Distributed target tracking in wireless camera networks

by

Sandeep Katragadda

Bachelor in Computer Science and Engineering 2009

Master in Information Technology 2012

A dissertation submitted to

The School of Electronic Engineering and Computer Science

in partial fulfilment of the requirements for the Degree of

Doctor of Philosophy

in the subject of

Interactive and Cognitive Environments

Queen Mary University of London

Mile End Road

E1 4NS, London, UK

June 2017

Acknowledgments

This PhD Thesis has been developed in the framework of, and according to, the rules of the Erasmus Mundus Joint Doctorate on Interactive and Cognitive Environments EMJD ICE [FPA n° 2010-0012] with the cooperation of the following Universities:

Alpen-Adria-Universität Klagenfurt – AAU

Queen Mary, University of London – QMUL

Technische Universiteit Eindhoven – TU/e

Università degli Studi di Genova – UNIGE

Universitat Politècnica de Catalunya – UPC

Acknowledgements

I wish to express my deep sense of gratitude to Prof. Andrea Cavallaro (my primary supervisor) for his timely and required guidance, constant supervision and for providing necessary information about this research. I wish to express my gratitude to Prof. Carlo Regazzoni (my secondary supervisor) for his support during my stay in University of Genova, Italy. I am thankful to Prof. Bernhard Rinner for supporting me to start this PhD in Alpen-Adria University of Klagenfurt, Austria. I would also like to express my sincere thanks to all my colleagues for their valuable suggestions and support. Finally, yet importantly, I would like to express my heartfelt thanks to my beloved parents for their blessings, my friends for their help and wishes, and to God who made all things possible.

I, Sandeep Katragadda, confirm that the research included within this thesis is my own work, that this is duly acknowledged below and my contribution indicated. Previously published material is also acknowledged below.

I attest that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other Intellectual Property Right, or contain any confidential material.

I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university.

Signature: Sandeep Katragadda

Date: 20 June 2017

First supervisor | Second supervisor | Author

Prof. Andrea Cavallaro | **Prof. Carlo S. Regazzoni** | **Sandeep Katragadda**

Distributed target tracking in wireless camera networks

Abstract

Distributed target tracking (DTT) is desirable in wireless camera networks to achieve scalability and robustness to node or link failures. DTT estimates the target state via information exchange and fusion among cameras. This thesis proposes new DTT algorithms to handle five major challenges of DTT in wireless camera networks, namely non-linearity in the camera measurement model, temporary lack of measurements (benightedness) due to limited field of view, redundant information in the network, limited connectivity of the network due to limited communication ranges and asynchronous information caused by varying and unknown frame processing delays. The algorithms consist of two phases, namely estimation and fusion. In the estimation phase, the cameras process their captured frames, detect the target, and estimate the target state (location and velocity) and its uncertainty using the Extended Information Filter (EIF) that handles non-linearity. In the fusion phase, the cameras exchange their local target information with their communicative neighbours and fuse the information. The contributions of this thesis are as follows. The target states estimated by the EIFs undergo weighted fusion. The weights are chosen based on the estimated uncertainty (error covariance) and the number of nodes with redundant information such that the information of benighted nodes and the redundant information get lower weights. At each time step, only the cameras having the view of the target and the cameras that might have the view of the target in the next time step participate in the fusion (tracking). This reduces the energy consumption of the network. The algorithm selects the cameras dynamically by using a threshold on their shortest distances (in the communication graph) from the cameras having the view of the target. Before fusion, each camera predicts the target information of other cameras to temporally align its information with the (asynchronous) information received from other cameras. The algorithm predicts the target state using the latest estimated velocity of the target. The experimental results show that the proposed algorithms achieve higher tracking accuracy than the state of the art under the five DTT challenges.

Contents

[1] This chapter is completely taken from [C5].
[2] This chapter is completely taken from [C3].
[3] This chapter is completely taken from [C2].

[4]This chapter is completely taken from [C1].

Published work

[C1] S. Katragadda and A. Cavallaro. A batch asynchronous tracker for wireless smart-camera networks. *Proc. of the 14th IEEE Int. Conf. on Advanced Video and Signal based Surveillance*, Lecce, Italy, 29 Aug - 1 Sep 2017. [ACCEPTED]

[C2] S. Katragadda, C. S. Regazzoni and A. Cavallaro. Average consensus-based asynchronous tracking. *Proc. of the 42nd IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, New Orleans, USA, 5-9 Mar 2017.

[C3] S. Katragadda and A. Cavallaro. Neighbour consensus for distributed visual tracking. *Proc. of the 10th IEEE Int. Conf. on Intelligent Sensors, Sensor Networks and Information Processing*, Singapore, 7-9 Apr 2015.

[C4] S. Katragadda, J. C. SanMiguel and A. Cavallaro. The Costs of Fusion in Smart Camera Networks. *Proc. of the 8th ACM/IEEE Int. Conf. on Distributed Smart Cameras*, Venezia Mestre, Italy, 4-7 Nov 2014.

[C5] S. Katragadda, J. C. SanMiguel and A. Cavallaro. Consensus protocols for distributed tracking in wireless camera networks. *Proc. of the 17th Int. Conf. on Information Fusion*, Salamanca, Spain, 7-10 Jul 2014.

Electronic preprints are available at *http://www.eecs.qmul.ac.uk/~andrea/publications.html*

List of symbols

α^{ij}	Relative capturing offset between C^i and C^j
α	The upper bound on the relative capturing offsets among the cameras
α^{max}	The maximum possible upper bound on the relative capturing offsets among the cameras
a	Power amplification factor ($Joules/bit/m^2$)
B	Number of complete benighted nodes in the network
$\mathbf{C}$	Set of cameras in the network
C^i, C^j	Cameras with indices i and j
c	Weight used in the average consensus update
Δ_{max}	The maximum degree of the communication graph
D	Diameter of the network
$\hat{D}$	Threshold hop distance that defines sink nodes
D_k^i	Hop distance of C^i from the nearest viewing node at k
E_t	Energy consumed for transmission
E_r	Energy consumed for reception
E_c	Energy ($Joules/bit$) consumed for running the communication components
ε	Average root mean square error
$f(\cdot)$	State transition function (target motion model)
$\mathbf{F}$	State transition matrix (if linear motion model)
F	Index representing the fusion centre
γ	Periodicity of consensus iterations
H^i	Homography matrix of C^i
$h^i(\cdot)$	State to measurement transition function of C^i
$\mathbf{H}^i$	State to measurement transition matrix (if linear measurement model) of C^i
$\mathbf{I}_k$	Set of inactive nodes of target at time k
$\mathcal{I}_k^i$	Frame captured by C^i at the local time k
$\mathbf{J}_{f,\mathbf{x}}(\cdot)$	Jacobian of $f(\cdot)$ w.r.t. $\mathbf{x}$
$\mathbf{J}_{h,\mathbf{x}}^i(\cdot)$	Jacobian of $h^i(\cdot)$ w.r.t $\mathbf{x}$
k	Time instant
$\mathbf{K}_k^i$	Time window around the capturing instant of k of C^i considered in batch fusion
$\mathcal{K}^{i,r}$	Set of capturing instants of C^i during r^{th} simulation run
L	Number of consensus iterations between consecutive captures
M	Number of Monte-Carlo simulations
m	Size of the measurement vector $\mathbf{z}_k^i$
n	Size of the state vector $\mathbf{x}_k$
N	Number of cameras
$\hat{N}$	Upper bound on the number of cameras
N_t	Number of trajectories
$\mathcal{N}^i$	Communicative neighbourhood of C^i
$\mathbf{N}_k$	Set of neighbouring nodes of target at time k
$\mathbf{N}_k^i$	Set of all N-Nodes in the communication range of C^i at time k
$\mathcal{P}_k^i$	Set of players in the FoV of C^i at the local time k
$p(\cdot)$	Probability density function

$\mathbf{P}_k^i$	Target state error covariance of C^i corresponding to the local time k before fusion
$\hat{\mathbf{P}}_k^i$	Target state error covariance of C^icorresponding to the local time k after fusion
$\hat{\mathbf{P}}_k^F$	Target state error covariance of fusion centre corresponding to the local time k after fusion
$Q(\cdot)$	Process noise covariance function
q	Process noise intensity in the unit interval
r_v	Viewing range of cameras
r_c	Communication range of cameras
$\mathbf{R}^i$	Measurement noise covariance matrix of C^i
$\mathbf{S}_k$	Set of sink nodes of target at time k
$\top$	Transpose
T	Inter-capturing period of cameras
t	Trajectory index
τ_k^i	Processing time of the frame captured by C^i at the local time k
τ^{max}	The maximum processing time of the captured frames of all cameras
τ^{min}	The minimum processing time of the captured frames of all cameras
$\mathbf{u}_k^i$	Target likelihood vector of C^i corresponding to the local time k
$\mathbf{U}_k^i$	Target likelihood matrix of C^i corresponding to the local time k
$\mathbf{V}_k$	Set of viewing nodes of target at time k
$\mathbf{V}_{k^+}$	Set of future viewing nodes of target at time $k+\Delta k$ $(0 \leq \Delta k \leq T)$
w_k^i	Weight given to the information of C^i corresponding to time k
$w_{k(l)}^i$	Weight given to the information of C^i corresponding to time k during the l^{th} consensus iteration of ICI
$\mathbf{x}_k$	Ground truth target state
$\mathbf{x}_k^i$	Estimated target state of C^i corresponding to the local time k
$\bar{\mathbf{x}}_k^i$	Predicted target state of C^i corresponding to the local time k
$\hat{\mathbf{x}}_k^F$	Fused target state of fusion centre corresponding to the local time k
$\hat{\mathbf{x}}_k^i$	Fused target state of C^i corresponding to the local time k
$\mathbf{y}_k^i$	Estimated target information vector of C^i corresponding to the local time k
$\mathbf{Y}_k^i$	Estimated target information matrix of C^i corresponding to the local time k
$\bar{\mathbf{y}}_k^i$	Predicted target information vector of C^i corresponding to the local time k
$\bar{\mathbf{Y}}_k^i$	Predicted target information matrix of C^i corresponding to the local time k
$\hat{\mathbf{y}}_k^F$	Fused target information vector of fusion centre corresponding to the local time k
$\hat{\mathbf{Y}}_k^F$	Fused target information matrix of fusion centre corresponding to the local time k
$\hat{\mathbf{y}}_k^i$	Fused target information vector of C^i corresponding to its local time k
$\hat{\mathbf{Y}}_k^i$	Fused target information matrix of C^i corresponding to its local time k
$\mathbf{y}_{k(l)}^i$	Target information vector of C^i corresponding to its local time k before starting the consensus iteration at $k+l$
$\mathbf{Y}_{k(l)}^i$	Target information matrix of C^i corresponding to its local time k before starting the consensus iteration at $k+l$
$\tilde{\mathbf{y}}_{k\|k'}^i$	Predicted target information vector of C^i corresponding to its local time k based on its target information corresponding to k'
$\tilde{\mathbf{Y}}_{k\|k'}^i$	Predicted target information matrix of C^i corresponding to its local time k based on its target information corresponding to k'
$\mathbf{z}_k^i$	Target measurement (pixel location) in the image plane of C^i captured at the local time k
$\mathbf{Z}_{1:k}$	Set of target measurements of all cameras C^i from time 1 to k
$\hat{\mathbf{z}}_k^i(r,t)$	Estimated target measurement of trajectory t corresponding to the local time k of C^i after r simulation runs
$\zeta_k^{i,t}$	Ground truth target measurement of trajectory t corresponding to the local time k of C^i

List of abbreviations

A-consensus	Average Consensus
AC	Average Consensus
ACAF	Average Consensus based Asynchronous Filter
aCDTT	asynchronous Consensus-based Distributed Target Tracking
BCI	Batch Covariance Intersection
CCI	Centralised Covariance Intersection
CEIF	Centralised Extended Information Filter
CEN	Centralised fusion
CH	Cluster Head
CubKF	Cubature Kalman Filter
DAPF	Distributed Asynchronous Particle Filter
DHIF	Distributed Hybrid Information Fusion
EICF	Extended Information Consensus Filter
EIF	Extended Information Filter
EIWCF	Extended Information Weighted Consensus Filter
EKCF	Extended Kalman Consensus Filter
EKF	Extended Kalman Filter
FC	Fusion Centre
FCI	Fast Covariance Intersection
FoV	Field of View
GKCF	Generalised Kalman Consensus Filter
HCMCI	Hybrid Consensus on Measurements and Consensus on Information
ICF	Information Consensus Filter
ICI	Iterative Covariance Intersection
IF	Information Filter
IFCI	Improved Fast Covariance Intersection
IWCF	Information Weighted Consensus Filter
KCF	Kalman Consensus Filter
KF	Kalman Filter
KL	Kullback-Leibler
KLA	Kullback-Leibler Average
MTE	Mean Tracking Error
N-consensus	Neighbour consensus
N-Nodes	Neighbouring nodes
NC	Neighbour consensus
pdf	probability density function
PF	Particle Filter
SAF	Sequential Asynchronous Filter
SAF-ED	Sequential Asynchronous Filter with Estimated Delays
UKF	Unscented Kalman Filter
WCN	Wireless Camera Network
WSN	Wireless Sensor Network

Chapter 1

Introduction

1.1 Motivation

Tracking is the process of estimating the locations of objects of interest (targets) across time. In visual tracking, cameras process their captured frames with computer-vision algorithms to detect the targets. The detected targets are represented using blobs [59], contours [88], bounding boxes [44] or their centroid locations in the image plane. The image plane detections (measurements) are often noisy or erroneous due to limitations in computer vision algorithms or scene constraints such as low illumination and low image sensor quality. Moreover, a camera may not detect the target all the time due to the directional sensing and/or occlusions [87]. To improve the coverage and reliability of tracking, multiple cameras are used [25, 55, 58, 90]. Fusion of information from multiple cameras aims at improving the accuracy of tracking. This thesis focuses on target tracking using wireless camera networks (WCNs).

Information fusion in WCNs can be centralised, decentralised or distributed [76]. In centralised fusion, all camera nodes send their local information to a fusion centre (FC) via single-hop or multi-hop communications [82]. Due to the limited communication ranges of wireless devices, routing protocols are used to establish multi-hop communication between each camera and the FC. As the FC takes the responsibility of tracking, there is less processing load on the cameras. In addition, FC has the entire information so the results are always optimal. The centralised fusion is vulnerable to node failures, especially the FC failure, and adding or removing some cameras requires entire routing information to be updated (i.e. not scalable). Moreover,

the network traffic is very high near to the FC causing communication failures. The decentralised scheme [34] considers various FCs that collect and fuse information from nodes in their neighbourhood. The allocation of nodes to FCs can be static [34] or dynamic [56]. To support topology changes and scalability, dynamic decentralisation (or clustering) is preferred. Similar to the centralised schemes, decentralised schemes are also vulnerable to node failures. Moreover, the static schemes are not scalable. In distributed fusion [73], each node runs an identical peer-to-peer algorithm to exchange information with other nodes. Flooding [23], consensus [65] and token passing [37] are widely used distributed fusion schemes. Distributed information fusion is desirable for target tracking in WCNs to achieve scalability and robustness to node or link failures and aims to achieve comparable performance to centralised algorithms, where global communications are assumed [76].

There exist five main challenges in distributed target tracking in WCNs: non-linearity, benighted cameras, redundant information, asynchronous measurements and limited network connectivity.

Non-linearity: Each camera estimates the target state (e.g. location on a reference plane) corresponding to the capturing instant via filtering. The filter uses the previous knowledge of the target, the measurement in the captured frame (if available), target motion model and the camera measurement model. The target state dynamics on the reference plane and the corresponding target measurements provided by the cameras (e.g. target coordinates in the image plane of each camera) are *non-linearly* related to each other [36]. Hence, the target state estimation techniques must consider the non-linear measurement model.

Benightedness: As the target moves in the scene, the set of cameras simultaneously detecting the same target (viewing nodes) changes over time [C4]. Due to the limited field of view (FoV) of the cameras, there are *benighted nodes* that do not have target measurements. Though the cameras are physically close to the target they might not have measurements because of the directional FoVs. See Figure 1.1. The thesis defines two types of benightedness: partial and complete. If a camera has the knowledge of previous target state but no current measurement, the thesis refers this as *partial benightedness* (p-benightedness) and the nodes are called *p-benighted nodes*. If a camera does not have any information of target (e.g. the camera is newly added to the network), the thesis refers this as *complete benightedness* (c-benightedness) and the nodes are called *c-benighted nodes*. Information fusion should consider the benightedness such that

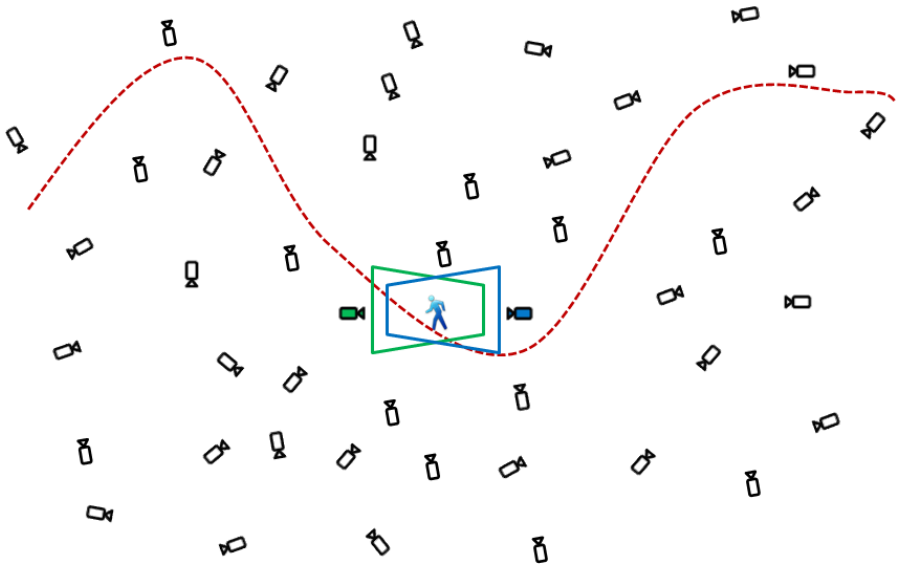


Figure 1.1: A wireless camera network with two viewing cameras (coloured) and many benighted cameras (white). The polygons represent the field of views of the cameras.

the c-benighted nodes do not affect the fusion result and the p-benighted nodes (nodes without measurements) are weighted less compared to the viewing nodes (nodes with measurements).

Redundancy: If all the cameras have the same previous knowledge of the target, it results in information redundancy in the network. The *redundancy* is proportional to the number of cameras in the network. Though the cameras do not provide any new information, the redundant information affects the fusion result (e.g. averaging). The distributed fusion must be done such that the effect is mitigated.

