usually this is not the case. In this work it was therefore proposed to approximate covariance matrices needed for the gain calculation, using recent measurements. This leads to a fully decentralized online algorithm for both distributed filters.

To investigate the performance of the introduced algorithms, simulations where done in MATLAB. Since the position estimation is based on distance measurements in the considered systems, the measurement equation becomes nonlinear. To obtain linear dynamics, the model was linearized around a steady state position.

By simulating the linearized system using a setup where the graph is static, it was found, that introducing a data fusion step increases the accuracy of the state estimation. This is true for both online and offline gain calculation. As expected, it was also shown that approximating the covariance matrices leads to a small performance reduction but is still applicable.

When free movement of the mobile devices was considered in the investigated example, the linearizion of the model around one steady state position was not applicable anymore. In this case it was assumed that the devices move with a velocity significantly smaller than the sampling rate. Therefore a switched system could be introduced where the dynamics of the position estimation and the robot movement were separated.

Using this model, simulations were done for a scenario involving graph changes. It was found, that a higher average coupling among the nodes is necessary if the graph changes, to ensure reliable performance. Furthermore, it was seen that the simple distributed filter performs better in this scenario than the one using a data fusion step. This pattern was also found when combining a static graph scenario with the switched model. Therefore it is assumed, that the accuracy of the state prediction, gained by using the switched model is more advantageous for the simple Kalman Filter, than the use of additional measurements is for the data fusion filter.

It was shown in this work, that by using distance measurements and communication among mobile devices for localization purposes, distributed position estimation could be improved and performed online. Therefore, this method has a high potential to be used in future applications involving challenging distributed localization problems, such as the control of autonomous vehicles or trucks moving in a convoy.

Appendix A

Details on the Kalman Gain Calculation

In this section the solutions to the optimization problems for the calculation of the Kalman gains used in Section 3 are derived in detail. First some useful matrix derivation rules will be given and then the Kalman gains for the different algorithms will be derived. The dependency of the matrices on the discrete time k will be dropped in this chapter to simplify notation.

A.1 Some Matrix Derivation Rules

The matrix derivation rules given below are taken from [5]. The use of $\text{vec}[.]$ indicates the vectorization of a matrix and $\otimes$ is the Kronecker Product.

$$\frac{\text{d tr}(AX^T)}{\text{d}X} = (\text{vec}[A])^T \tag{A.1a}$$

$$\frac{\text{d tr}(BX^TA)}{\text{d}X} = (\text{vec}[AB])^T \tag{A.1b}$$

$$\frac{\text{d tr}(XAX^T)}{\text{d}X} = \Big(\text{vec}[X(A + A^T)]\Big)^T \tag{A.1c}$$

$$\frac{\text{d tr}(AXBX^TC)}{\text{d}X} = \Big(\text{vec}[A^TC^TXB^T + CAXB]\Big)^T \tag{A.1d}$$

$$\frac{\text{d vec}[A^TXB]}{\text{d vec}[X]} = (B \otimes A)^T \tag{A.1e}$$

$$\text{vec}[ABC] = (C^T \otimes A)\text{vec}[B] \tag{A.1f}$$

A.2 Global Kalman Filter

This section derives the Kalman Matrix for the global Kalman Filter algorithm discussed in Section 3.1. To simplify notation the covariance matrix $\mathbf{P}(k|k)$ is rewritten to

$$\begin{aligned} &\mathbf{P}(k|k) = \begin{bmatrix}\mathbf{I} & \mathbf{K}\end{bmatrix} F \begin{bmatrix}\mathbf{I} & \mathbf{K}\end{bmatrix}^T \\ \text{with} \quad &F = \begin{bmatrix}\mathbf{I} \\ -\mathbf{C}\end{bmatrix} \mathbf{P}(k|k-1) \begin{bmatrix}\mathbf{I} \\ -\mathbf{C}\end{bmatrix}^T + \begin{bmatrix}\mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}\end{bmatrix}\end{aligned} \tag{A.2}$$

to separate the part F of $\mathbf{P}(k|k)$ which is not a function of $\mathbf{K}$. Using a partition for F

$$F = \begin{bmatrix}F_{11} & F_{12} \\ F_{21} & F_{22}\end{bmatrix} = \begin{bmatrix}\mathbf{P}(k|k-1) & -\mathbf{P}(k|k-1)\mathbf{C}^T \\ -\mathbf{C}\mathbf{P}(k|k-1) & \mathbf{C}\mathbf{P}(k|k-1)\mathbf{C}^T + \mathbf{R}\end{bmatrix} \tag{A.3}$$

the covariance matrix (A.2) can be rewritten as

$$\mathbf{P}(k|k) = F_{11} + \mathbf{K}F_{12} + F_{21}\mathbf{K}^T + \mathbf{K}F_{22}\mathbf{K}^T. \tag{A.4}$$