Asynchronous measurements: Most multi-camera tracking algorithms assume that the cameras in the network capture the frames synchronously [47, 50, 81, C3, C5]. However, this is not the case in reality. *Asynchronous captures* are caused by unknown relative clock offsets and processing delays generated by the local computer-vision pipeline. The inherent drifts in the local clock frequencies of the nodes result in relative clock offsets [72]. The local clock frequencies may drift between 1 and 100 parts per million (ppm) [27, 72]. While time synchronisation protocols can estimate and compensate for the timing offsets, they significantly increase the communication overhead and therefore energy consumption, thus reducing the network lifetime [32, 77]. The processing delays are significant due to the large amount of frame data to be processed to produce object detections [18, 67, 68, 74]. The local frame-processing delays are not negligible ($\approx$40ms) and may vary from frame to frame and from camera to camera as they depend on the frame size, scene complexity, processing capabilities of the node, the object

detection algorithms used and the number of observed targets [68]. Additionally, there exist delays in communication that depend on the number of participating nodes and the amount of data exchanged [80, 83]. Even if delays are comparable among camera nodes, the local information in a camera corresponds to the instant of capturing and not to the instant of transmission. While fusion of synchronous information increases tracking accuracy [50, C5], when information is exchanged asynchronously the accuracy may degrade significantly.

Limited connectivity: The viewing nodes might not be in communication range of each other because of the limited communication ranges (see Figure 1.2). Such *limited network connectivity* makes the information exchange and fusion among the viewing nodes challenging. The thesis classifies WCN connectivity into three types: *full connectivity* where the communication range of each camera is so high that any pair of cameras in the network are always single-hop neighbours, *full connectivity among viewing nodes* where any two viewing nodes are always single-hop neighbours, and *limited connectivity* where the communication ranges of the cameras are very low and make even the viewing nodes multi-hop neighbours.

During distributed fusion in fully connected networks (e.g. when cameras are up to $100m$ away from each other [87]) or in fully connected viewing nodes, the nodes receive and fuse the target information from all viewing nodes in the network [22, C4]. Distributed fusion in the case of limited connectivity can be done through consensus [50, C4]. Consensus algorithms are distributed protocols that aim at reaching an agreement on a decision variable using iterative peer-to-peer communication among the nodes [65]. Unlike other distributed fusion algorithms [9,32,40,92], consensus-based algorithms do not require full connectivity among viewing nodes nor prior knowledge of the routing tables [C4]. Average consensus (A-consensus) is a widely used consensus algorithm for target tracking in wireless sensor networks (WSNs) [6, 35] and WCNs [21, 50, C5]. A-consensus aims at having decisions at all nodes (e.g. target state) to converge to their average. The Extended Kalman Consensus Filter (EKCF) [21] handles non-linearity by using the Extended Kalman Filter (EKF) and limited connectivity by using A-consensus. The Information Consensus Filter (ICF) [15] handles benightedness by weighting the local state estimates based on the uncertainty and limited connectivity by using A-consensus. A-consensus approaches require the knowledge of the maximum degree of connectivity. The Iterative Covariance Intersection (ICI) [41] handles benightedness and limited connectivity without requiring such knowledge and achieves better performance than ICF. There exist sequential

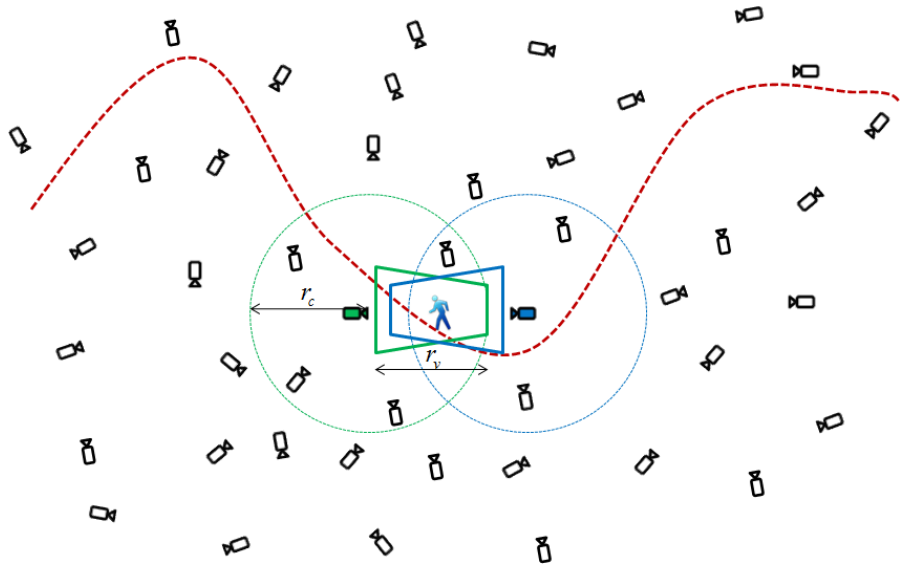


Figure 1.2: A wireless camera network with two viewing cameras (coloured). The dotted circles represent the communication range of the viewing cameras. The polygons represent the field of views of the cameras. Key – r_c: communication range, r_v: viewing range.

and batch processing algorithms to handle asynchronous information [9]. The asynchronous Consensus-based Distributed Target Tracking (aCDTT) [33] achieves consensus on the maximum certain state without performing fusion. These algorithms do not handle the remaining challenges. Moreover, the consensus algorithms achieve consensus among all network nodes, and therefore the total energy consumption for communication and computation increases with the number of nodes (for a given number of viewing nodes) [C4]. Hence, along with the mentioned challenges, reducing the participation of non-viewing nodes and thereby the total energy consumption in the network is also desirable because limited energy consumption is important for any WSN and in particular, WCN [52].

1.2 Problem formulation

Consider a WCN consisting of N cameras represented by $\mathbf{C} = \{C^1, C^2, ..., C^N\}$ to track a target moving on a common ground plane. Each camera C^i $(1 \leq i \leq N)$ consists of an image sensor, a processor and a wireless communication module. Each camera C^i has a directional FoV with viewing range r_v and communication range r_c (see Figure 1.2). Let $\mathcal{N}^i$ be the set of cameras in the communication range of C^i.

Let the target dynamics for a temporal interval Δk be

$$\mathbf{x}_k = f(\mathbf{x}_{k-\Delta k}, \Delta k) + \mathbf{w}_{k-\Delta k}. \tag{1.1}$$

Here, $\mathbf{x}_k = [x_k\ y_k\ \dot{x}_k\ \dot{y}_k]^\top$ is the target state at k where $[x_k\ y_k]^\top$ and $[\dot{x}_k\ \dot{y}_k]^\top$ are the position and velocity of the target at k on the ground plane, respectively, and $n = 4$ is the size of the state vector $\mathbf{x}_k$. The function $f(\cdot)$ is the state transition function from $k - \Delta k$ to k and $\mathbf{w}_{k-\Delta k}$ is the process noise which is assumed to be Gaussian with zero mean and covariance matrix $\mathbf{Q}(k - \Delta k, k)$. This thesis considers the motion model $f(\cdot)$ to be linear (nearly constant velocity model) and is defined by the state transition matrix $\mathbf{F}(k - \Delta k, k)$ and the process noise covariance matrix $\mathbf{Q}(k - \Delta k, k)$ as in [9, 32, 40]:

$$\mathbf{x}_k = \mathbf{F}(k - \Delta k, \Delta k)\mathbf{x}_{k-\Delta k} + \mathbf{w}_{k-\Delta k}. \tag{1.2}$$

Here,

$$\begin{aligned} \mathbf{F}(k - \Delta k, k) &= \begin{bmatrix} \mathbf{I}_2 & (\Delta k)\mathbf{I}_2 \\ \mathbf{0}_2 & \mathbf{I}_2 \end{bmatrix}, \\ \mathbf{Q}(k - \Delta k, k) &= q^2 \begin{bmatrix} d(\Delta k, 3)\mathbf{I}_2 & d(\Delta k, 2)\mathbf{I}_2 \\ d(\Delta k, 2)\mathbf{I}_2 & d(\Delta k, 1)\mathbf{I}_2 \end{bmatrix}, \end{aligned} \tag{1.3}$$

where $\mathbf{I}_2$ and $\mathbf{0}_2$ are the 2×2 unit and zero matrices respectively, q is the process noise intensity in the unit interval and $d(\Delta k, b) = \frac{|\Delta k|^b}{b}$. The target is visible in at least one camera at any time step k and it does not move more than $2r_v$ between consecutive frame captures of cameras.

Let $\mathbf{z}_k^i$ be the measurement of C^i corresponding to the target state $\mathbf{x}_k$. The measurement model of camera C^i is defined as

$$\mathbf{z}_k^i = h^i(\mathbf{x}_k) + \mathbf{v}_k^i, \tag{1.4}$$

where $\mathbf{v}_k^i$ is an additive Gaussian noise with zero mean and covariance $\mathbf{R}^i = \sigma^2\mathbf{I}_2$. Here σ is the standard deviation of the measurement error. The function $h^i(\cdot)$ is the state to measurement transition function that encodes calibration information of C^i, and $m = 2$ is the size of the measurement vector $\mathbf{z}_k^i$. If C^i captures a frame $\mathcal{I}_k^i$ at its local time k, it performs target detection to obtain the measurement $\mathbf{z}_k^i$. Let τ_k^i be the corresponding frame processing time. At time step k, only a subset of cameras, $\mathbf{V}_k$, can view the target because of the directionality and limited viewing range. If there are no viewing nodes at a time instant, the location is called blind region. The thesis assumes that there are no blind regions. Let T be the desired inter-capturing period for all cameras. In the synchronous case, the cameras capture frames at $\{0, T, 2T, ...\}$. In the asynchronous case, the cameras capture frames at different instants. Let α^{ij} be the relative offset

in the frame captures of C^i and C^j. Note that $\alpha^{ij} = 0$ represents the synchronous captures of C^i and C^j.

The thesis assumes that the cameras are calibrated and there are no false measurements (i.e. no false positive or false negative detections). The measurements are independent from camera to camera. The cameras do not have information about the network or its neighbours such as communication topology, routing tables, and vision graph (graph representing camera pairs having overlapping FOVs). The thesis also assumes that each camera can communicate with all cameras in its communicative neighbourhood without packet losses and communication delays.

The objective of this thesis is to make each viewing camera $C^i \in \mathbf{V}_k$ estimate the target state $\hat{\mathbf{x}}_k^i$ corresponding to its capturing instant k by fusing the local and the other cameras information under the presence of above mentioned challenges.

1.3 Contributions

The detailed contributions of the thesis are:

1. *Information weighting for non-linear systems*. This thesis performs weighted information fusion in two stages. The first one is inter-camera information weighting, where each cameras estimate is weighted based on the uncertainty [6, 15]. The weights are inversely proportional to the error covariance of the estimates so the benighted nodes get lower weights. If a camera does not have the target information, the error covariance is considered to be infinite so the weight is zero. If an estimate so reliable that its error covariance is close to zero, the estimate gets the highest weight. Note that the weights are normalised before the fusion step. The second one is intra-camera information weighting where each camera weights its prior information and the measurement information differently. Intra-camera weighting handles redundancy by weighting less the redundant prior information compared to the measurement information [50]. The weighting depends on the amount of redundancy, i.e. number of nodes in the network. There are no state of the art methods that apply the two stages in non-linear systems such as cameras. This thesis uses the Extended Information Filter (EIF) that handles non-linearity to estimate the target state at each camera and apply the information weighting on the estimates to handle benightedness and redundancy [C5].

2. *Distributed selection and fusion of target location-based subnetwork*. In the existing consensus approaches, the entire network participates in consensus irrespective of the number

of cameras having target measurement [50]. To reduce the participation of benighted nodes, the thesis identifies dynamically a set of nodes in the neighbourhood of the viewing nodes, achieve consensus only among these nodes and thereby reducing the energy consumption of the network [C3]. This thesis considers only the nodes in the neighbourhood of viewing nodes up to a certain number of hops. The number is decided based on the communication and viewing ranges.

3. *Information alignment via predictions*. There are no works that deal with asynchronous information fusion in distributed visual tracking. This thesis proposes an approach to avoid fusion of asynchronous information where each camera predicts the target state estimate corresponding to its capturing instant based on the received information that might correspond to different time instants. After predicting the information of other cameras, the camera fuses the temporally aligned local and the predicted target information and updates the estimate corresponding to its capturing instant. This thesis uses the approach for the networks with limited connectivity using average consensus framework [C2] and also for the networks with full connectivity using batch fusion framework.

1.4 Organisation of thesis

This thesis is organised as follows.

Chapter 1: This chapter introduces the distributed tracking and the challenges of distributed tracking in WCNs. This also presents the problem formulation.

Chapter 2: This chapter describes the state of the art on multi-camera tracking focussing on state estimation, and centralised and distributed fusion of local state estimations.

Chapter 3: This chapter proposes the Extended Information Consensus Filter (EICF) and the Extended Information Weighted Consensus Filter (EIWCF) that handle limited connectivity, non-linearity, benightedness and redundancy. The simulations that evaluate their performance by comparing with EKCF under the presence these challenges are also presented.

Chapter 4: This chapter proposes the Neighbour consensus (N-consensus) that dynamically selects the consensus nodes. The chapter also presents the experimental performance comparison of N-consensus with EICF and ICI under the presence of limited connectivity and benightedness.

Chapter 5: This chapter proposes the Average Consensus-based Asynchronous Filter (ACAF) and the simulations that compare the performance of ACAF with ICF and aCDTT under the

presence of limited connectivity, benightedness and asynchronous measurements. The simulation considers simulated tracks as well as real tracks from the APIDIS dataset [1].

Chapter 6: This chapter proposes the Batch Asynchronous Filter (BAF) and the experimental comparison with sequential and other batch methods. The experiments are conducted using real tracks and cameras from the APIDIS dataset [1].

Chapter 7: Finally, this chapter summarises the thesis and presents the possible future research directions.

Chapter 2

State of the art

This chapter presents the state of the art on state estimation and information fusion especially focussing on challenges: non-linearity, benightedness, asynchronous measurements and limited connectivity.

2.1 State estimation

Bayesian filter [20] running at each camera C^i estimates the target probability density function (pdf) $p(\mathbf{x}^i_k|\mathbf{z}^i_{1:k})$ (hereafter referred as *posterior*) corresponding to its capturing instant k. The estimation process involves two steps, namely prediction and update. In the prediction step, a predicted posterior (hereafter referred as *prior*) is computed based on the posterior $p(\mathbf{x}^i_{k-T}|\mathbf{z}^i_{1:k-T})$ corresponding to the previous capturing instant $k-T$ and the state transition pdf $p(\mathbf{x}^i_k|\mathbf{x}^i_{k-T})$ computed using $f(\cdot)$ as

$$p(\mathbf{x}^i_k|\mathbf{z}^i_{1:k-T}) = \int p(\mathbf{x}^i_k|\mathbf{x}^i_{k-T})p(\mathbf{x}^i_{k-T}|\mathbf{z}^i_{1:k-T})d\mathbf{x}^i_{k-T}. \tag{2.1}$$

Using the current measurement $\mathbf{z}^i_k$ and the measurement model given by the function $h^i(\cdot)$, the target likelihood (hereafter referred as *likelihood*) $p(\mathbf{z}^i_k|\mathbf{x}^i_k)$ is computed. In the update step, the posterior is computed based on the prior and the likelihood using the Bayes' rule as

$$p(\mathbf{x}^i_k|\mathbf{z}^i_{1:k}) = \frac{p(\mathbf{z}^i_k|\mathbf{x}^i_k)p(\mathbf{x}^i_k|\mathbf{z}^i_{1:k-T})}{\int p(\mathbf{z}^i_k|\mathbf{x}^i_k)p(\mathbf{x}^i_k|\mathbf{z}^i_{1:k-T})d\mathbf{x}^i_k}. \tag{2.2}$$

The recursion of (2.1) and (2.2) is initialised by $p(\mathbf{x}_0^i|\mathbf{z}_0^i)$. The estimated pdf can be approximated as a Gaussian pdf represented using its mean $\mathbf{x}_k^i$ and covariance $\mathbf{P}_k^i$ as

$$\begin{aligned}\mathbf{x}_k^i &= E\{\mathbf{x}_k|\mathbf{z}_{1:k}^i\} = \int \mathbf{x}_k p(\mathbf{x}_k^i|\mathbf{z}_{1:k}^i)d\mathbf{x}_k,\\ \mathbf{P}_k^i &= E\{(\mathbf{x}_k-\mathbf{x}_k^i)(\mathbf{x}_k-\mathbf{x}_k^i)^\top|\mathbf{z}_{1:k}^i\} = \int \mathbf{x}_k\mathbf{x}_k^\top p(\mathbf{x}_k^i|\mathbf{z}_{1:k}^i)d\mathbf{x}_k.\end{aligned} \tag{2.3}$$

The Kalman Filter (KF) is the optimal approximation of the Bayesian Filter for sequential estimation in linear Gaussian systems [46]. The modified versions of KF, such as the Extended Kalman Filter (EKF) [2], the Unscented Kalman Filter (UKF) [45] and the Cubature Kalman Filter (CubKF) [3] are used to estimate in non-linear Gaussian systems. Particle Filter (PF) [24] also deals with linear/non-linear non-Gaussian systems. The following sections briefly present two filters, namely the Information Filter (IF) [15] and the Extended Information Filter (EIF) [60].

2.1.1 Information Filter

The Information Filter (IF) [15] is an alternative form of KF and uses the Fisher information matrix (i.e. the inverse of the covariance of the state estimate error). If $f(\cdot)$ and $h^i(\cdot)$ are linear functions, there exists a state transition matrix $\mathbf{F}(k-\Delta k,k)$ such that $f(\mathbf{x}_{k-\Delta k},\Delta k) = \mathbf{F}(k-\Delta k,k)\mathbf{x}_{k-\Delta k}$ and a state-to-measurement transition matrix $\mathbf{H}^i$ such that $h^i(\mathbf{x}_k) = \mathbf{H}^i\mathbf{x}_k$. If $\mathbf{x}_k^i$ and $\mathbf{P}_k^i$ are the state estimate and its associated error covariance estimate corresponding to the capturing instant k of node C^i respectively, the IF [2] represents the local posterior as the information matrix $\mathbf{Y}_k^i = {\mathbf{P}_k^i}^{-1}$ and the information vector $\mathbf{y}_k^i = {\mathbf{P}_k^i}^{-1}\mathbf{x}_k^i$. The IF at each node C^i computes the target prior ($[\bar{\mathbf{y}}_k^i\ \bar{\mathbf{Y}}_k^i]$), likelihood ($\left[\mathbf{u}_k^i\ \mathbf{U}_k^i\right]$) and thereby the local posterior corresponding to the capturing instant k through a prediction step followed by an update step. The prediction step of IF is:

$$\begin{aligned}\bar{\mathbf{Y}}_k^i &= \left[\mathbf{F}(k-T,k){\mathbf{Y}_{k-T}^i}^{-1}\mathbf{F}(k-T,k)^\top + \mathbf{Q}(k-T,k)\right]^{-1},\\ \bar{\mathbf{y}}_k^i &= \bar{\mathbf{Y}}_k^i\mathbf{F}(k-T,k){\mathbf{Y}_{k-T}^i}^{-1}\mathbf{y}_{k-T}^i.\end{aligned} \tag{2.4}$$

The update step of IF is:

$$\begin{aligned}\mathbf{y}_k^i &= \bar{\mathbf{y}}_k^i + \mathbf{u}_k^i,\\ \mathbf{Y}_k^i &= \bar{\mathbf{Y}}_k^i + \mathbf{U}_k^i,\end{aligned} \tag{2.5}$$

where $\mathbf{u}_k^i = {\mathbf{H}^i}^\top {\mathbf{R}^i}^{-1}\mathbf{z}_k^i$ and $\mathbf{U}_k^i = {\mathbf{H}^i}^\top {\mathbf{R}^i}^{-1}\mathbf{H}^i$. If the camera C^i has no measurement of the target (i.e. non-viewing camera), $\mathbf{u}_k^i = \mathbf{0}_{n\times 1}$ and $\mathbf{U}_k^i = \mathbf{0}_{n\times n}$. The prior is considered as the posterior

at k. The use of IF is advantageous compared to KF because the computation complexity of the update step in IF (2.5) is smaller than in KF.

2.1.2 Extended Information Filter

If $f(\cdot)$ or $h^i(\cdot)$ or both are non-linear, EKF [2] can be used. EKF assumes linearity at the best available state using the Taylor series approximation. EKF handles mild non-linearities where the first-order approximations of Jacobians are available. The concept of IF can be applied to EKF, resulting in EIF [60]. The prediction step of EIF is:

$$\begin{aligned}\bar{\mathbf{Y}}^i_k &= \left[\mathbf{J}_{f,\mathbf{x}}(k-T,k){\mathbf{Y}^i_{k-T}}^{-1}\mathbf{J}_{f,\mathbf{x}}(k-T,k)^\top + \mathbf{J}_{f,\mathbf{w}}(k-T,k)\mathbf{Q}(k-T,k)\mathbf{J}_{f,\mathbf{w}}(k-T,k)^\top\right]^{-1}_{,}\\ \bar{\mathbf{y}}^i_k &= \bar{\mathbf{Y}}^i_k f({\mathbf{Y}^i_{k-T}}^{-1}\mathbf{y}^i_{k-T}),\end{aligned} \tag{2.6}$$

where $\mathbf{J}_{f,\mathbf{x}}(\cdot)$ is the Jacobian of $f(\cdot)$ with respect to $\mathbf{x}^i_{k-T} = {\mathbf{Y}^i_{k-T}}^{-1}\mathbf{y}^i_{k-T}$ and $\mathbf{J}_{f,\mathbf{w}}(\cdot)$ is the Jacobian of $f(\cdot)$ with respect to $\mathbf{w}_{k-T}$. The update step of EIF is:

$$\begin{aligned}\mathbf{y}^i_k &= \bar{\mathbf{y}}^i_k + \mathbf{u}^i_k,\\ \mathbf{Y}^i_k &= \bar{\mathbf{Y}}^i_k + \mathbf{U}^i_k,\end{aligned} \tag{2.7}$$

where, $\mathbf{u}^i_k$ and $\mathbf{U}^i_k$ are the information contribution terms (represent likelihood) defined as:

$$\begin{aligned}\mathbf{u}^i_k &= {\mathbf{J}^i_{h,\mathbf{x}}}^\top {\mathbf{R}^i}^{-1}[\mathbf{z}^i_k - h^i(\bar{\mathbf{x}}^i_k) + \mathbf{J}^i_{h,\mathbf{x}}\bar{\mathbf{x}}^i_k],\\ \mathbf{U}^i_k &= {\mathbf{J}^i_{h,\mathbf{x}}}^\top {\mathbf{R}^i}^{-1}\mathbf{J}^i_{h,\mathbf{x}}.\end{aligned} \tag{2.8}$$

Here, $\bar{\mathbf{x}}^i_k = {\bar{\mathbf{Y}}^i_k}^{-1}\bar{\mathbf{y}}^i_k$ and $\mathbf{J}^i_{h,\mathbf{x}}$ is the Jacobian of $h^i(\cdot)$ with respect to $\bar{\mathbf{x}}^i_k$.

The Homography matrix-based state to measurement transition function $h^i(\cdot)$ of camera C^i is as follows:

$$h^i(\mathbf{x}_k) = \begin{bmatrix} \frac{H^i(1,1)x_k + H^i(1,2)y_k + H^i(1,3)}{H^i(3,1)x_k + H^i(3,2)y_k + H^i(3,3)} \\ \frac{H^i(2,1)x_k + H^i(2,2)y_k + H^i(2,3)}{H^i(3,1)x_k + H^i(3,2)y_k + H^i(3,3)} \end{bmatrix} = \begin{bmatrix} \frac{x_p}{z_p} \\ \frac{y_p}{z_p} \end{bmatrix}. \tag{2.9}$$

The values $H^i(1,1),...,H^i(3,3)$ are the elements of the homography matrix H^i that maps the ground plane with the image plane of C^i. The Jacobian of the homography-based measurement

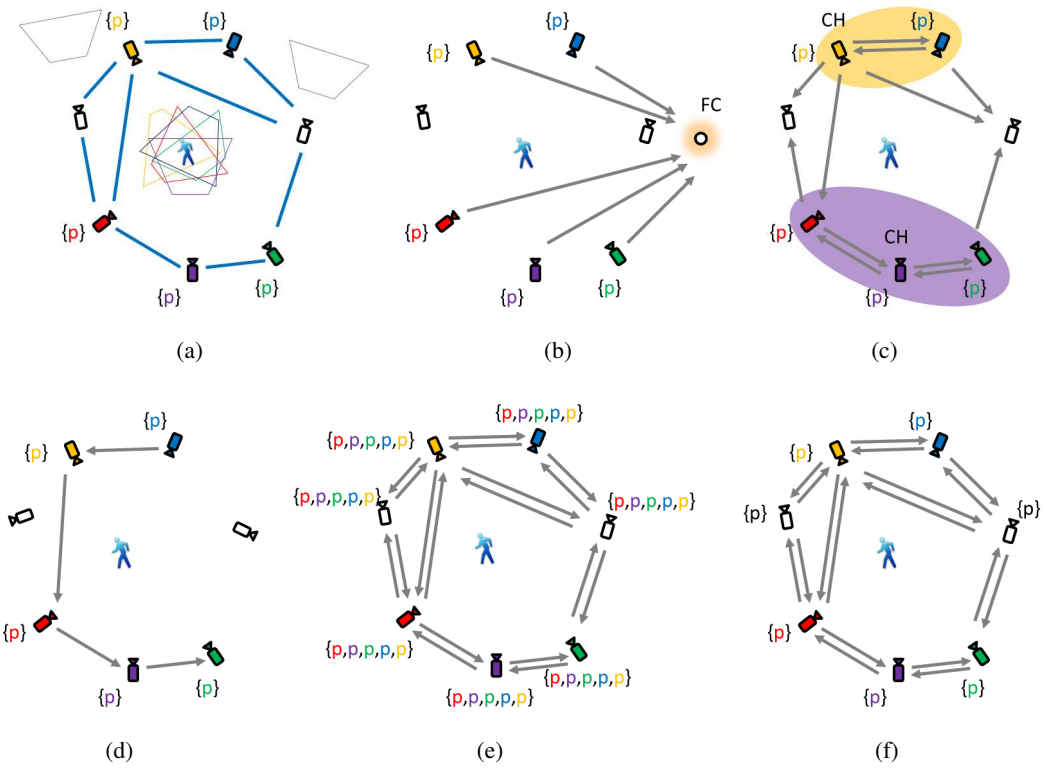


Figure 2.1: Information flow in fusion schemes. (a) Connectivity and field of views of given wireless camera network. (b) Centralised. (c) Dynamic clustering. (d) Token passing. (e) Flooding. (f) Consensus. Key. FC: Fusion centre. CH: Cluster head. p: local posterior. Cameras viewing the target are shown in colours. Blue lines: Communication links. Grey arrows: Information flow.

model w.r.t $\bar{\mathbf{x}}_k^i$ is as follows:

$$\mathbf{J}_{h,\mathbf{x}}^i = \frac{dh^i(\mathbf{x}_k)}{d\mathbf{x}}\bigg|_{\mathbf{x}_k=\bar{\mathbf{x}}_k^i} = \left[\begin{array}{cccc} \frac{H^i(1,1)(z_p)-(x_p)H^i(3,1)}{z_p^2} & \frac{H^i(1,2)(z_p)-(x_p)H^i(3,2)}{z_p^2} & 0 & 0 \\ \frac{H^i(2,1)(z_p)-(y_p)H^i(3,1)}{z_p^2} & \frac{H^i(2,2)(z_p)-(y_p)H^i(3,2)}{z_p^2} & 0 & 0 \end{array}\right]\Bigg|_{\mathbf{x}_k=\bar{\mathbf{x}}_k^i}. \quad (2.10)$$

2.2 Information fusion

The cameras produce different local estimates because of different and random noise levels of the cameras. The estimates are fused to produce an estimate with a minimum possible error. Fusion schemes define when and what information to share under specific communication architectures [17]. The fusion schemes share raw data (e.g. measurements) or decisions (e.g. estimations) [86]. In the former case, measurements or features such as likelihoods are fused to obtain the global estimate. In the latter case, local estimates are fused to get the global estimate. The following sections describe how information is fused in wireless networks when using five different fusion schemes, namely centralised fusion, flooding, token passing, consensus and dynamic

clustering (Figure 2.1). The schemes assume that the network routing tables are not available.

2.2.1 Centralised information fusion

There are two types of centralised fusion, namely centralised decision fusion and centralised measurement fusion. In *centralised decision fusion* [71, 82], all viewing nodes send their local posteriors to a FC for computing the global posterior. Let $p_k^i = p(\mathbf{x}_k^i|\mathbf{z}_{1:k}^i)$ be the local posterior computed by camera C^i using a local Bayesian filter (e.g. EIF). Fusion of local posteriors is done such that the weighted sum of Kullback-Leibler (KL) divergences of fusion result $\hat{p}_k^F$ w.r.t. each pdf p_k^i, $D_{KL}(\hat{p}_k^F|p^i)$, is minimised, i.e.

$$\hat{p}_k^F = \arg\inf_p \sum_{i=1}^{N} \pi_k^i D_{KL}(p|p_k^i). \tag{2.11}$$

The superscript F is used to indicate that the result is available at the FC. KL-divergence of a pdf p w.r.t. a pdf q, $D_{KL}(p|q)$, indicates the loss of information when q is approximated as p. π_k^i in (2.11) indicates the weight given to pdf p_k^i such that

$$0 \leq \pi_k^i \leq 1, \ \sum_{i=1}^{N} \pi_k^i = 1. \tag{2.12}$$

The weighted geometric mean of given pdfs satisfies (2.11) [6], i.e.

$$p(\hat{\mathbf{x}}_k^F|\mathbf{Z}_{1:k}) = \frac{\prod_{i=1}^{N} \left[p(\mathbf{x}_k^i|\mathbf{z}_{1:k}^i)\right]^{\pi_k^i}}{\int \prod_{i=1}^{N} \left[p(\mathbf{x}_k^i|\mathbf{z}_{1:k}^i)\right]^{\pi_k^i} d\mathbf{x}_k^i}. \tag{2.13}$$

$\hat{p}_k^F$ is called the Kullback-Leibler Average (KLA). If all the pdfs are given the same priority, i.e. $\pi_k^i = \frac{1}{N}, \forall C^i$, (2.13) provides the *unweighted KLA* [6], otherwise, it provides the *weighted KLA* [41]. Covariance intersection algorithms [31, 64] compute weighted KLA by selecting the weights for the Gaussian pdfs such that the uncertainty in the resulting pdf is minimised. The weights are computed based on the traces [64] or the determinants [31] of the covariances of the pdfs.

The information form of the centralised decision fusion is as follows. All viewing nodes or all the cameras send their local posteriors ($\mathbf{y}_k^i$ and $\mathbf{Y}_k^i$) to a FC for computing the global posterior ($\hat{\mathbf{y}}_k^F$ and $\hat{\mathbf{Y}}_k^F$) [71, 82]. The weighted KLA of N Gaussian pdfs $p(\mathbf{x}_k^i|\mathbf{z}_k^i) = \mathcal{N}(\mathbf{x}_k : {\mathbf{Y}_k^i}^{-1}\mathbf{y}_k^i, {\mathbf{Y}_k^i}^{-1}), \forall C^i$

with weights π_k^i satisfying (2.12) is another Gaussian pdf $p(\hat{\mathbf{x}}_k^F|\mathbf{Z}_{1:k}) = \mathcal{N}(\mathbf{x}_k : \hat{\mathbf{Y}}_k^{F^{-1}}\hat{\mathbf{y}}_k^F, \hat{\mathbf{Y}}_k^{F^{-1}})$ [6, 7], where

$$\begin{aligned} \hat{\mathbf{Y}}_k^F &= \sum_{i=1}^{N} \pi_k^i \mathbf{Y}_k^i, \\ \hat{\mathbf{y}}_k^F &= \sum_{i=1}^{N} \pi_k^i \mathbf{y}_k^i. \end{aligned} \tag{2.14}$$

The weights, π_k^i, used during the fusion of posteriors vary depending on the fusion algorithm. Improved Fast Covariance Intersection (IFCI) [31] computes the weights based on the determinants of the information matrices and produces better fusion estimates than Fast Covariance Intersection [64]. The global state estimate and the corresponding error covariance can be calculated using:

$$\begin{aligned} \hat{\mathbf{x}}_k^F &= \hat{\mathbf{Y}}_k^{F^{-1}}\hat{\mathbf{y}}_k^F, \\ \hat{\mathbf{P}}_k^F &= \hat{\mathbf{Y}}_k^{F^{-1}}. \end{aligned} \tag{2.15}$$

Benighted cameras and the cameras with high measurement noise have uncertain estimates. KLA weight less the information. The proof is as follows:

$$\hat{\mathbf{x}}_k^F = \hat{\mathbf{Y}}_k^{F^{-1}}\hat{\mathbf{y}}_k^F = \left[\sum_{i=1}^{N} \pi_k^i \mathbf{Y}_k^i\right]^{-1} \left[\sum_{i=1}^{N} \pi_k^i \mathbf{y}_k^i\right] = \left[\sum_{i=1}^{N} \pi_k^i \mathbf{P}_k^{i\,-1}\right]^{-1} \left[\sum_{i=1}^{N} \pi_k^i \mathbf{P}_k^{i\,-1}\mathbf{x}_k^i\right]. \tag{2.16}$$

As each local estimate $\mathbf{x}_k^i$ is weighted based on its uncertainty $\mathbf{P}_k^i$, weighted or unweighted decision fusion is robust to p-benightedness. A limitation of unweighted KLA is that it does not consider c-benightedness whereas weighted KLA does. Another limitation of decision fusion is that it considers the same prior multiple times during fusion. The likelihood is null for benighted nodes so higher the number of benighted nodes higher the influence of priors on the decision fusion result. As priors of all node are the same (because they are computed from the same previous global estimate), it is unnecessary to consider them again and again in the fusion. Due to these redundant priors, the results of the decision fusion are not optimal.

In *centralised measurement fusion*, all the viewing nodes send their likelihoods ($\mathbf{u}_k^i$ and $\mathbf{U}_k^i$) to a FC. The fusion step at the FC performs addition of the likelihood terms from of N nodes in

the network as:

$$\begin{aligned} \hat{\mathbf{y}}_k^F &= \bar{\mathbf{y}}_k^F + \sum_{i=1}^{N} \mathbf{u}_k^i, \\ \hat{\mathbf{Y}}_k^F &= \bar{\mathbf{Y}}_k^F + \sum_{i=1}^{N} \mathbf{U}_k^i. \end{aligned} \tag{2.17}$$

As the prior is considered only once, the problem of redundant priors does not arise in measurement fusion. Moreover, this handles also c- and p-benightedness.

The next section will introduce the corresponding distributed versions. The distributed fusion schemes aim to produce either the centralised decision fusion result given in (2.14) or the centralised measurement fusion result given in (2.17).

2.2.2 Distributed information fusion

In distributed fusion, each node exchanges its information with its communicative neighbours so the connectivity of the networks plays a key role.

In *flooding* (or dissemination) [39, 84] all the viewing nodes broadcast their local posterior to all or to a subset of nodes (e.g. viewing nodes) in the network. If the network is fully connected information is distributed in a single iteration of information exchange [53]. Some authors refer this case as *non-centralised* fusion because each node acts as the FC in centralised case but is not actually the centralised way. In the case of limited connectivity, flooding requires multiple iterations of communications. In each iteration, each node sends its own and the previously received information to its neighbours. Eventually, all participating nodes have the same set of posteriors [22, 23]. See Figure 2.1(e). Then, each participating node performs fusion, updates its local posterior. For large networks with limited connectivity, flooding has high communication cost, high processing cost and high memory requirements [69, C4]. This scheme is therefore suitable for sharing low amounts of information when high connectivity exists among the nodes.

Token passing [37, 38, 70] is a sequential estimator in which viewing nodes form an aggregation chain (AC). Each node in the AC receives a partial posterior from the previous one, updates this posterior using its local posterior and sends the result to the next node. The process finishes when all AC nodes are visited once. The most informative node (decided based on the local posterior and the global knowledge of the network) is selected as the *next* node [39]. The *last* AC node provides the global posterior at the current time step. See Figure 2.1(d). Then, this node initiates the AC for the next time step (often also becoming the *first* AC node). The sequential estimation and the transmission of high dimensional estimations such as PF posteriors

cause latency [39]. Nastasi and Cavallaro [62] applied such a fusion scheme to smart camera networks using distributed PFs assuming that viewing nodes can communicate with each other. The scheme is suitable when cameras with overlapping FoVs are connected (i.e. fully connected viewing nodes) or routing tables are provided.

In *dynamic clustering*, the viewing nodes negotiate locally and form clusters where a node is selected as cluster head. The node generates the global posterior by fusing its own and received posterior from the other cluster members. Static clusters can be created to track targets based on their overlapping sensing regions [34] and might use more nodes per cluster than needed. In order to cluster only the viewing nodes, dynamic clustering adapts the cluster membership depending on the target location and the network topology. Medeiros *et al.* [56] proposed a dynamic clustering technique to create multiple single-hop clusters of viewing nodes. As the target moves, the clusters modify their members so that only viewing nodes are members of a cluster. See Figure 2.1(c). EKF is used as the underlying filter for estimating the target state to handle non-linearity in the measurement model. Here, inter-cluster data fusion is not supported. Each cluster head sends the fused estimates to a base station where the final fusion step is performed similarly to the centralised fusion. The formation and maintenance of clusters add communication and computation overhead in the network [C4]. Iterative message exchange (or negotiation) is required to select the cluster head and to propagate the cluster membership over time. In this case cluster formation is a distributed process and fusion is a decentralised process. Cluster formation and cluster-head selection add computation and communication costs and increase latency.