Using (A.4) the matrix equality $\mathrm{tr}(A) = \mathrm{tr}(A^T)$ and the symmetry of $\mathbf{P} = \mathbf{P}^T$ the cost function $J = \mathrm{tr}(\mathbf{P}(k|k))$ in (3.5) becomes

$$J = \mathrm{tr}(\mathbf{P}(k|k)) = \mathrm{tr}(F_{11}) + 2\mathrm{tr}(F_{12}\mathbf{K}^T) + \mathrm{tr}(\mathbf{K}F_{22}\mathbf{K}^T).$$

With the matrix derivation rules (A.1a) and (A.1c) and the symmetry of $\mathbf{P} = \mathbf{P}^T$ its derivative is given by

$$\frac{\mathrm{d}J}{\mathrm{d}\mathbf{K}} = (\mathrm{vec}[F_{12} + \mathbf{K}F_{22}])^T. \tag{A.5}$$

Since (A.5) has to be zero to obtain a minimum, the optimal Kalman gain $\mathbf{K}$ is given by

$$\begin{aligned}\mathbf{K} &= -F_{12}(F_{22})^{-1}. \\ &= \mathbf{P}(k|k-1)\mathbf{C}^\mathrm{T}\left[\mathbf{C}\mathbf{P}(k|k-1)\mathbf{C}^\mathrm{T} + \mathbf{R}\right]^{-1}\end{aligned} \tag{A.6}$$

A.3 Distributed Kalman filter using Local Measurements

This section derives the Kalman Matrix for the distributed Kalman Filter algorithm discussed in Section 3.2. To obtain an equation for $\bar{K}_i$ the minimum point of the cost function J with respect to $\bar{K}_i$ has to be calculated using

$$0 = \frac{\mathrm{d}J}{\mathrm{d}\bar{K}_i} = \frac{\mathrm{d}J}{\mathrm{d}\bar{\mathbf{K}}}\frac{\mathrm{d}\bar{\mathbf{K}}}{\mathrm{d}\bar{K}_i}. \tag{A.7}$$

The outer derivative in (A.7) is given in (A.5) where $\bar{\mathbf{K}}$ is vectorized. Using the matrix derivation rule (A.1e) the inner derivative of (A.7) becomes

$$\frac{\mathrm{d}\ \mathrm{vec}[\bar{\mathbf{K}}]}{\mathrm{d}\ \mathrm{vec}[\bar{K}_i]} = \left(V_i \otimes \bar{U}_i\right)^T. \tag{A.8}$$

Using (A.5) and (A.8) in (A.7) and applying (A.1f) the condition for optimality becomes

$$0 = \bar{U}_i \left(F_{12} + \bar{\mathbf{K}} F_{22}\right) V_i^T. \tag{A.9}$$

Separating both parts and using (3.10) leads to

$$\begin{aligned} 0 &= \bar{U}_i \left(F_{12}\right) V_i^T + \sum_{j=1}^{\eta} \left(\underbrace{\bar{U}_i\ \bar{U}_j^T}_{=0 \text{ if } i \neq j} \bar{K}_j V_j F_{22} V_i^T \right) \\ &= \bar{U}_i \left(F_{12}\right) V_i^T + \bar{K}_i V_i F_{22} V_i^T. \end{aligned} \tag{A.10}$$

Since (A.10) has to be zero to obtain a minimum, the optimal Kalman gain $\bar{K}_i$ is given by

$$\bar{K}_i = -\Upsilon_i (\bar{\Psi}_{ij})^{-1} \tag{A.11}$$

with

$$\Upsilon_i = \bar{U}_i F_{12} V_i^T = \bar{U}_i \mathbf{P}(k|k-1)(C_i)^T \tag{A.12}$$

$$\bar{\Psi}_{ij} = V_i F_{22} V_i^T = C_i \mathbf{P}(k|k-1) C_i^T + R_i. \tag{A.13}$$

A.4 Distributed Kalman filter with Data Fusion

This section derives the Kalman matrix and the weighting matrix for the distributed Kalman Filter algorithm discussed in Section 3.3.2. Both derivations are partially based on ideas given in [2].

Kalman Matrix

Since the covariance matrix given in (3.30) differs from (3.4) the decomposition of $\bar{\mathbf{P}}(k|k)$ changes to

$$\bar{\mathbf{P}}(k|k) = \mathbf{W} \begin{bmatrix} \mathbf{I} & \bar{\mathbf{K}} \end{bmatrix} F \begin{bmatrix} \mathbf{I} & \bar{\mathbf{K}} \end{bmatrix}^T \mathbf{W}^T \tag{A.14}$$

$$\text{with} \quad F = \begin{bmatrix} \mathbf{I} \\ -\bar{\mathbf{C}} \end{bmatrix} \bar{\mathbf{P}}(k|k-1) \begin{bmatrix} \mathbf{I} \\ -\bar{\mathbf{C}} \end{bmatrix}^T + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{R} \end{bmatrix}.$$

Using again a partition for F

$$\begin{aligned} F &= \begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix} \\ &= \begin{bmatrix} \bar{\mathbf{P}}(k|k-1) & -\bar{\mathbf{P}}(k|k-1)\bar{\mathbf{C}}^T \\ -\bar{\mathbf{C}}\bar{\mathbf{P}}(k|k-1) & \bar{\mathbf{C}}\bar{\mathbf{P}}(k|k-1)\bar{\mathbf{C}}^T + \mathbf{R} \end{bmatrix} \end{aligned} \tag{A.15}$$

the covariance matrix (3.30) can be rewritten as

$$\mathbf{P}(k|k) = \mathbf{W}\left(F_{11} + \bar{\mathbf{K}}F_{12} + F_{21}\bar{\mathbf{K}}^T + \bar{\mathbf{K}}F_{22}\bar{\mathbf{K}}^T\right)\mathbf{W}^T. \tag{A.16}$$

Using (A.16), the matrix equality $\mathrm{tr}(A) = \mathrm{tr}(A^T)$ and the symmetry of $\mathbf{P} = \mathbf{P}^T$ the cost function $J = \mathrm{tr}(\mathbf{P}(k|k))$ becomes

$$\begin{aligned} J = \mathrm{tr}(\mathbf{P}(k|k)) &= \mathrm{tr}(\mathbf{W}F_{11}\mathbf{W}^T) + 2\mathrm{tr}(\mathbf{W}F_{12}\bar{\mathbf{K}}^T\mathbf{W}^T) \\ &\quad + \mathrm{tr}(\mathbf{W}\bar{\mathbf{K}}F_{22}\bar{\mathbf{K}}^T\mathbf{W}^T). \end{aligned} \tag{A.17}$$

To obtain an equation for $\bar{K}_i$ the minimum point of the cost function J with respect to $\bar{K}_i$ has to be calculated using

$$0 = \frac{\mathrm{d}J}{\mathrm{d}\bar{K}_i} = \frac{\mathrm{d}J}{\mathrm{d}\bar{\mathbf{K}}}\frac{\mathrm{d}\bar{\mathbf{K}}}{\mathrm{d}\bar{K}_i}. \tag{A.18}$$

With the matrix derivation rules (A.1b) and (A.1d) the outer derivative of (A.18) is given by

$$\frac{\mathrm{d}J}{\mathrm{d}\bar{\mathbf{K}}} = \left(\mathrm{vec}[\mathbf{W}^T\mathbf{W}(F_{12} + \bar{\mathbf{K}}F_{22})]\right)^T. \tag{A.19}$$

while the inner derivative is equivalent to (A.8). This leads to the following condition for optimality:

$$0 = \mathrm{vec}[\bar{U}_i\mathbf{W}^T\mathbf{W}\left(F_{12} + \bar{\mathbf{K}}F_{22}\right)\bar{V}_i{}^T]. \tag{A.20}$$

By separating both parts of (A.20) and using (3.31) one gets

$$\begin{aligned} 0 &= \mathrm{vec}[\bar{U}_i\mathbf{W}^T\mathbf{W}\left(F_{12}\right)\bar{V}_i{}^T] \\ &\quad + \sum_{j=1}^{\eta}\left(\mathrm{vec}[\bar{U}_i\mathbf{W}^T\mathbf{W}\bar{U}_j{}^T\bar{K}_j\bar{V}_jF_{22}\bar{V}_i{}^T]\right) \end{aligned} \tag{A.21}$$

By introducing

$$\begin{aligned}\bar{\Psi}_{ij} &= \mathrm{vec}[\bar{U}_i \mathbf{W}^T \mathbf{W} F_{12} \bar{V}_i{}^T] \\ &= -\mathrm{vec}[\bar{U}_i \mathbf{W}^T \mathbf{W} \bar{\mathbf{P}}(k|k-1) \bar{\mathbf{C}} i^T \bar{V}_i{}^T] \\ \bar{\Xi}_{ij} &= \bar{U}_i \mathbf{W}^T \mathbf{W} \bar{U}_j{}^T \\ \bar{\Upsilon}_{ij} &= \bar{V}_j F_{22} \bar{V}_i^T = \bar{V}_j \bar{\mathbf{C}} \bar{\mathbf{P}}(k|k-1) \bar{\mathbf{C}}^T \bar{V}_i^T + \bar{V}_j \mathbf{R} \bar{V}_i^T\end{aligned}$$

and using the matrix rule for vectorization, (A.21) becomes

$$-\bar{\Psi}_{ij} = \sum_{j=1}^{\eta} \Big((\bar{\Upsilon}_{ij}{}^T \otimes \bar{\Xi}_{ij}) \mathrm{vec}[\bar{K}_j] \Big) . \tag{A.22}$$