Reaching *consensus* means that all nodes have the same value(s) for the considered variable(s) such as the target state [66, 84]. Consensus algorithms are distributed protocols that aim at reaching an agreement on a decision variable (e.g. target state) using iterative peer-to-peer communication among the nodes. Examples include agreement on average of local variables and maximum (or minimum) of local variables. Average consensus aims at all nodes to converge to the average of the local variables. Maximum (or minimum) consensus aims at all nodes to converge to the maximum (or minimum) of the local variables. Each iteration of a consensus protocol consists of two main steps, namely information exchange with communicative neighbours and consensus update. Depending on the desired agreement type, the protocol defines what to exchange and how to update. Each node broadcasts its information (e.g. local state estimate) to its neighbours. Each receiving node updates its information based on its own information and the re-

ceived information. The updated information is again broadcasted to its neighbours. The process of information exchange and update (consensus update) are done until the network reaches the convergence on the target state. In each iteration, nodes exchange information with neighbours and perform fusion using the average [6], gossip [5,12], maximum or minimum [30] approaches. See Figure 2.1(f). Consensus schemes operate at two time scales: collecting measurements and performing iterations between consecutive measurement collections [69]. The advantages of this scheme are robustness to node failures (similar to other distributed approaches) and the availability of global posterior at all nodes. Moreover, this scheme does not require routing protocols or knowledge about nodes (e.g. observation models or FOVs), the network (e.g. communication graph), thus coping with topology changes and link failures. The communication and computation costs are high as all nodes (viewing and not viewing) exchange their local estimates and perform fusion [C4].

Multiple FCs exist in all schemes except centralised fusion. All nodes operate similarly in consensus, whereas only viewing nodes operate in flooding, token passing and dynamic clustering. Token passing and dynamic clustering require negotiation among nodes (prior to fusion) to decide whom to pass the token to and to propose cluster-head candidates, respectively. If there is no direct communication among the viewing nodes, dynamic clustering forms multiple single-hop clusters, flooding requires several iterations, and token passing needs routing tables for multi-hop communication. In the case of limited connectivity, consensus is the only way to perform information fusion without requiring knowledge of the network so this thesis focuses its research on consensus approaches to solve the above mentioned challenges. The following section presents the state of the art on consensus-based tracking assuming synchronous captures.

2.3 Consensus-based tracking

2.3.1 P-benightedness

The distributed Kalman Consensus Filter (KCF) [66] computes local estimates ($\mathbf{x}_k^i$) via KFs. KCF performs the averaging of the local state estimates by weighting the viewing and the benighted nodes equally so KCF [21,73] does not handle p-, c-benightedness and redundancy. The Generalised KCF (GKCF) [48] and the ICF [6] deal with the p-benightedness by weighting the state estimates based on their uncertainty so they achieve higher estimation accuracy than KCF. ICF makes the network with limited connectivity converge to the unweighted KLA of the local esti-

mates [6, C5]. ICF performs the A-consensus on the local state information $(\mathbf{y}^i_k, \mathbf{Y}^i_k)$. The initial consensus terms are $\mathbf{y}^i_{k(0)} = \mathbf{y}^i_k$ and $\mathbf{Y}^i_{k(0)} = \mathbf{Y}^i_k$. The subscript $k(l)$ indicates that the information corresponds to k before starting the consensus iteration at $k+l$. Each node C^i that is running ICF exchanges its posterior $(\mathbf{y}^i_{k(l)}, \mathbf{Y}^i_{k(l)})$ with communicative neighbours where benighted nodes send either null posterior $(\mathbf{0}, \mathbf{0})$ (if c-benighted) or predicted posterior [50] (if p-benighted) as information. Each node C^i executes a consensus step as:

$$\mathbf{y}^i_{k(l+\gamma)} = \mathbf{y}^i_{k(l)} + c \sum_{C^j \in \mathcal{N}^i} (\mathbf{y}^j_{k(l)} - \mathbf{y}^i_{k(l)}), \tag{2.18}$$

where γ is the periodicity of consensus iterations. The same process is applied to $\mathbf{Y}^i_{k(l)}$. The values c can be set to guarantee the convergence to the average of the initial estimates of all nodes after L iterations [65]. The weights (c) used in the consensus update affect the convergence rate [85]. The speed of convergence to the posterior average depends on the connectivity of the network. The number of edges connected to a node in a communication graph is called the degree of the node. Maximum-degree weights and metropolis weights are two types of weights used to achieve average consensus [84]. Metropolis weight selection requires the knowledge of the communication graph whereas the maximum-degree weight selection do not need any global knowledge of the communication graph. Let Δ_{max} be the maximum degree of instantaneous communication graph, selecting $c \in \left(0, \frac{1}{\Delta_{max}}\right)$ as the weight, guarantees the convergence [65]. If Δ_{max} is not available, the weight $c = \frac{1}{\hat{N}-1}$ guarantees convergence to the average, where $\hat{N}$ is the upper bound of the number of nodes in the network. This is because $0 < \frac{1}{\hat{N}-1} < \frac{1}{\Delta_{max}}$. The larger the value of c the faster the convergence. However, if the value is equal or more than Δ_{max}, the average consensus algorithm becomes unstable. The constraint of the convergence to the average is that the underlying communication graph must be connected and the collection of infinitely occurring communication graphs must be jointly connected [84]. Asymptotically, the nodes converge to their average as

$$\begin{aligned} \hat{\mathbf{y}}^i_k &= \lim_{l\to\infty} \mathbf{y}^i_{k(l)} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{y}^i_k, \\ \hat{\mathbf{Y}}^i_k &= \lim_{l\to\infty} \mathbf{Y}^i_{k(l)} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{Y}^i_k, \end{aligned} \tag{2.19}$$

which is the unweighted KLA as in (2.14) [71]. If the number of c-benighted nodes (cameras having no knowledge of the target information) is B, the consensus result should be

$$\begin{aligned}\hat{\mathbf{y}}_k^i &= \frac{1}{N-B}\sum_{i=1}^{N}\mathbf{y}_k^i,\\ \hat{\mathbf{Y}}_k^i &= \frac{1}{N-B}\sum_{i=1}^{N}\mathbf{Y}_k^i.\end{aligned} \tag{2.20}$$

Hence, ICF handles p-benightedness but not c-benightedness. The communication cost of ICF is higher than KCF because in each consensus iteration, KCF exchanges only the state estimate whereas ICF exchanges also the corresponding error covariance encoded in the information matrix.

2.3.2 C-benightedness

ICI converges to the weighted KLA of the initial estimates and outperforms ICF in estimation accuracy [41]. Such weighted averaging results in optimal fusion estimates when the nodes are uncorrelated [6]. ICI achieves consensus using the initial terms $\mathbf{y}_{k(0)}^i = \mathbf{y}_k^i$ and $\mathbf{Y}_{k(0)}^i = \mathbf{Y}_k^i$. Each node C^i executes the consensus step as:

$$\mathbf{y}_{k(l+\gamma)}^i = \sum_{C^j \in \{C^i \bigcup \mathcal{N}^i\}} w_k^{i,j}\mathbf{y}_{k(l)}^j, \tag{2.21}$$

where $w_k^{i,j}$ is the weight given to C^j by C^i based on the covariance information of all neighbours. The traces [64] or determinants [31] of the information matrices are considered. $w_k^{i,j} = 0$ if C^j is a c-benighted node. The c-benighted nodes get zero weight so they do not affect the fusion. The same process is applied to $\mathbf{Y}_{k(l)}^i$. The advantage of ICI is that it does not require the knowledge of the maximum degree Δ_{max} of the underlying communication graph whereas A-consensus approaches such as KCF, ICF and GKCF do. ICI guarantees convergence but not to the centralised weighted KLA as in (2.14).

The Batch Covariance Intersection (BCI) [75] converges to the centralised weighted KLA by using A-consensus approach in ICI framework. BCI works similarly to bridge consensus. Bridge consensus [16] performs distributed scalar averaging in the presence of c-benighted nodes. The weight is zero for the c-benighted nodes and one for the non benighted nodes. It achieves consensus on the average of the values and average of the weights in parallel. The ratio between the average of values and the average of weight produces the average of the non-benighted nodes val-

ues. BCI applies the same concept to local posteriors, considers their uncertainty in the weights and performs the A-consensus (similar to (2.18)) on the initial terms $\mathbf{y}^i_{k(0)} = w^i_k \mathbf{y}^i_k$, $\mathbf{Y}^i_{k(0)} = w^i_k \mathbf{Y}^i_k$ and $v^i_{k(0)} = w^i_k$. Asymptotically, they converge to

$$\begin{aligned} \lim_{l\to\infty} \mathbf{y}^i_{k(l)} &= \frac{1}{N}\sum_{i=1}^{N} w^i_k \mathbf{y}^i_k, \\ \lim_{l\to\infty} \mathbf{Y}^i_{k(l)} &= \frac{1}{N}\sum_{i=1}^{N} w^i_k \mathbf{Y}^i_k, \\ \lim_{l\to\infty} v^i_{k(l)} &= \frac{1}{N}\sum_{i=1}^{N} w^i_k. \end{aligned} \tag{2.22}$$

BCI computes the weighted KLA estimate in (2.14) as:

$$\begin{aligned} \hat{\mathbf{y}}^i_k &= \left[\lim_{l\to\infty} \mathbf{y}^i_{k(l)}\right]\left[\lim_{l\to\infty} v^i_{k(l)}\right]^{-1} = \left[\frac{1}{N}\sum_{i=1}^{N} w^i_k \mathbf{y}^i_k\right]^{-1}\left[\frac{1}{N}\sum_{i=1}^{N} w^i_k\right] = \sum_{i=1}^{N} \pi^i_k \mathbf{y}^i_k, \\ \hat{\mathbf{Y}}^i_k &= \left[\lim_{l\to\infty} \mathbf{Y}^i_{k(l)}\right]\left[\lim_{l\to\infty} v^i_{k(l)}\right]^{-1} = \left[\frac{1}{N}\sum_{i=1}^{N} w^i_k \mathbf{Y}^i_k\right]^{-1}\left[\frac{1}{N}\sum_{i=1}^{N} w^i_k\right] = \sum_{i=1}^{N} \pi^i_k \mathbf{Y}^i_k. \end{aligned} \tag{2.23}$$

As the weights w^i_k are selected based on the trace or determinant of the information matrix, $w^i_k = 0$ if C^i is a c-benighted node. The c-benighted nodes get zero weight so they do not affect the fusion. The consensus estimate in (2.23) is independent of the number of nodes and benighted nodes and hence BCI is not affected by the presence of c-benighted nodes. Due to the third consensus term, BCI has higher communication cost than ICF and ICI. Moreover, as BCI uses the A-consensus approach, it requires the knowledge of Δ_{max}. ICI and BCI handle p- and c-benightedness but not redundant priors that cause cross-correlation among the nodes [50].

2.3.3 Redundant priors

The Information-Weighted Consensus Filter (IWCF) extends the ICF and handles also the redundancy via proper relative weighting of the priors and likelihoods before initiating the consensus iterations [8, 19, 43, 50, C5]. The weight depends on the number of nodes because the number of redundant priors is equal to the number of nodes. The consensus terms of IWCF are initialised as

$$\begin{aligned} \mathbf{y}^i_{k(0)} &= N\left(\frac{\bar{\mathbf{y}}^i_k}{N} + \mathbf{u}^i_k\right) = \bar{\mathbf{y}}^i_k + N\mathbf{u}^i_k, \\ \mathbf{Y}^i_{k(0)} &= N\left(\frac{\bar{\mathbf{Y}}^i_k}{N} + \mathbf{U}^i_k\right) = \bar{\mathbf{Y}}^i_k + N\mathbf{U}^i_k, \end{aligned} \tag{2.24}$$

so that the average of the states is equivalent to the centralised estimate given in (2.17). The value N in (2.24) indicates the number of redundant priors in the network. The average consensus is performed (similar to (2.18)) on $\mathbf{y}^i_{k(0)}$ and $\mathbf{Y}^i_{k(0)}$. Asymptotically, all nodes' estimates converge to their average estimate as

$$\begin{aligned}\hat{\mathbf{y}}^i_k &= \lim_{l\to\infty} \mathbf{y}^i_{k(l)} = \frac{1}{N}\sum_{i=1}^{N} \bar{\mathbf{y}}^i_k + \sum_{i=1}^{N} \mathbf{u}^i_k, \\ \hat{\mathbf{Y}}^i_k &= \lim_{l\to\infty} \mathbf{Y}^i_{k(l)} = \frac{1}{N}\sum_{i=1}^{N} \bar{\mathbf{Y}}^i_k + \sum_{i=1}^{N} \mathbf{U}^i_k.\end{aligned} \tag{2.25}$$

C-benighted nodes neither have the priors (i.e. $\bar{\mathbf{y}}^i_k = \mathbf{0}_{n\times 1}$ and $\bar{\mathbf{Y}}^i_k = \mathbf{0}_{n\times n}$) nor likelihoods (i.e. $\mathbf{i}^i_k = \mathbf{0}_{n\times 1}$ and $\mathbf{I}^i_k = \mathbf{0}_{n\times n}$). In the presence of B c-benighted nodes, the number of redundant priors is $N-B$, so the desired result is

$$\begin{aligned}\hat{\mathbf{y}}^i_k &= \frac{1}{N-B}\sum_{i=1}^{N} \bar{\mathbf{y}}^i_k + \sum_{i=1}^{N} \mathbf{u}^i_k, \\ \hat{\mathbf{Y}}^i_k &= \frac{1}{N-B}\sum_{i=1}^{N} \bar{\mathbf{Y}}^i_k + \sum_{i=1}^{N} \mathbf{U}^i_k.\end{aligned} \tag{2.26}$$

As $\frac{1}{N} < \frac{1}{N-B}$, IWCF underweights the priors in the presence of c-benighted nodes, which is a drawback of IWCF.

Similarly to the IWCF, the Hybrid Consensus on Measurements - Consensus on Information (HCMCI) [8] handles p-benightedness and redundancy of the priors. HCMCI achieves consensus on the prior and the likelihood in parallel and appropriately weighs the consensus prior and the consensus likelihood. HCMCI is equivalent to IWCF but requires double the cost of communication because of two parallel consensus algorithms. IWCF and HCMCI achieve better estimation accuracy than KCF, GKCF, ICF and ICI. ICF, IWCF and HCMCI do not handle c-benightedness. The Distributed Hybrid Information Fusion (DHIF) [79] is robust to p- and c-benightedness, and redundancy. DHIF exchanges priors and likelihoods in parallel (similarly to HCMCI). The weights used for fusion of priors are chosen based on the relative uncertainties (similarly to ICI) so c-benighted nodes do not affect the result. The likelihoods are fused using a summation. The method uses a single iteration so there are no redundant priors. On the other hand, the asymptotic properties of consensus are lost.

2.3.4 Non-linearity

All the above mentioned algorithms use KF or IF for local state estimation. EKCF [21] uses the EKF for local state estimation. Similar to KCF, EKCF does not handle benightedness and redundancy. The EICF and EIWCF are similar to ICF and IWCF respectively but they use EIF for local state estimation to handle non-linearity in the measurement model [C5]. There exist also consensus algorithms that use PF [69] for non-linear and non-Gaussian systems.

2.3.5 Asynchronous captures

All the above consensus-filters work only in synchronous settings. There exist average consensus-based methods that work with asynchronous communications [10, 13, 14, 29, 57, 89] and dynamic changes in the consensus variable [51]. However, they cannot be applied to distributed asynchronous tracking as they do not consider the continuously changing dynamics of the consensus variable (the target state in this case). aCDTT is a maximum consensus-based approach that makes the network converge to the most certain state among the local and received estimates [33]. If multiple cameras produce different estimates having the same uncertainty, a tie-break rule is required to select the consensus state. To handle the tie-break in the distributed framework, nodes have to exchange the source node ID of each state along with the state information. Because aCDTT does not fuse local information of the cameras, it does not reduce the uncertainty on the state. Moreover, the nodes may update their local estimates with the selected estimate that might correspond to a different time instant. aCDTT guarantees convergence only if the number of consensus iterations is at least the diametre of the network. The diametre of a network is defined as the shortest distance (in hops) between the most distant nodes. However, there is no mechanism that specifies how all the nodes agree that a sufficient number of consensus iterations are performed. The advantage of average consensus over maximum consensus is that in average consensus the nodes fuse the information instead of selection and thereby reducing the uncertainty of the local states.

2.4 Distributed asynchronous tracking

This section presents the state of the art on asynchronous tracking in fully connected networks. Distributed asynchronous tracking can be performed using sequential [92] or batch methods [9]. These sequential and batch methods assume full connectivity among all the nodes [9, 33, 92] or

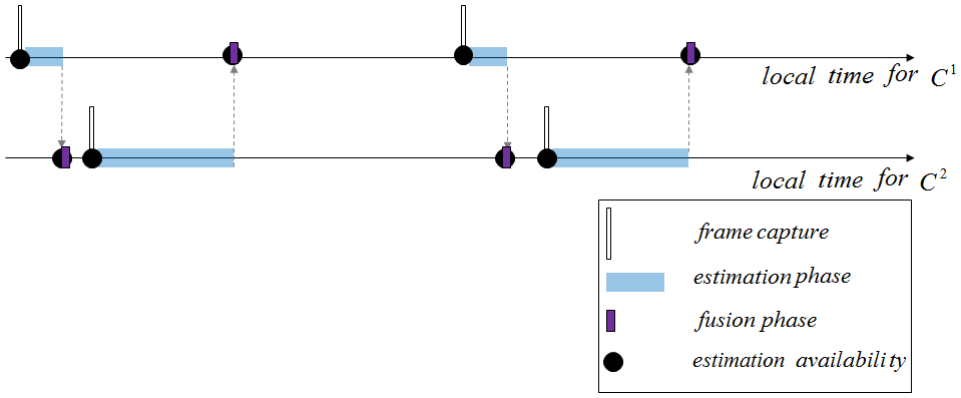


Figure 2.2: The timeline of the sequential fusion. KEY – C^i: camera i, the grey arrows indicate the communication.

at least among the nodes that observe the target simultaneously [40] and perform non-centralised fusion. However, WCNs do not guarantee such full connectivity so the methods cannot always be applied always.