This leads to a system of linear equations for K and can be represented in matrix form as

$$\begin{bmatrix} \mathrm{vec}[\bar{K}_1] \\ \vdots \\ \mathrm{vec}[\bar{K}_\eta] \end{bmatrix} = - \begin{bmatrix} \bar{\Upsilon}_{11}{}^T \otimes \bar{\Xi}_{11} & \dots & \bar{\Upsilon}_{1\eta}{}^T \otimes \bar{\Xi}_{1\eta} \\ \vdots & \ddots & \vdots \\ \bar{\Upsilon}_{\eta 1}{}^T \otimes \bar{\Xi}_{\eta 1} & \dots & \bar{\Upsilon}_{\eta\eta}{}^T \otimes \bar{\Xi}_{\eta\eta} \end{bmatrix}^{-1} \begin{bmatrix} \bar{\Psi}_1 \\ \vdots \\ \bar{\Psi}_\eta \end{bmatrix} \tag{A.23}$$

Weighting Matrix

It was shown in [2] that (3.34) can be solved using Lagrange multipliers[1], what leads to the conditions for optimality given by

$$\Gamma \begin{bmatrix} W^T \\ \Lambda \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{n \times n_i} \\ \mathbf{I}_{n_i \times n_i} \end{bmatrix} \quad \text{with} \quad \Gamma = \begin{bmatrix} \Phi & e \\ e^T & \mathbf{0}_{n_i \times n_i} \end{bmatrix} . \tag{A.24}$$

The general solution of (A.24) in terms of ν can be obtained by using the Moore-Penrose pseudo inverse, leading to

$$\begin{bmatrix} W^T \\ \Lambda \end{bmatrix} = \begin{bmatrix} \Phi^{-1} e \left(e^T \Phi^{-1} e \right)^{-1} \\ -\left(e^T \Phi^{-1} e \right)^{-1} \end{bmatrix} + \Gamma^0 \nu \tag{A.25}$$

where Γ^0 is a matrix containing the vectors spanning the null space of Γ. Since (A.25) is usually under determined, the additional minimization problem

$$\begin{aligned} W &= \underset{W}{\mathrm{argmin}} \ \mathrm{tr} \left(W W^T \right) \\ &\quad \text{s.t.} \quad \text{(A.24)} \end{aligned} \tag{A.26}$$

[1] To simplify notation the subscripts of $W_{[ji]}$ and Φ_{ji} will be droped.

is introduced to calculate ν in (A.25). By using (A.25) the cost function $J = \mathrm{tr}\left(WW^T\right)$ of (A.26) becomes

$$\begin{aligned} &J = (\delta + \Gamma_1^0\nu)^T(\delta + \Gamma_1^0\nu) \\ &\text{with} \quad \delta = \Phi^{-1}e\left(e^T\Phi^{-1}e\right)^{-1} \end{aligned} \tag{A.27}$$

where Γ_1^0 is the part of Γ^0 corresponding to W^T in (A.25). Since the derivative of (A.27) with respect to ν is given by

$$\frac{\mathrm{d}J}{\mathrm{d}\nu} = 2(\Gamma_1^0\nu + \delta)^T\Gamma_1^0$$

ν becomes

$$\nu = -(\Gamma_1^0)^T\delta.$$

Using (A.28) in (A.25) the optimal W can be written as

$$W = \delta^T\left(\mathbf{I} - \Gamma_1^0(\Gamma_1^0)^T\right). \tag{A.28}$$

Appendix B

Additional plots

In this appendix additional plots are included to show some details on the performance of the investigated algorithms. Explanations age given below the figures. For details please refer to Section 4.3.1 and Section 4.3.2.