2.4.1 Sequential methods

In *sequential methods*, the nodes estimate the target state not only when they have measurements but also when they receive target information from other nodes (see Figure 2.2). As nodes estimate the target location corresponding to their capturing instants or the instants of reception from other nodes, they might not correct their estimates corresponding to their capturing instants using the information received from other nodes. In the sequential methods, each node broadcasts its measurement to all other nodes in the network [9, 92] or only to its neighbours [40]. Each node estimates the target state corresponding to its own sampling instant as well as at every instant in which the measurements of other nodes are received. In the Sequential Asynchronous Kalman Filter (SAF) [92], the nodes obtain the measurements and exchange the information with all other nodes. When a node receives a measurement (own or from another node), it computes the estimate corresponding to the reception instant using the received measurement. The limitations of SAF are: SAF does not consider the processing delays. SAF considers the reception instant of the information as the capturing instant of the sender and estimates the target states corresponding to the reception instants. As the processing delay is significant in camera networks, SAF does not perform well in camera networks, i.e. the accuracy of the estimates corresponding to the reception instants decreases if the processing delays are not considered into account during the estimation. This is because the received estimates correspond to a different instant (capturing

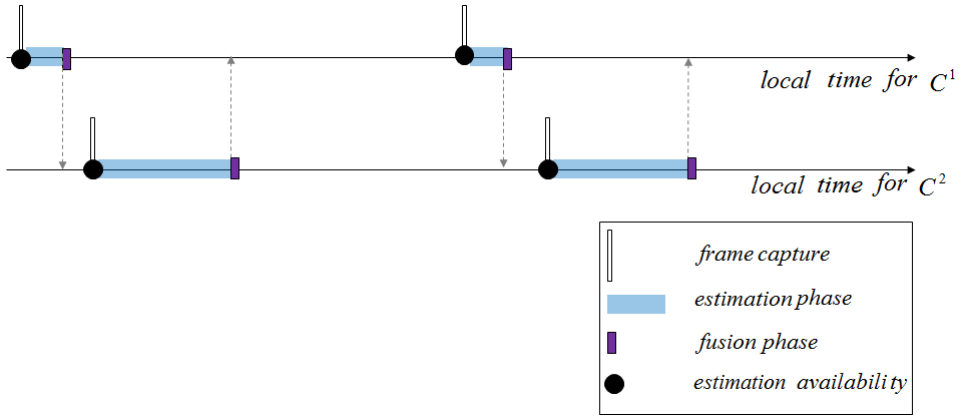


Figure 2.3: The timeline of the batch fusion. KEY – C^i: camera i, the grey arrows indicate the communication.

instant of the senders) than the reception instant. The estimation at the receiver takes place at the reception instants and results in less accurate estimates. The SAF is extended to model and estimate the delays (SAF-ED) [40]. Distributed Asynchronous Particle Filter (DAPF) [40] is another sequential method. Unlike SAF, SAF-ED and DAPF consider the processing delays. The nodes exchange the local state information. When a node receives state information from another node, it estimates the delay and thereby the sampling instant corresponding to the received information. If the sampling instant of the received information is older (earlier) than its local sampling instant, the node predicts the received information corresponding to the local sampling instant. If the local sampling instant is older than the sampling instant of the received information, the local information corresponding to the sampling instant of the received information is predicted. Finally, the temporally aligned information is fused. The limitation of SAF, SAF-ED and DAPF is that the fusion result corresponds to the sampling instant of the latest senders so only the global posterior corresponding to the latest sampling instant is available. The nodes do not correct the local estimates corresponding to their capturing instants.

2.4.2 Batch methods

In *batch methods*, the nodes estimate the target state only when they have measurements and during the estimation, they consider also the information received from other nodes (see Figure 2.3). In batch methods, each node broadcasts its measurement to all other nodes in the network. Each node estimates the target state corresponding only to its own sampling instant using its own measurement and the most recently received measurements from other nodes [9]. Batch methods have less processing load compared to the sequential methods because of the less number of esti-

mations [32]. However, there are no batch methods that consider processing delays in fusion till now. Consensus-based methods such as aCDTT also update the estimate corresponding to the capturing instant so this thesis considers also them as batch methods.

Except in DAPF and aCDTT, in all other sequential and batch methods, nodes exchange measurements. In the case of camera networks, the measurement model is different for each camera. Moreover, the measurement noise might be varying and depends on the target-to-camera distance. Hence, these methods require transmission of calibration information (measurement model and the instantaneous measurement noise model) along with the measurements. Even if the models are static, it is necessary that all the cameras must have the knowledge of the models to avoid transmission but it limits the scalability of the network. To save the communication cost and to provide scalability, some multi-sensor fusion architectures [42] prefer the exchange of local posteriors. DAPF is one among them.

2.5 Discussion

Distributed tracking algorithms aim to achieve the same tracking accuracy as their corresponding centralised tracking algorithms. In both distributed and centralised fusion algorithms, the local state estimates should be weighted based on their uncertainty. Any Bayesian filter uses the previous knowledge of the target state (predicted posterior). Ideally, the previous knowledge is the same at a capturing instant irrespective of the number of cameras so the previous knowledge should be considered only once per capturing instant though multiple measurements are available from multiple cameras. This redundancy is handled by appropriately weighting the previous knowledge and the new measurements. There are many works that perform distributed fusion of probability density functions in WSNs and WCNs. The sets of challenges handled by each of them are different. Table 2.1 summarises the state of the art. Distributed tracking in WCN must handle non-linear camera measurement model and processing delays that result in asynchronous captures. The remaining challenges are common for all types of WSNs. In the case of limited connectivity, consensus is the only way to perform information fusion without requiring knowledge of the network so consensus approaches are required to solve the above mentioned challenges. The maximum consensus-based approaches do not perform fusion whereas A-consensus or ICI methods perform information fusion. A major drawback of consensus is that the entire network participates in consensus irrespective of the number of cameras in the network. In the

Table 2.1: Challenges handled by state of the art methods for distributed fusion of probability density functions

Reference	limited connectivity	non-linearity	benightedness	redundant information	asynchronous captures
[73] (Song)	✓				
[39] (Hlinka)		✓			
[62] (Nastasi)		✓			
[21] (Ding)	✓	✓			
[11] (Bhuvana)	✓	✓			
[56] (Medeiros)	✓	✓			
[48] (Kamal)	✓		✓		
[6] (Battistelli)	✓	✓	✓		
[41] (Hlinka)	✓		✓		
[75] (Sun)	✓		✓		
[81] (Wang)	✓		✓		
[49, 50] (Kamal)	✓		✓	✓	
[79] (Wang)	✓		✓	✓	
[47] (Kamal)	✓	✓	✓	✓	
[54] (Liu)	✓	✓	✓	✓	
[8] (Battistelli)	✓	✓	✓	✓	
[7] (Battistelli)	✓	✓	✓		
[92] (Zhu)					✓
[33] (Giannini)	✓				✓
[40] (Hlinka)		✓	✓		✓
[9] (Beaudeau)					✓
Chapter 3 ([C5])	✓	✓	✓	✓	
Chapter 4 ([C3])	✓	✓	✓		
Chapter 5 ([C2])	✓		✓		✓
Chapter 6 ([C1])			✓		✓

case of very large WCN, where a small set of cameras can view the target at a time, consensus results in spending lot of energy for computation and communication. It is desirable to limit the number of cameras participating in consensus. Moreover, there are no consensus-based fusion methods that perform asynchronous tracking till now. In the sequential asynchronous tracking methods, the nodes do not update their local estimates corresponding to the capturing instants using the received information. In contrast, in the batch methods, the nodes collect all the information from their neighbours and fuse it to update the local estimates corresponding to the capturing instants. Different nodes have different target measurements. Different nodes might have also different priors so exchange of local state information (posterior) rather than measurement information (likelihood) is desirable to consider the differences in priors along with the

measurements. However, there are no batch methods that perform asynchronous tracking by exchanging the local state information (posteriors).

Chapter 3

Distributed Extended Information Filter[1]

This chapter proposes two new consensus-based distributed tracking algorithms for camera networks namely, the Extended Information Consensus Filter (EICF) and the Extended Information Weighted Consensus Filter (EIWCF) to handle the four challenges, namely non-linearity, limited connectivity, benightedness and redundancy. To handle non-linearity, each camera running the distributed filters use EIF to compute the target pdf. To handle limited network connectivity, the distributed filters use A-consensus approach. To handle benightedness, they compute KLA of the local pdfs using A-consensus. KLA of the local pdfs is a pdf which is a result of weighted fusion that gives less weight to the benighted nodes than the viewing nodes. As the number of redundant priors is equal to the number of nodes in the network, EIWCF weights the priors such that only one prior is considered in the fusion result. The following sections present in detail the two new filters EICF and EIWCF.

3.1 Extended information consensus

This chapter applies weighted averaging to EIF and propose two distributed filters for tracking targets in WCNs, EICF1 and EICF2, which compute the local information, $\mathbf{y}_k^i$ and $\mathbf{Y}_k^i$, differently.

EICF1 runs at each node C^i and computes the local information values, $\mathbf{y}_k^i$ and $\mathbf{Y}_k^i$, based on

[1]This chapter is completely taken from [C5].

their own respective measurement information, $\mathbf{u}_k^i$ and $\mathbf{U}_k^i$:

$$\begin{aligned} \mathbf{y}_k^i &= \bar{\mathbf{y}}_k^i + \mathbf{u}_k^i, \\ \mathbf{Y}_k^i &= \bar{\mathbf{Y}}_k^i + \mathbf{U}_k^i, \end{aligned} \tag{3.1}$$

and then exchange the values $\mathbf{y}_k^i$ and $\mathbf{Y}_k^i$ with neighbours to achieve average consensus.

EICF2 computes local information values, $\mathbf{y}_k^i$ and $\mathbf{Y}_k^i$, based on its own measurement information and also that of neighbouring nodes:

$$\begin{aligned} \mathbf{y}_k^i &= \bar{\mathbf{y}}_k^i + \sum_{C^j \in \{C^i \cup \mathcal{N}^i\}} \mathbf{u}_k^j, \\ \mathbf{Y}_k^i &= \bar{\mathbf{Y}}_k^i + \sum_{C^j \in \{C^i \cup \mathcal{N}^i\}} \mathbf{U}_k^j. \end{aligned} \tag{3.2}$$

EICF2 reaches convergence faster than EICF1, at the cost of additional communication to send the measurement information terms. Hence, EICF2 is recommended when sufficient communication resources are available.

The iterative information exchange between neighbours results in redundancy which causes correlation among the nodes' estimates. Hence, the EICF results are sub-optimal because of such correlation among the individual node estimates. In the update step of a filter (see (2.7)), the two terms involved are the priors, $\bar{\mathbf{y}}_k^i$ and $\bar{\mathbf{Y}}_k^i$, and the measurement information about the target, $\mathbf{u}_k^i$ and $\mathbf{U}_k^i$. The prior information is the result of the prediction on previous estimates, $\mathbf{y}_k^i$ and $\mathbf{Y}_k^i$, which are computed after consensus. Hence, the redundancy always lies in the prior information terms, $\bar{\mathbf{y}}_k^i$ and $\bar{\mathbf{Y}}_k^i$.

3.2 Extended information weighted consensus

Via proper weighting of prior and measurement information, IWCF mitigates the problem of redundancy [50]. By applying the concept of IWCF to EIF, This chapter proposes a non-linear distributed filter called the Extended Information Weighted Consensus Filter (EIWCF). Here the prior information is weighted by $1/N$ and the consensus proposals are prepared as:

$$\begin{aligned} \mathbf{y}_{k(0)}^i &= \frac{1}{N}\bar{\mathbf{y}}_k^i + \mathbf{u}_k^i, \\ \mathbf{Y}_{k(0)}^i &= \frac{1}{N}\bar{\mathbf{Y}}_k^i + \mathbf{U}_k^i. \end{aligned} \tag{3.3}$$

After achieving consensus on the terms (after L iterations), the results are multiplied by N:

$$\begin{aligned}\hat{\mathbf{y}}_k^i &= N\mathbf{y}_{k(L\gamma)}^i,\\ \hat{\mathbf{Y}}_k^i &= N\mathbf{Y}_{k(L\gamma)}^i.\end{aligned} \tag{3.4}$$

Here, γ is the periodicity of consensus iterations. These estimates are not affected by non-linearity, naivety and redundancy. However, EIWCF requires the knowledge of the number of nodes in the network (see (3.3) and (3.4)) whereas EICF1 and EICF2 do not. If sufficient communication resources to receive neighbours' measurement information, $\mathbf{u}_k^j$ and $\mathbf{U}_k^j$, are available, EICF2 achieves faster convergence than EICF1.

3.3 Cost analysis

3.3.1 Computation cost

The number of scalars operations performed by a node is taken as its computational cost S [4]. The computational cost, S, of EKCF is:

$$\begin{aligned}S_{EKCF} = {} & 11n^3 + 6n^2 + 4n + 17 + 3nm + 2nm^2\\ & + mn^2 + 2m + d\left(\frac{n^2}{2} + \frac{5n}{2}\right) + Ln(2+d),\end{aligned} \tag{3.5}$$

where L and d represent the number of consensus iterations involved and degree of the node, respectively; n is the size of the state vector and m is the size of the measurement vector. The computation cost of EICF1 is:

$$\begin{aligned}S_{EICF1} = {} & \frac{25n^3}{3} + 4n^2 + \frac{5n}{3} + 2nm^2 + mn^2 + 5nm\\ & + m + n + 17 + L\left(d(n^2+3n) + \frac{3n^2}{2} + \frac{n}{2}\right).\end{aligned} \tag{3.6}$$

The computation cost of EICF2 is:

$$\begin{aligned}S_{EICF2} = {} & \frac{25n^3}{3} + 4n^2 + \frac{5n}{3} + 2nm^2 + mn^2 + 5nm + m\\ & + n + 17 + L\left(d(n^2+3n) + \frac{3n^2}{2} + \frac{n}{2}\right) + d\left(\frac{3n}{2} + \frac{n^2}{2}\right).\end{aligned} \tag{3.7}$$

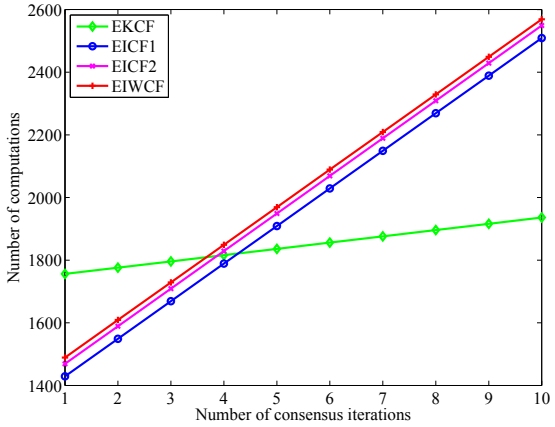


Figure 3.1: Computation cost at a camera node C^i vs. number of consensus iterations. Comparison of the computation cost of the Extended Kalman Consensus Filter (EKCF) [21], the Extended Information Consensus Filter1 (EICF1), the Extended Information Consensus Filter2 (EICF2) and the Extended Information Weighted Consensus Filter (EIWCF).

Similarly, the computation cost of EIWCF is:

$$\begin{aligned} S_{EIWCF} = {} & \frac{25n^3}{3} + 6n^2 + \frac{11n}{3} + 2nm^2 + mn^2 \\ & + 5nm + m + n + 17 + L\left(d(n^2+3n) + \frac{3n^2}{2} + \frac{n}{2}\right). \end{aligned} \tag{3.8}$$

EICF2 has $d \cdot (n + n^2/2 + n/2)$ more computations than EICF1 because of the $\sum \mathbf{u}$ and $\sum \mathbf{U}$ operations in (3.2). EIWCF has $2(n+n^2)$ more operations than EICF1 because of the additional multiplications with N and $1/N$ as shown in (3.3) and (3.4).

Figure 3.1 presents the number of computations for each filter with respect to the number of consensus iterations for a state vector of size $n = 5$ and degree $d = 2$ (here degree refers to the number of communicative neighbours of each node). EKCF has a higher computation cost at lower numbers of consensus iterations because of its more complex update step. By increasing the number of consensus iterations, the computations required for handling the covariance information of new information filters significantly increase and surpass the cost of EKCF.

3.3.2 Communication cost

The number of scalars transmitted by a node is taken as its communication cost $\hat{S}$. For EKCF, this cost is relatively low because only the state vector is exchanged in each iteration. In the first iteration of EKCF, each node sends the measurement information terms, $\mathbf{u}_k^i$ and $\mathbf{U}_k^i$, and the prior

state vector to its neighbours. Hence the number of scalars, $\hat{S}$, transmitted by each node is:

$$\hat{S}_{EKCF} = n + \frac{n(n+1)}{2} + Ln, \tag{3.9}$$

where L is the number of consensus iterations considered. EICF1 exchanges the local estimate, the information vector and the corresponding uncertainty matrix, the information matrix, in each iteration. Hence the communication cost is:

$$\hat{S}_{EICF1} = L\left(n + \frac{n(n+1)}{2}\right). \tag{3.10}$$

EICF2 exchanges the same information as EICF1 but during the first iteration the measurement information, $\mathbf{u}_k^i$ and $\mathbf{U}_k^i$, is also sent to neighbours. Hence the communication cost is:

$$\hat{S}_{EICF2} = n + \frac{n(n+1)}{2} + L\left(n + \frac{n(n+1)}{2}\right). \tag{3.11}$$

Because of the additional $\mathbf{u}_k^i$ and (the upper triangle of) $\mathbf{U}_k^i$ terms, it has an additional communication cost of $n + n(n+1)/2$ scalars. EIWCF exchanges the local estimate, information vector, and the corresponding uncertainty matrix, information matrix, in each iteration. Hence the communication cost is:

$$\hat{S}_{EIWCF} = L\left(n + \frac{n(n+1)}{2}\right). \tag{3.12}$$

As the covariance matrices are symmetric only the upper triangular matrix elements are transferred while sending information matrices $\mathbf{U}_k^i$ and $\mathbf{Y}_k^i$, the cost is $n(n+1)/2$ instead of n^2.

Figure 3.2 presents the number of scalars needed to be transmitted by each node for each filter with respect to the number of consensus iterations considered.

3.4 Simulations

3.4.1 Setup

This chapter compares the new distributed non-linear filters EICF1, EICF2 and EIWCF with the state of the art filter EKCF [21] and the Centralised Extended Information Filter (CEIF) that uses measurement information from all nodes and estimates the target state according to (2.17). As a performance measure, this chapter uses the mean error defined as the $L2-$norm between the true and estimated target position after a given number of consensus iterations [50] (i.e. between $\mathbf{x}_k$

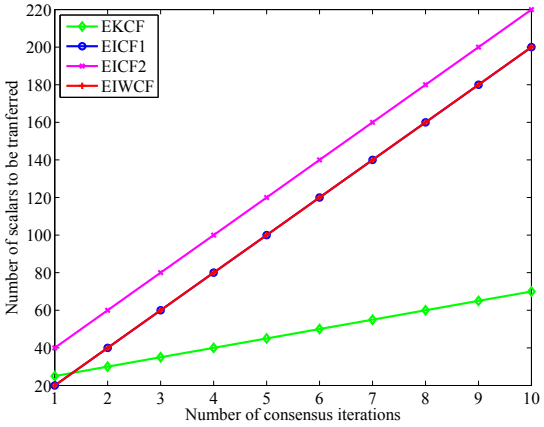


Figure 3.2: Communication cost at a camera node C^i vs. number of consensus iterations. Comparison of the computation cost of the Extended Kalman Consensus Filter (EKCF) [21], the Extended Information Consensus Filter1 (EICF1), the Extended Information Consensus Filter2 (EICF2) and the Extended Information Weighted Consensus Filter (EIWCF).