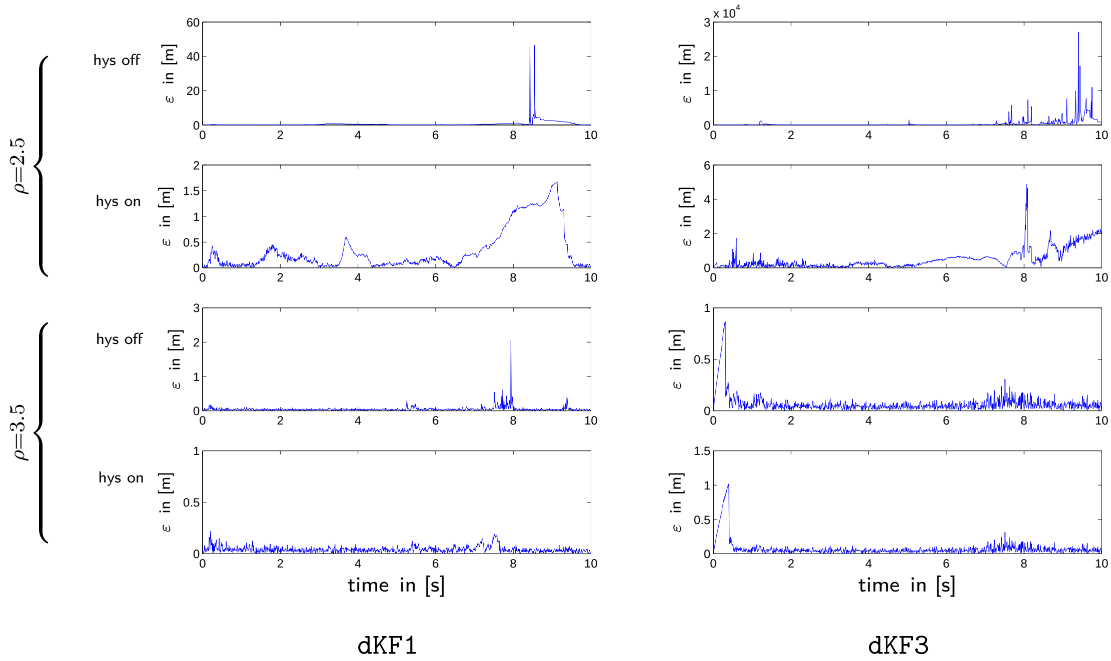
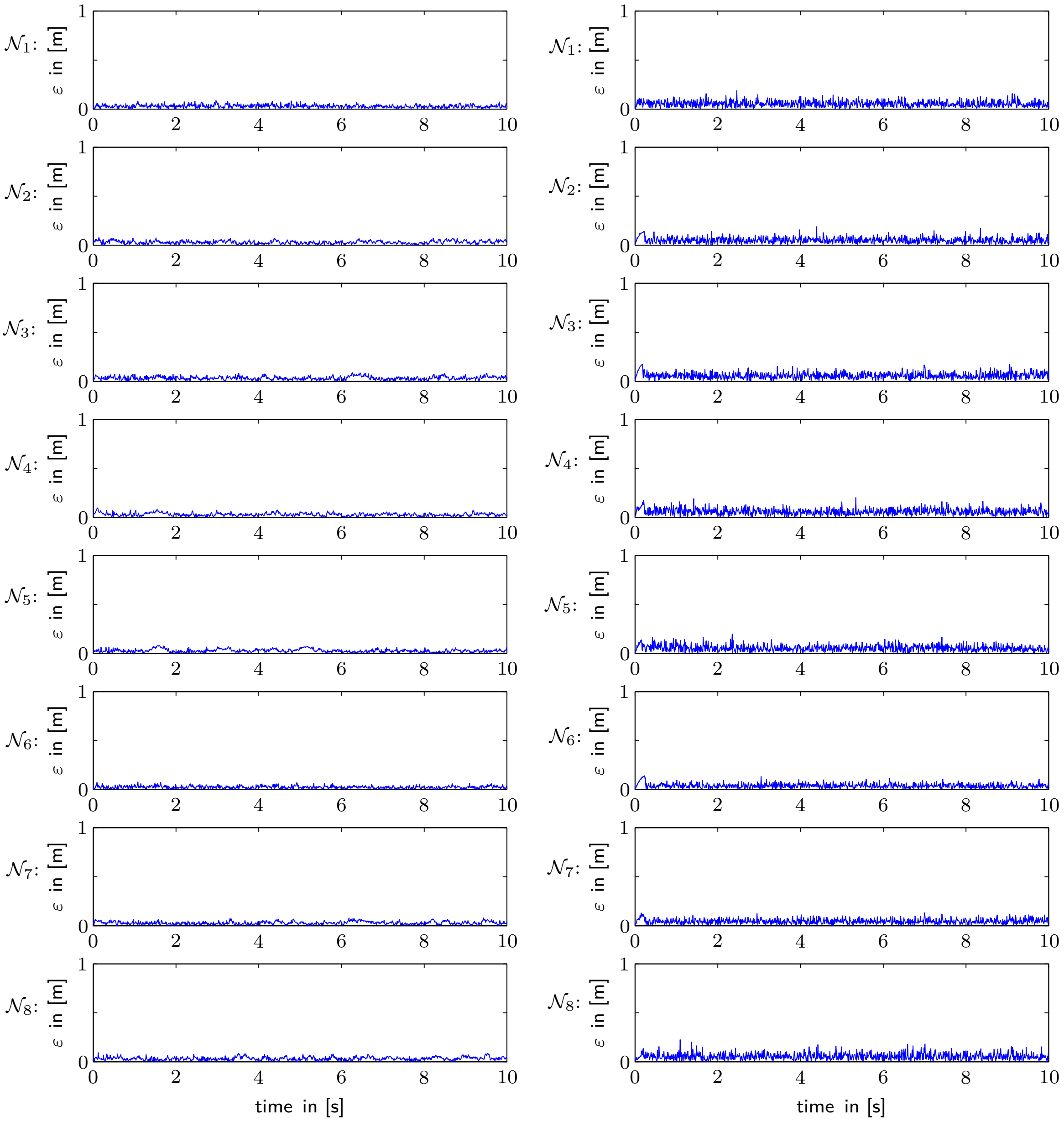