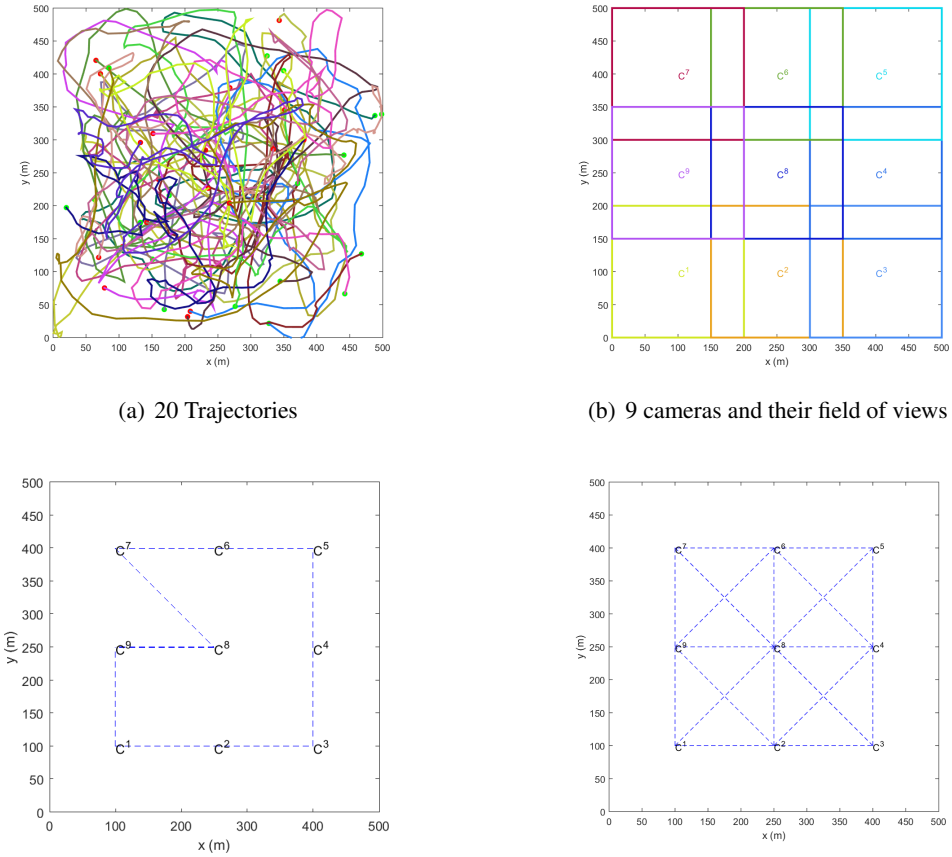
and $\hat{\mathbf{x}}_k^i$). This chapter analyses the faster convergence to the CEIF results.

A simulated target moving in a 500m$\times$500m area which is under observation of $N = 9$ cameras with overlapping FoVs is considered. The FoV of each camera is assumed to be a square region of 200m$\times$200m. The target follows the motion model given by (1.2) and (1.3) with $q = 10$.

The measurement model of the cameras follows (1.4). The state to measurement transition function $h^i(\cdot)$ follows (2.9) with $\sigma^2 = 60$, i.e the standard deviation of measurement error is $\sigma = 7.7$pixels. The homography matrix values of each camera are taken from one of the cameras of the APIDIS dataset [1] whose values are:

$$H^i = \begin{bmatrix} 397.2508 & 95.2020 & 287280 \\ 51.7437 & 396.9189 & 139100 \\ 0.0927 & 0.1118 & 605.2481 \end{bmatrix}. \tag{3.13}$$

This chapter performs the experiments for two types of network connectivity. The first type is limited connectivity with a low average network degree equal to 2 (see Figure 3.3(c)) assuming a very limited communication range for the nodes. The second type considers full connectivity where the degree is higher than the previous case (see Figure 3.3(d)). The communication range is larger and each node communicates with more than two nodes. This setup assumes direct communication between cameras with overlapping FoVs. For each connectivity type, $N_t = 20$ target trajectories are generated (see Figure 3.3(a)), where each track is estimated using $M = 10$ Monte-Carlo simulation runs. These two cases are necessary to analyse accuracy or speed of



(a) 20 Trajectories

(b) 9 cameras and their field of views

(c) Limited connectivity

(d) Full connectivity

Figure 3.3: The simulation setup with a wireless camera network tracking 20 trajectories. Cameras are located in a 500m×500m area and are represented as $C^1,..,C^9$. Their FoVs are represented by $200m \times 200m$ coloured square regions and, the connectivities among them are shown using the blue lines.

convergence with a varying degree of connectivity. The experiments are conducted assuming $\gamma = 1$.

3.4.2 Results

Figure 3.4(a) presents the mean tracking error of the filters for the network shown in Figure 3.3(c). Figure 3.4(b) shows the same results with a focus on the new filters. Figure 3.5(a) presents the mean tracking error of the filters with the setup shown in Figure 3.3(d). Figure 3.5(b) shows the same results with a focus on the new filters. By analysing the tracking error with different filters (Figure 3.4(a) and Figure 3.5(a)), it can be observed that the newly developed distributed filters perform better than EKCF. EICF2 converges to the optimal centralised estimate faster than EICF1 because EICF2 considers neighbours information also. Both EICF1 and EICF2 outperform EKCF, but still the performance of EIWCF is higher compared to EICF1 and

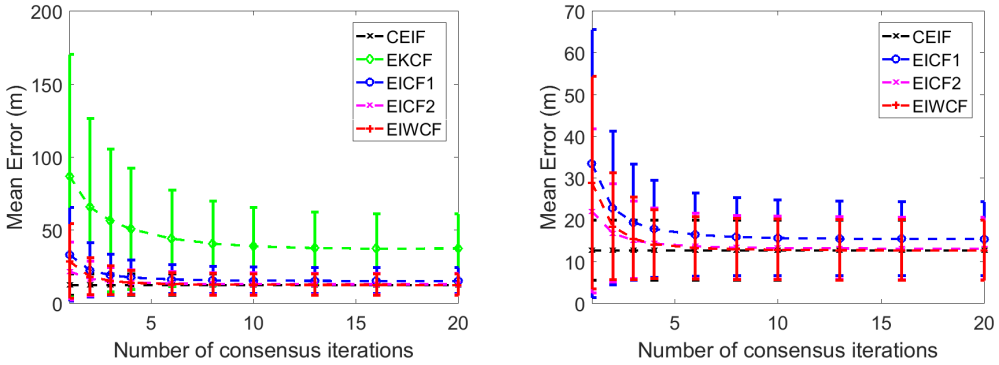


Figure 3.4: Comparison of accuracy for limited connectivity shown in Figure 3.3(c). The compared filters are the Extended Kalman Consensus Filter (EKCF) [21], the Extended Information Consensus Filter1 (EICF1), the Extended Information Consensus Filter2 (EICF2) and the Extended Information Weighted Consensus Filter (EIWCF). Figure 3.4(b) is zoom of Figure 3.4(a) with a focus on the proposed filters EICF1, EICF2 and EIWCF.

EICF2, which do not take the redundancy into account. Only EIWCF converges to the optimal centralised estimate because the prior information (which holds redundant data) and the measurement information are weighted properly to avoid the effect of redundancy. It can also be observed that, with the increase in degree of connectivity, the speed of convergence increases for all filters (see Figure 3.4(a) and Figure 3.5(a)).

3.5 Summary

This chapter proposed the Extended Information Consensus Filter (EICF) by combining the Extended Information Filter (EIF) and Information Consensus Filter (ICF). The proposed filter performs weighted averaging while addressing the problem of naive nodes and non-linearity. The information matrix that represents the uncertainty of each estimated state is used as its weight. As benighted nodes or nodes with high measurement noise levels produce more uncertain estimates so their estimates get lower weights compared to that of the viewing nodes. To overcome the redundancy problem, it also proposed the Extended Information Weighted Consensus Filter (EIWCF) by combining the Extended Information Filter (EIF) and Information Weighted Consensus Filter (IWCF). EIWCF handles naivety, redundancy and non-linearity, and achieves faster convergence by properly weighting prior and measurement information. However, it requires the knowledge of the number of nodes in the network. To consider only one prior out of N priors, each node's prior is weighted less such that the fusion results are as if one prior is used. The

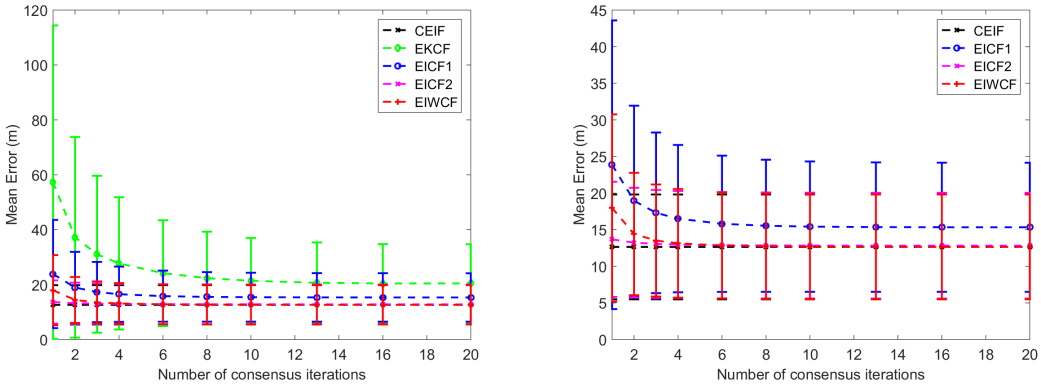


Figure 3.5: Comparison of accuracy for full connectivity shown in Figure 3.3(d). The compared filters are the Extended Kalman Consensus Filter (EKCF) [21], the Extended Information Consensus Filter1 (EICF1), the Extended Information Consensus Filter2 (EICF2) and the Extended Information Weighted Consensus Filter (EIWCF). Figure 3.5(b) is zoom of Figure 3.5(a) with a focus on the proposed filters EICF1, EICF2 and EIWCF.

performance of the EICF and the EIWCF is analysed via simulated WCNs. The results show that the proposed approaches outperform the EKCF and, with additional communication and computation cost, they converge to the centralised result. The proposed algorithms involve the exchange of both the state and the corresponding error covariance so they have higher communication costs than the state of the art consensus algorithms that involve the exchange of the state vector only. Similarly, As the proposed algorithms process also the uncertainty information, they have higher computation costs than the state of the art consensus algorithms that process the state vector only.

Chapter 4

Neighbour Consensus Filter[1]

This chapter presents N-consensus, an algorithm that reduces the cost of the consensus process for distributed visual target tracking without compromising on tracking accuracy. N-consensus reduces the number of nodes involved in the consensus process. N-consensus fuses target posteriors computed by *viewing nodes* (i.e. the cameras viewing the same target) only and limits the number of nodes participating in consensus to those within a specified number of hops from the viewing nodes. Hops are the connections the target information passes through to reach one node from another node in the network. The number of connections is called the hop distance or hop count. Two viewing cameras are separated by at most the sum of their viewing ranges so the maximum possible hop count that guarantees the shortest communication path between two viewing nodes is used as the threshold to allow participation in consensus. A node that is in the threshold hop distance from any of the viewing nodes is allowed to participate in consensus. The neighbourhood of a target is defined based on the hop distance from its viewing nodes. N-consensus achieves consensus only in the neighbourhood.

4.1 Target neighbourhood identification

The Neighbouring nodes (N-Nodes) of the target at time step k, $\mathbf{N}_k$, include all nodes that are viewing the target at the current time step, *current viewing nodes*, $\mathbf{V}_k$, and the nodes that might view the target at the next time step, *future viewing nodes*, $\mathbf{V}_{k^+}$. Nodes other than N-Nodes are *inactive nodes*, $\mathbf{I}_k$ and they do not participate in the consensus process. The inactive nodes that are

[1]This chapter is completely taken from [C3].

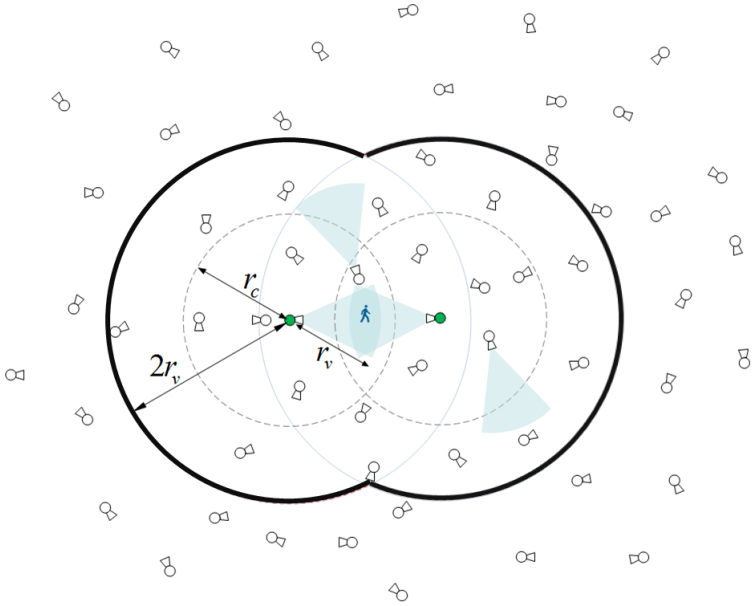


Figure 4.1: Illustration of the consensus neighbourhood (Black) of a target. A network of wireless camera nodes, the field of views (blue) of four nodes, viewing range, r_v, communication range (dotted grey), r_c, of two current viewing nodes (green) are shown. Here r_c and r_v are the physical distances.

single-hop neighbours of N-Nodes act as border between the N-Nodes and the remaining inactive nodes by not transmitting any information. This chapter calls them *sink nodes*, $\mathbf{S}_k$, because they always receive information (from N-Nodes) but they do not send it. The relation between $\mathbf{C}$, $\mathbf{N}_k$, $\mathbf{V}_k$, $\mathbf{V}_{k^+}$, $\mathbf{I}_k$ and $\mathbf{S}_k$ is as follows:

$$\mathbf{C} = \underbrace{\mathbf{V}_k \bigcup \mathbf{V}_{k^+}}_{\mathbf{N}_k} \bigcup \mathbf{I}_k \text{ and } \mathbf{S}_k \subseteq \mathbf{I}_k. \tag{4.1}$$

A constraint for using N-consensus is that the N-Nodes of a target must be connected at any time step.

Ideally, the future viewing nodes include the nodes having overlapping FoVs with all the current viewing nodes. However, nodes are unaware of the FoV information of other nodes so this chapter considers all nodes that are located within twice the viewing range, $2r_v$, distance from each current viewing node as future viewing nodes. This chapter selects the value $2r_v$ because the maximum possible physical distance between two nodes with overlapping FoVs is $2r_v$ [52], and a current viewing node passing information to all nodes within $2r_v$ guarantees that the information is available at all current and future viewing nodes. Figure 4.1 shows a scenario where the number of current viewing nodes is 2. The N-Nodes (nodes within $2r_v$ distance from

the current viewing nodes) are surrounded by a red boundary.

4.2 Distributed neighbourhood fusion

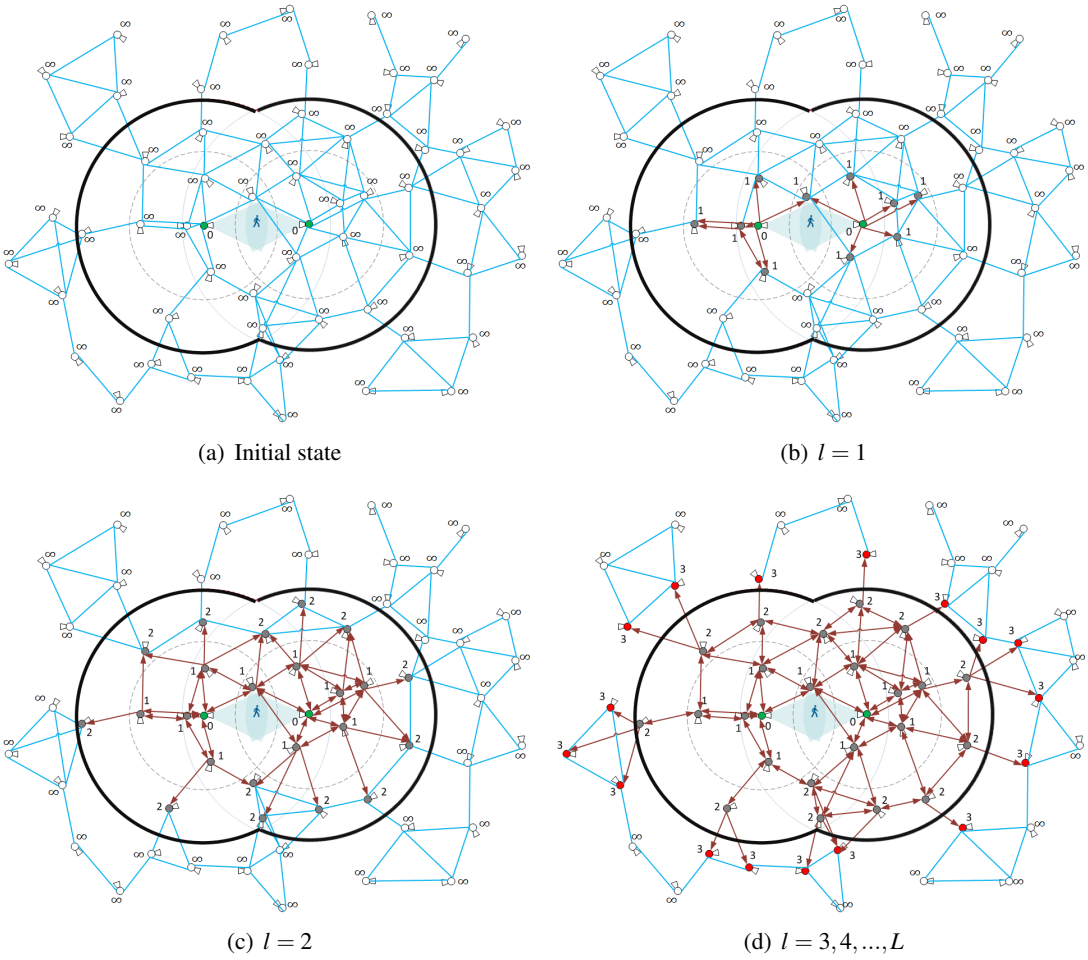
Relative distances among the nodes are required to check if a node is within $2r_v$ distance from any of the current viewing nodes. As the physical locations of the nodes are unknown, the relative distance between two nodes can be approximated either using hop counts [26] or using radio signal strength [63]. This chapter uses the former and perform iterative limited-multi-hop search [61] to identify $\mathbf{V}_{k^+}$. The maximum possible hop distance between a future viewing node and its nearest current viewing node $\hat{D} = \lceil \frac{2r_v}{r_c} \rceil$. This chapter uses this value as the threshold to identify future viewing nodes, which include 1-hop, 2-hop, ..., $\hat{D}$-hop neighbours of each current viewing node. Note that this chapter considers current viewing nodes as 0-hop neighbours of themselves. Consensus is achieved among the $\mathbf{N}_k$ nodes (N-Nodes). Note that, for various r_c and r_v values $\hat{D}$ varies as:

$$\hat{D} > 2, \text{if } r_c < r_v \text{ and}$$

$$\hat{D} = \begin{cases} 2, & \text{if } r_v \leq r_c < 2r_v, \\ 1, & \text{if } r_c \geq 2r_v. \end{cases} \tag{4.2}$$

At each time step, current viewing nodes initialise their hop distance, $D^H_{i,k}$, to zero and compute the local posterior, $\left(\mathbf{y}^i_k, \mathbf{Y}^i_k\right)$, using EIF [60] to handle the non-linearities. Each non-viewing node (node with no target measurement) identifies itself as an inactive node and initialises its hop distance D^i_k to infinity and its local posterior to null, i.e.$\mathbf{y}^i_k = \mathbf{0}_{n\times 1}$ and $\mathbf{Y}^i_k = \mathbf{0}_{n\times n}$. (Figure 4.2(a)). Current viewing nodes initiate the iterative process of information exchange and consensus update.

During the information exchange step, all identified N-Nodes send messages containing their local posterior, $\left(\mathbf{y}^i_k, \mathbf{Y}^i_k\right)$, and hop distance, D^i_k, to all their neighbours. If C^j is the neighbour of C^i, the maximum possible hop distance of C^j from the current viewing nodes is one more than the hop distance of C^i from the current viewing nodes. Hop distance proposal of a node C^j is defined as its hop distance computed by incrementing the hop distance received from a neighbour C^j by one. Each receiving node within the communication range increments the received hop distance by one (i.e. $D^i_k + 1$) and uses the value as a *hop distance proposal* made by the sender. The hop distance of C^j before receiving the message from C^i is called its *local hop distance*. The receiving nodes update their hop distances to the minimum of their local hop distance and the



(a) Initial state (b) $l = 1$

(c) $l = 2$ (d) $l = 3, 4, ..., L$

Figure 4.2: Illustration of N-consensus when the threshold hop distance $\hat{D} = 2$. Black: consensus region, blue: connectivity among the nodes, brown: information flow, green: current viewing nodes, grey: future viewing nodes, red: sink nodes, white: inactive nodes. (a) Initially, only the current viewing nodes identify themselves as N-Nodes. (b), (c) In each consensus iteration ($l \leq \hat{D}$) the neighbourhood of already known N-Nodes identify themselves as future viewing nodes based on their hop distances (shown next to each node). (d) During $l = 3$, the neighbourhood of known N-Nodes at 3-hop distance ($D_k^i > \hat{D}$) identify themselves as sink nodes. The nodes status does not change when $l > 3$.

hop distance proposals (made by their neighbours). $\mathbf{V}_{k^+}$ and $\mathbf{S}_k$ are updated based on the new hop distances.

In the consensus update step, all identified N-Nodes fuse their local posteriors with the received posteriors using the IFCI algorithm [31] as follows:

$$[\mathbf{y}_{k(l)}^i \ \mathbf{Y}_{k(l)}^i] = \left[\sum_{C^{i'} \in \mathbf{N}_k^i} w_{k(l)}^{i'} \mathbf{y}_{k(l-1)}^{i'} \sum_{C^{i'} \in \mathbf{N}_k^i} w_{k(l)}^{i'} \mathbf{Y}_{k(l-1)}^{i'} \right], \tag{4.3}$$

where

$$w_{k(l)}^{i'} = \frac{\left|\sum_{C^{j'}\in\mathbf{N}_k^i} Y_{k(l-1)}^{j'}\right| + \left|Y_{k(l-1)}^{i'}\right| - \left|\sum_{C^{j'}\in\mathbf{N}_k^i} Y_{k(l-1)}^{j'} - Y_{k(l-1)}^{i'}\right|}{\sum_{C^j\in\mathbf{N}_k^i}\left[\left|\sum_{C^{j'}\in\mathbf{N}_k^i} Y_{k(l-1)}^{j'}\right| + \left|Y_{k(l-1)}^{j}\right| - \left|\sum_{C^{j'}\in\mathbf{N}_k^i} Y_{k(l-1)}^{j'} - Y_{k(l-1)}^{j}\right|\right]}. \tag{4.4}$$

Here $\left|.\right|$ represents determinant and l is the iteration index and $\mathbf{N}_k^i$ is the set of all N-Nodes in the communication range of C^i at k. N-consensus algorithm is more complex than A-consensus and trace based ICI because of the computation of determinants.

In iterations $l = 1,2,...,\hat{D}$, future viewing nodes that are 1-hop, 2-hop, ..., $\hat{D}$-hop neighbours are identified. When $l = \hat{D}+1$, sink nodes are identified. When $l > \hat{D}+1$, the sets $\mathbf{V}_k, \mathbf{V}_{k^+}, \mathbf{I}_k, \mathbf{S}_k$ do not change.

Between two time steps, L consensus iterations are run. Figure 4.2 illustrates the N-consensus iterations with an example. When using fewer iterations than the threshold (i.e. $L < \hat{D}$), the posterior might not be available at all future viewing nodes. For example, in Figure 4.2(b), $l = 1$ and $\hat{D} = 2$, all 1-hop neighbours are identified but 2-hop neighbours are not yet identified so the target posterior is not available at the 2-hop neighbours. In the next time step, if the target enters the FoV of any 2-hop neighbour, the EIF running at the node fails to compute the posterior because the posterior from the previous time step is not available. Hence, N-consensus cannot perform tracking unless a minimum of $L = \hat{D}$ iterations is used. Running the consensus algorithm for $\hat{D}$ iterations ensures the identification of all N-Nodes and for $\hat{D}+1$ iterations ensures the information exchange by all the N-Nodes. For example, in Figure 4.2, though all N-Nodes are identified during iteration $l = 2$ (Figure 4.2(c)), some neighbouring N-Nodes started exchanging information during iteration 3 (Figure 4.2(d)). N-consensus ensures that the node having a measurement at k but not at $k-1$ holds the posterior from $k-1$ because the node must have participated in consensus at $k-1$ as a future viewing node.

4.3 Simulations

The thesis compares the performance of four fusion algorithms, namely centralised fusion using Improved Fast Covariance Intersection (CCI) [31], distributed fusion using A-consensus (AC) [C5], Iterative Covariance Intersection (ICI) [41] and the proposed N-consensus (NC) using numerical simulations.

The thesis uses as performance measures accuracy and communication cost. At each time

step, the average of the position estimates of all the N-Nodes is considered as the estimated target position. *Accuracy* is the Euclidean distance between the estimated target positions and the corresponding ground-truth positions on the ground plane. *Communication cost* can be evaluated either as the total number of scalars transmitted in the network or as energy consumption for their transmission and reception. The energy spent not only depends on the number of scalars (or the number of bits) transmitted but also on the communication range of each transmitting node. The energy is calculated by summing transmission energy $E_t = E_e b + apr_c^2$ and receiving energy $E_r = E_e b$, where E_e is the electrical energy ($Joules/bit$) used for running transmitter or receiver components, a is the power amplification ($Joules/bit/m^2$) required to guarantee acceptable received signal strength within the communication range r_c and, b is the number of bits transmitted or received [78].

4.3.1 Setup

Let the WCN contain 256 homogeneous cameras that monitor a $500m \times 500m$ area. Each camera has a (directional) viewing range $r_v = 50m$ and 90^o FoV. The position and FoV of each camera are kept constant (Figure 4.3(a)). The motion model of the target is as given by (1.2) and (1.3) with $q = 10$.

This chapter uses $Nt = 20$ trajectories (Figure 4.3(b)) for performance analysis. Each trajectory is estimated using $M = 10$ Monte-Carlo simulations. The measurement model of the cameras follows (1.4). The state to measurement transition function $h^i(\cdot)$ follows (2.9) with $\sigma^2 = 60$, i.e the standard deviation of measurement error is $\sigma = 7.7$pixels. The values $H^i(1,1),...,H^i(3,3)$ are the elements of the homography matrix H^i and are taken from one of the cameras of APIDIS dataset[6] as:

$$H^i = \begin{bmatrix} 397.2508 & 95.2020 & 287280 \\ 51.7437 & 396.9189 & 139100 \\ 0.0927 & 0.1118 & 605.2481 \end{bmatrix}. \tag{4.5}$$

At each time step, the local posteriors (to be fused) are estimated by EIF running at each node. Both in A-consensus and ICI, non-viewing nodes use predicted posteriors as their local posteriors and $\frac{0.65}{\Delta_{max}}$ as the weight of each neighbour's information for A-consensus update assuming that each camera knows the maximum degree (Δ_{max}) of the underlying communication graph [65]. The tracking experiment is conducted by considering the communication range r_c of each node

[6] http://sites.uclouvain.be/ispgroup/index.php/Softwares/APIDIS, last accessed February 2015.

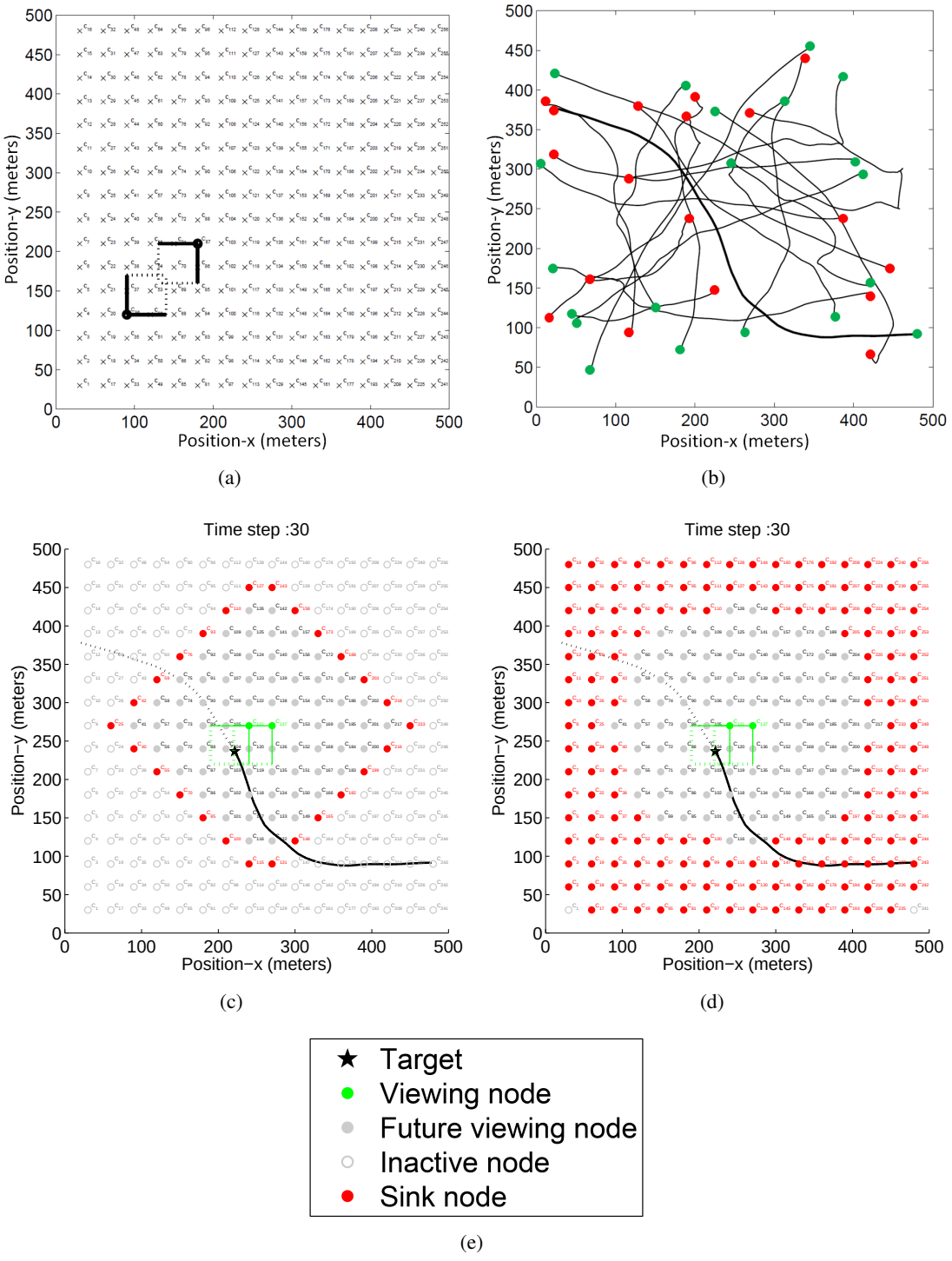
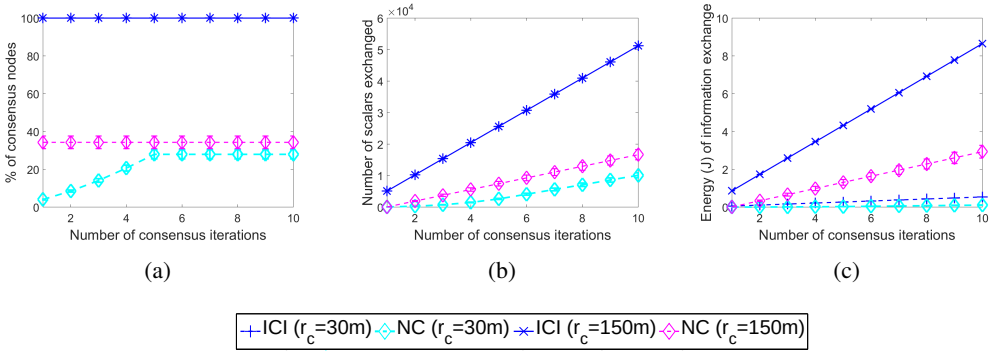


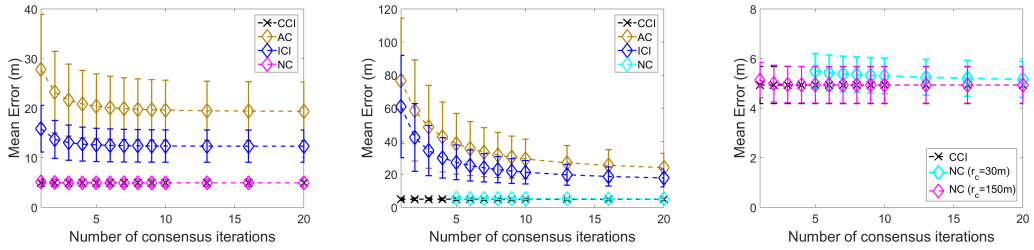
Figure 4.3: Simulation setup. (a) Wireless camera network surveilling a $500m \times 500m$ area using cameras $C^1, ..., C^{256}$. Each camera has a Field of View (FoV) of $50m \times 50m$ on the ground plane (black). (b) Sample trajectories used in the experiments with their starting points (green) and ending points (red). (c) and (d) N-Nodes at time step 30 of the bold track shown in (b) when the communication range of nodes are $r_c = 30m$ ($\hat{D} = 5$) and $r_c = 150m$ ($\hat{D} = 1$), respectively. The viewing cameras and their FoVs are shown in green.



(a) (b) (c)

(d)

Figure 4.4: Communication cost analysis. (a) The average percentage of nodes participating in consensus for the two r_c values ($30m$ and $150m$). Average communication cost in terms of (b) number of scalars transmitted and (c) energy (Joules) spent. The algorithms are Iterative Covariance Intersection (ICI) [41] and the proposed Neighbourhood-consensus (NC). Note that the results of ICI for the two r_c values overlap. As the results of Average Consensus (AC) [C5] are the same as ICI, they are not reported here.



(a) (b) (c)

Figure 4.5: Accuracy analysis when the communication range, r_c, of the nodes is (a) $30m$ and (b) $150m$. The algorithms are: Centralised fusion using IFCI (CCI) [31], distributed fusion using Average Consensus (AC) [C5], Iterative Covariance Intersection (ICI) [41] and the proposed Neighbourhood-consensus (NC). (c) is the zoom of (a) and (b) with a focus on NC.

as $30m$ ($< r_v$) and $150m$ ($> 2r_v$). This chapter analyses the accuracy and communication cost of tracking the 20 trajectories for a different number of consensus iterations, L in both cases.

4.3.2 Results

The hop distance thresholds $\hat{D} = \lceil \frac{2r_v}{r_c} \rceil$ are 5 and 1 when the r_c values are $30m$ and $150m$, respectively. The N-Nodes in each case are shown in Figure 4.3(c) and 4.3(d). More nodes are identified as N-Nodes for $r_c = 150m$ compared to $r_c = 30m$ because the higher communication range turns more nodes to be 1-hop (future viewing nodes) and 2-hop neighbours (sink nodes). N-Nodes are not completely identified until $\hat{D}$ iterations, so for $r_c = 30m$ the percentage of N-Nodes increases for iterations 1 to 5 and from the 5^{th} iteration the value does not change. Note

that the increment is not linear and depends on the number of newly identified future viewing nodes in each iteration. Figure 4.4(a) shows the percentage of nodes participating in consensus in ICI and N-consensus. By using N-consensus, the number of participating nodes is reduced to approximately 35%.

Each transmission of N-consensus involves a message containing information vector $\mathbf{y}_k^i$, information matrix $\mathbf{Y}_k^i$ and hop distance. The target state vector size is $n = 4$ and the information matrix is symmetric. To reduce the number of scalar transmissions this chapter sends only the upper triangular values, i.e. 10 scalars. Hence, the posterior $(\mathbf{y}_k^i, \mathbf{Y}_k^i)$ contains 14 scalars each of which is a 32-bit floating point number. The hop distance is a 16-bit integer. Therefore, each N-consensus message contains $(14 \times 32) + (1 \times 16) = 464$ bits. The values of E_e and a are considered as $50nJ$ and $0.1nJ/b/m^2$, respectively. As only the N-Nodes participate in consensus, the number of transmissions is smaller than that of ICI and A-consensus in which all nodes participate. The total communication cost is therefore smaller in N-consensus than that of ICI and A-consensus. As the N-Nodes are more for $r_c = 150m$ than for $r_c = 30m$, the number of scalars transmitted are also more for $r_c = 150m$ than for $r_c = 30m$ (Figure 4.4(b)) of N-consensus. The energy consumption is also smaller in N-consensus than that of ICI and A-consensus. While the number of scalars transmitted by ICI (and A-consensus) is the same for the two communication ranges, the energy spent is different because of the different transmission ranges. As one would expect, N-consensus with $r_c = 30m$ consumes less energy than N-consensus with $r_c = 150m$ and ICI (and A-consensus) with both the communication ranges (Figure 4.4(c)) because of the smaller number of N-Nodes and the lower transmission range. The communication cost of A-consensus and ICI are the same so the cost of A-consensus is not shown in Figure 4.4.

As mentioned earlier, N-consensus cannot perform tracking until all N-Nodes are identified so tracking is not feasible when using consensus iterations less than D. For $r_c = 30m$, until the 5^{th} iteration the error is not available (Figure 4.5). For $r_c = 150m$, $D = 1$ so tracking is performed for all the iterations used. The N-consensus estimate achieves faster convergence to the centralised estimate compared to the other algorithms. The error computed using CCI is $4.9m$. N-consensus with $r_c = 150m$ spends $0.34J$ energy for 2 iterations and has a mean error $5.0m$, whereas N-consensus with $r_c = 30m$ spends $0.26J$ for 20 iterations and has a mean error $5.1m$.

4.4 Summary

This chapter proposed N-consensus, an algorithm for achieving consensus on target posteriors among only a set of Neighbouring Nodes (N-Nodes) of the target. N-Nodes include current viewing nodes (0-hop neighbours) and future viewing nodes (1-hop, 2-hop, ..., $\hat{D}$-hop neighbours). N-consensus selects $\hat{D}$ based on the viewing and the communication ranges of the nodes. Consensus update fuses posteriors using the covariance intersection algorithm to avoid the necessity of topology information and provides better fusion estimates. Experimental results show that the proposed N-consensus approach provides better accuracy and requires fewer communication resources compared to average consensus and iterative covariance intersection. The advantages of reducing the communication cost are as follows. transmitting less information in the network not only reduces the energy consumption but also results in better channel utilisation. It also reduces communication delays because the channel is more time available to access. In the case of battery powered cameras, reduction of communication and computation costs increases the lifetime of the network. Unlike other consensus approaches, in N-consensus, the target state is available at N-Nodes only so non N-Nodes (inactive nodes) cannot take decisions about the target.