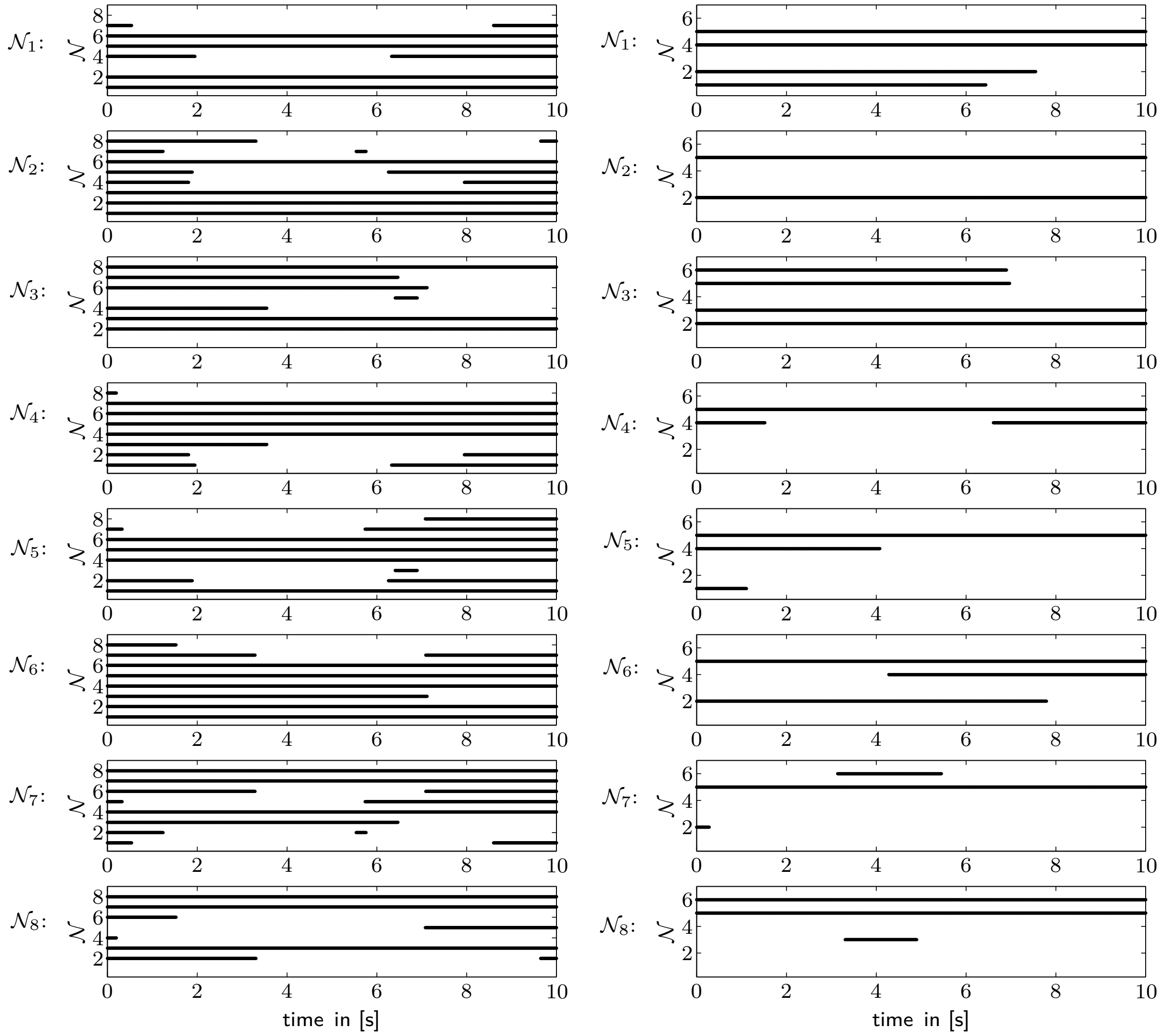
Fig. B.1: Comparison of estimation error ε of node $\mathcal{N}_8$ over time. All four scenarios visualized in Fig. 4.5 are given from top to bottom. From left to right, the performance of dKF1 and dKF3 can be compared. This is the unscaled version of Fig. 4.5.



dKF1, switched model

dKF3, switched model

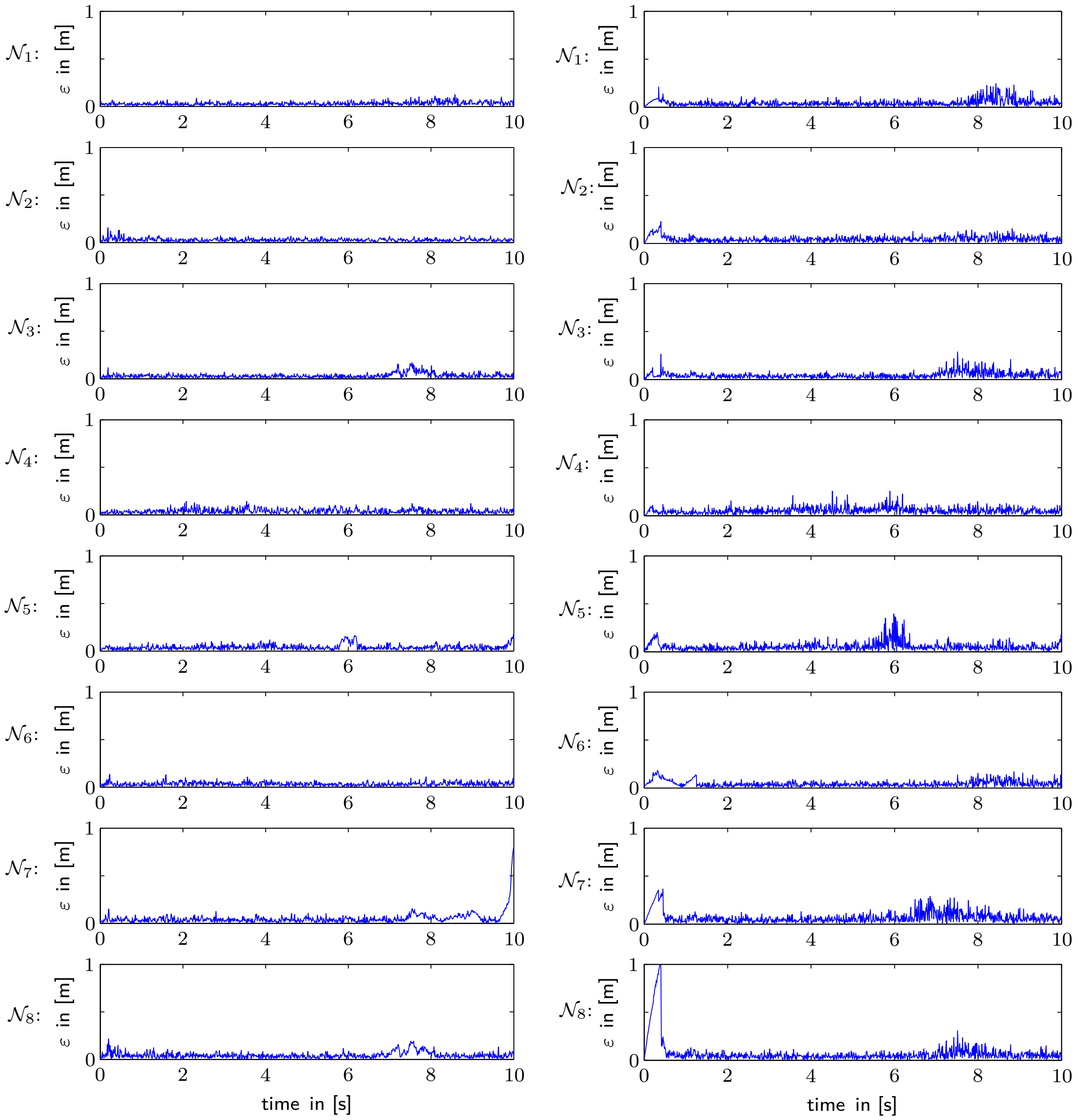
Fig. B.2: Comparison of the position estimation error ε (see (4.19)) for algorithm dKF1 and dKF3 when the Kalman gain is calculated online and the switched model is used (third scenario in Fig. 4.3) with the static graph scenario. The error is plotted for every node $\mathcal{N}_i$ from top to bottom of the figure.



Neighboring nodes $\mathcal{N}_{[i]}$.

Reachable reference nodes $\mathcal{M}_{[i]}$.

Fig. B.3: Visualization of the graph changes in $\mathcal{G}^{\eta}$ and $\mathcal{G}^{\mu}$ when using a communication radius of $\rho = 3.5$m and a hysteresis to reduce switches (forth scenario in Fig. 4.5) in the dynamic graph scenario. The communication topology is given for every mobile node from top to bottom of the figure.



dKF1 dKF3

Fig. B.4: Comparison of estimation error ε (see (4.19)) for algorithm dKF1 and dKF3 when using a communication radius of $\rho = 3.5$m and a hysteresis to reduce switches (forth scenario in Fig. 4.5) in the dynamic graph scenario. The error is plotted for every node $\mathcal{N}_i$ from top to bottom of the figure.

Bibliography

[1] Peter Alriksson. *State Estimation for Distributed and Hybrid Systems*. PhD thesis, Department of Automatic Control, Lund University, Sweden, September 2008.

[2] Peter Alriksson and Anders Rantzer. Model based information fusion in sensor networks. *Proceedings of the 17th IFAC World Congress*, 2008.

[3] Brian D. O. Anderson, I. Moore, and John Barratt. *Optimal Filtering*. Prentice-Hall, 1979.

[4] Dennis S. Bernstein. Some open problems in matrix theory arising in linear systems and control. *Linear Algebra and its Applications*, pages 162–164:409–432, 1992.

[5] Mike Brookes. The matrix reference manual, Imperial College, London, UK. [online 28.06.2009] http://www.ee.ic.ac.uk/hp/staff/dmb/matrix/intro.html, 2005.

[6] Toshio M. Chin, William C. Karl, and Alan S. Willsky. Sequential filtering for multi-frame visual reconstruction. *Signal Processing*, 28(3):311 – 333, 1992.

[7] P.A. Cook. Stable control of vehicle convoys for safety and comfort. *Automatic Control, IEEE Transactions on*, 52(3):526–531, March 2007.

[8] Craig Gotsman and Yehuda Koren. Distributed graph layout for sensor networks. In *In Proc. Internat. Symposium on Grpah Drawing*, pages 273–284, 2004.

[9] Meng Ji and M. Egerstedt. Distributed formation control while preserving connectedness. *Decision and Control, 2006 45th IEEE Conference on*, pages 5962–5967, Dec. 2006.

[10] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME-Jurnal of Basic Engineering*, 82: Series D:35–45, 1960.

[11] U.A. Khan and J.M.F. Moura. Distributing the kalman filter for large-scale systems. *Signal Processing, IEEE Transactions on*, 56(10):4919–4935, Oct. 2008.

[12] M. Mesbahi. On a dynamic extension of the theory of graphs. volume 2, pages 1234–1239 vol.2, 2002.

[13] Angus P. Andrews Mohinder S. Grewal. *Kalman Filtering: Theory and Practice Using MATLAB*. John Wiley & Sons, Inc., second edition edition, 2001.

[14] R. Olfati-Saber and J.S. Shamma. Consensus filters for sensor networks and distributed sensor fusion. In *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, pages 6698–6703, Dec. 2005.

[15] Nissanka B. Priyantha, Hari Balakrishnan, Erik Demaine, and Seth Teller. Anchor-free distributed localization in sensor networks, 2003.

[16] Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. The cricket location-support system. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 32–43, New York, NY, USA, 2000. ACM.

[17] S.I. Roumeliotis and G.A. Bekey. Distributed multirobot localization. *Robotics and Automation, IEEE Transactions on*, 18(5):781–795, Oct 2002.

[18] Adam Smith, Hari Balakrishnan, Michel Goraczko, and Nissanka Priyantha. Tracking moving devices with the cricket location system. In *MobiSys '04: Proceedings of the 2nd international conference on Mobile systems, applications, and services*, pages 190–202, New York, NY, USA, 2004. ACM.

[19] A. Speranzon, C. Fischione, and K.H. Johansson. Distributed and collaborative estimation over wireless sensor networks. In *Decision and Control, 2006 45th IEEE Conference on*, pages 1025–1030, Dec. 2006.

[20] Roger Wattenhofer Thomas Locher, Pascal von Rickenbach. *Distributed Computing and Networking*, chapter Sensor Networks Continue to Puzzle: Selected Open Problems, pages 25–38. 2008.

[21] Soo-Yeong Yi and Byoung-Wook Choi. Autonomous navigation of indoor mobile robots using a global ultrasonic system. *Robotica*, 22(04):369–374, 2004.

[22] Jaegeol Yim, Chansik Park, Jaehun Joo, and Seunghwan Jeong. Extended kalman filter for wireless lan based indoor positioning. *Decision Support Systems*, 45(4):960 – 971, 2008. Information Technology and Systems in the Internet-Era.

[23] M.M. Zavlanos and G.J. Pappas. Distributed connectivity control of mobile networks. *Robotics, IEEE Transactions on*, 24(6):1416–1428, Dec. 2008.