Chapter 5

Average Consensus-based Asynchronous Filter[1]

In the asynchronous case, the cameras capture frames at different instants. This chapter assumes the asynchronism to be partial (as in [33]), where α is the upper bound of the relative capturing offset (see Figure 5.1), i.e. $\alpha = \alpha^{max} = \max_{i,j}\{\alpha^{ij}\}$. Each camera knows T and α. This chapter proposes an Average Consensus-based Asynchronous tracking Filter (ACAF) for WCNs. The nodes perform information-alignment with respect to their capturing instant by predicting the information of the neighbours. This chapter first presents the Bayesian formulation of the idea and then present its implementation under Gaussian assumptions using the IF.

5.1 Bayesian asynchronous consensus

At the beginning of the fusion phase, each node C^i performs three predictions. The first predicts the target pdf for $k-\alpha$ based on the computed local pdf $p(\mathbf{x}^i_k|\mathbf{z}^i_{1:k})$ (backward prediction) as

$$p(\tilde{\mathbf{x}}^i_{k-\alpha}|\mathbf{z}^i_{1:k}) = \int p(\tilde{\mathbf{x}}^i_{k-\alpha}|\mathbf{x}^i_k)p(\mathbf{x}^i_k|\mathbf{z}^i_{1:k})d\mathbf{x}^i_k. \tag{5.1}$$

This chapter uses $k-\alpha$ because other cameras in the network must have captured at most α time steps earlier or at most α time steps later (partial asynchronism assumption). The second prediction is based on the previously known pdf $p(\hat{\mathbf{x}}^i_{k'}|\mathbf{z}^i_{1:k'})$ (forward prediction) as

$$p(\tilde{\mathbf{x}}^i_{k-\alpha}|\mathbf{z}^i_{1:k'}) = \int p(\tilde{\mathbf{x}}^i_{k-\alpha}|\hat{\mathbf{x}}^i_{k'})p(\hat{\mathbf{x}}^i_{k'}|\mathbf{z}^i_{1:k'})d\hat{\mathbf{x}}^i_{k'}. \tag{5.2}$$

[1] This chapter is completely taken from [C2].

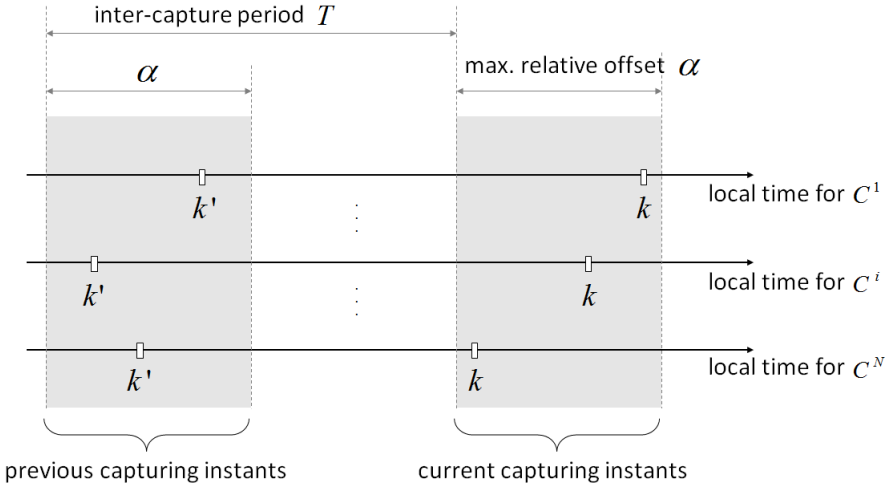


Figure 5.1: Local clocks in partial asynchronism. Key – T: inter-frame capturing period, α: amount of asynchronism.

The predicted pdfs $p(\tilde{\mathbf{x}}^i_{k-\alpha}|\mathbf{z}^i_{1:k})$ and $p(\tilde{\mathbf{x}}^i_{k-\alpha}|\mathbf{z}^i_{1:k'})$ are then compared and the one with the lowest uncertainty is considered. This step helps to avoid over-prediction when a camera does not have measurements due to an occlusion or the limited (directional) FoV. Let $p(\tilde{\mathbf{x}}^{i,*}_{k-\alpha}|\mathbf{z}^i_{1:k})$ be the pdf with lower uncertainty. The third predicts the pdf for the capturing instant k based on the certain predicted pdf corresponding to $k-\alpha$ (forward prediction) as

$$p(\tilde{\mathbf{x}}^i_{k(\tau^i_k)}|\mathbf{z}^i_{1:k}) = \int p(\tilde{\mathbf{x}}^i_{k(\tau^i_k)}|\tilde{\mathbf{x}}^{i,*}_{k-\alpha})p(\tilde{\mathbf{x}}^{i,*}_{k-\alpha}|\mathbf{z}^i_{1:k})d\tilde{\mathbf{x}}^{i,*}_{k-\alpha}. \tag{5.3}$$

In other words, it predicts the target pdf for the same capturing instant k via backward and forward predictions. The subscript $k(l)$ is used to indicate that the information corresponds to k before starting the consensus iteration at $k+l$ ($l \geq \tau^i_k$). τ^i_k is the estimation delay of C^i at k.

Camera nodes fuse these predicted pdfs $p(\tilde{\mathbf{x}}^i_{k(\tau^i_k)}|\mathbf{z}^i_{1:k}), \forall C^i$ via distributed average consensus. Let γ be the periodicity of the consensus iterations. Each node can compute the elapsed time after an estimation phase and after each consensus iteration.

Each consensus iteration involves two predictions, one before the transmission and one after the reception. Before transmission, each node C^i predicts the pdf for the transmission instant $k+l$ based on the predicted pdf $p(\tilde{\mathbf{x}}^i_{k(l)}|\mathbf{z}^i_{1:k})$ corresponding to the capturing instant k (forward prediction) as

$$p(\tilde{\mathbf{x}}^i_{k+l}|\mathbf{z}^i_{1:k}) = \int p(\tilde{\mathbf{x}}^i_{k+l}|\tilde{\mathbf{x}}^i_{k(l)})p(\tilde{\mathbf{x}}^i_{k(l)}|\mathbf{z}^i_{1:k})d\tilde{\mathbf{x}}^i_{k(l)}. \tag{5.4}$$

The predicted pdf $p(\tilde{\mathbf{x}}^i_{k+l}|\mathbf{z}^i_{1:k})$ represents the opinion of the sender C^i at the transmission instant.

The node C^i sends the predicted pdf $p(\tilde{\mathbf{x}}^i_{k+l}|\mathbf{z}^i_{1:k})$ to its neighbours $\mathcal{N}^i$.

If C^i receives a similarly predicted pdf $p(\tilde{\mathbf{x}}^j_{k''}|\mathbf{z}^j_{1:k''})$ from C^j at any local time k'' after its capturing instant (i.e. $k'' \in [k, k+l]$), it stores the received pdf in a buffer. During the consensus update at $k+l$, C^i predicts the pdf of C^j for C^i's capturing instant k based on the received pdf (reverse prediction) as

$$p(\tilde{\mathbf{x}}^j_{k(l)}|\mathbf{z}^j_{1:k''}) = \int p(\tilde{\mathbf{x}}^j_k|\tilde{\mathbf{x}}^j_{k''}) p(\tilde{\mathbf{x}}^j_{k''}|\mathbf{z}^j_{1:k''}) d\tilde{\mathbf{x}}^j_{k''}. \tag{5.5}$$

This predicted pdf represents the opinion of the sender C^j for the capturing instant k of the receiver C^i. Now, C^i fuses the temporally aligned pdfs $p(\tilde{\mathbf{x}}^i_{k(l)}|\mathbf{z}^i_{1:k})$ and $p(\tilde{\mathbf{x}}^j_{k(l)}|\mathbf{z}^j_{1:k''}), \forall C^j \in \mathcal{N}^i$ as

$$p(\tilde{\mathbf{x}}^i_{k(l+\gamma)}|\mathbf{z}^i_{1:k}) = \frac{p(\tilde{\mathbf{x}}^i_{k(l)}|\mathbf{z}^i_{1:k})^c p(\tilde{\mathbf{x}}^j_{k(l)}|\mathbf{z}^j_{1:k''})^{1-c}}{\int p(\tilde{\mathbf{x}}^i_{k(l)}|\mathbf{z}^i_{1:k})^c p(\tilde{\mathbf{x}}^j_{k(l)}|\mathbf{z}^j_{1:k''})^{1-c} d\tilde{\mathbf{x}}^i_{k(l)}}, \tag{5.6}$$

where c is the weight given to the instantaneous pdf of the node. The fusion happens for all the received neighbours' pdfs. The fusion result is used in the next consensus iteration that repeats (5.4), (5.5) and (5.6) at $k+l+\gamma$.

Asymptotically, the fusion result converges to the KLA of the predicted local pdfs of all the cameras, i.e.

$$p(\tilde{\mathbf{x}}^+_{k(\infty)}|\mathbf{z}^+_{1:k}) = \frac{\prod_{j=1}^{N} p(\tilde{\mathbf{x}}^j_{k(\tau^j_k)}|\mathbf{z}^j_{1:k})^{\frac{1}{N}}}{\int \prod_{j=1}^{N} p(\tilde{\mathbf{x}}^j_{k(\tau^j_k)}|\mathbf{z}^j_{1:k})^{\frac{1}{N}} d\tilde{\mathbf{x}}^j_{k(\tau^j_k)}}. \tag{5.7}$$

The superscript $+$ (instead of a camera index) represents that the result is available at all cameras. As each node C^i is aware of its own capturing instant, it replaces its contribution in the KLA, i.e. the predicted local pdf $p(\tilde{\mathbf{x}}^i_{k(\tau^i_k)}|\mathbf{z}^i_{1:k})$, with the actual local pdf $p(\mathbf{x}^i_k|\mathbf{z}^i_{1:k})$. This is the correction step and is as follows:

$$p(\hat{\mathbf{x}}^i_k|\mathbf{z}^i_{1:k}) = \frac{p(\tilde{\mathbf{x}}^i_{k(\infty)}|\mathbf{z}^i_{1:k}) p(\tilde{\mathbf{x}}^i_{k(\tau^i_k)}|\mathbf{z}^i_{1:k})^{-\frac{1}{N}} p(\mathbf{x}^i_k|\mathbf{z}^i_{1:k})^{\frac{1}{N}}}{\int p(\tilde{\mathbf{x}}^i_{k(\infty)}|\mathbf{z}^i_{1:k}) p(\tilde{\mathbf{x}}^i_{k(\tau^i_k)}|\mathbf{z}^i_{1:k})^{-\frac{1}{N}} p(\mathbf{x}^i_k|\mathbf{z}^i_{1:k})^{\frac{1}{N}} d\tilde{\mathbf{x}}^i_{k(\infty)}}. \tag{5.8}$$

To avoid the fusion of information corresponding to subsequent frame captures, a node terminates its consensus phase if the time elapsed since the frame capture is $T - \alpha$. Figure 5.2 shows the block diagram of ACAF.

The next section derives an approximation of the above Bayesian fusion method under Gaus-

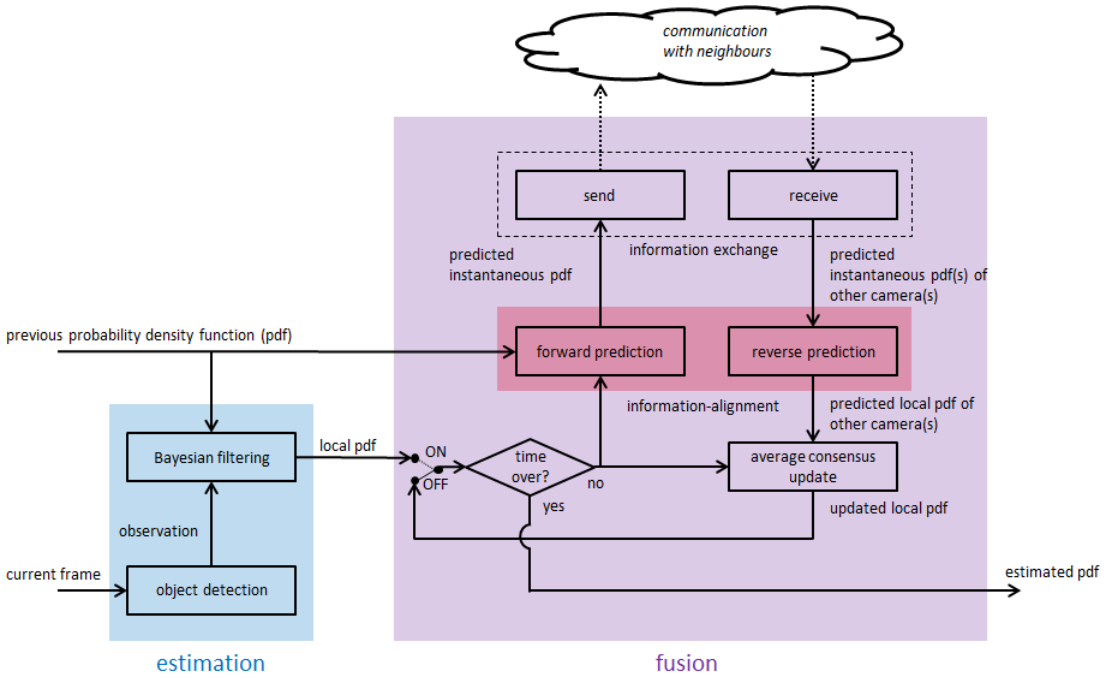


Figure 5.2: Block diagram of ACAF running in each camera. The Switch is ON during the first consensus iteration. Otherwise OFF.

sian assumptions.

5.2 Information Filter based asynchronous consensus

In the *estimation phase*, each node C^i computes the local pdf $p(\mathbf{x}_k^i|\mathbf{z}_{1:k}^i)$ represented by $(\mathbf{x}_k^i, \mathbf{P}_k^i)$ using the Information Filter [15]. Here, $\mathbf{x}_k^i$ and $\mathbf{P}_k^i$ represent the minimum mean square error estimate and the corresponding error covariance of the estimated target pdf $p(\mathbf{x}_k^i|\mathbf{z}_{1:k}^i)$. The information pair corresponding to the estimate $(\mathbf{x}_k^i, \mathbf{P}_k^i)$ is $(\mathbf{y}_k^i, \mathbf{Y}_k^i) = (\mathbf{P}_k^{i\,-1}\mathbf{x}_k^i, \mathbf{P}_k^{i\,-1})$.

In the *fusion phase*, each C^i performs backward prediction of the pair from k to $k-\alpha$ as

$$\begin{aligned}\tilde{\mathbf{Y}}_{k-\alpha|k}^i &= \left(\mathbf{F}(k,k-\alpha)\mathbf{Y}_k^{i\,-1}\mathbf{F}(k,k-\alpha)^\top + \mathbf{Q}(k,k-\alpha)\right)^{-1}, \\ \tilde{\mathbf{y}}_{k-\alpha|k}^i &= \tilde{\mathbf{Y}}_{k-\alpha|k}^i\mathbf{F}(k,k-\alpha)\left(\mathbf{Y}_k^{i\,-1}\mathbf{y}_k^i\right);\end{aligned} \tag{5.9}$$

and forward prediction of the pair from k' to $k-\alpha$ as

$$\begin{aligned}\tilde{\mathbf{Y}}_{k-\alpha|k'}^i &= \left(\mathbf{F}(k',k-\alpha)\hat{\mathbf{Y}}_{k'}^{i\,-1}\mathbf{F}(k',k-\alpha)^\top + \mathbf{Q}(k',k-\alpha)\right)^{-1}, \\ \tilde{\mathbf{y}}_{k-\alpha|k'}^i &= \tilde{\mathbf{Y}}_{k-\alpha|k'}^i\mathbf{F}(k',k-\alpha)\left(\hat{\mathbf{Y}}_{k'}^{i\,-1}\hat{\mathbf{y}}_{k'}^i\right).\end{aligned} \tag{5.10}$$

Here, $(\hat{\mathbf{y}}_{k'}^i, \hat{\mathbf{Y}}_{k'}^i)$ is the information pair of the known pdf corresponding to $k' < k$. As the certainty

of a distribution is proportional to the trace of its information matrix, the information pair between $(\tilde{\mathbf{y}}^i_{k-\alpha|k}, \tilde{\mathbf{Y}}^i_{k-\alpha|k})$ and $(\tilde{\mathbf{y}}^i_{k-\alpha|k'}, \tilde{\mathbf{Y}}^i_{k-\alpha|k'})$ with the higher trace is considered as the winning pair $(\tilde{\mathbf{y}}^{i,*}_{k-\alpha}, \tilde{\mathbf{Y}}^{i,*}_{k-\alpha})$.

C^i performs forward prediction of the pair from $k-\alpha$ to k as

$$\begin{aligned}\tilde{\mathbf{Y}}^i_{k(\tau^i_k)} &= \left(\mathbf{F}(k-\alpha,k)\tilde{\mathbf{Y}}^{i,*}_{k-\alpha}{}^{-1}\mathbf{F}(k-\alpha,k)^\top + \mathbf{Q}(k-\alpha,k)\right)^{-1},\\ \tilde{\mathbf{y}}^i_{k(\tau^i_k)} &= \tilde{\mathbf{Y}}^i_{k(\tau^i_k)}\mathbf{F}(k-\alpha,k)\left(\tilde{\mathbf{Y}}^{i,*}_{k-\alpha}{}^{-1}\tilde{\mathbf{y}}^{i,*}_{k-\alpha}\right).\end{aligned} \tag{5.11}$$

Each consensus iteration at $k+l$ ($l \geq \tau^i_k$) consists of four steps, namely forward prediction, information alignment, fusion and correction. The *forward prediction* of the pair from k to $k+l$ is performed as

$$\begin{aligned}\tilde{\mathbf{Y}}^i_{k+l} &= \left(\mathbf{F}(k,k+l)\tilde{\mathbf{Y}}^i_{k(l)}{}^{-1}\mathbf{F}(k,k+l)^\top + \mathbf{Q}(k,k+l)\right)^{-1},\\ \tilde{\mathbf{y}}^i_{k+l} &= \tilde{\mathbf{Y}}^i_{k+l}\mathbf{F}(k,k+l)\left(\tilde{\mathbf{Y}}^i_{k(l)}{}^{-1}\tilde{\mathbf{y}}^i_{k(l)}\right).\end{aligned} \tag{5.12}$$

C^i transmits the pair $(\tilde{\mathbf{y}}^i_{k+l}, \tilde{\mathbf{Y}}^i_{k+l})$ to its neighbours $\mathcal{N}^i$.

The second step is *information alignment*. Let $k'' \in [k, k+l]$ be the local time instant when C^i receives the pair $(\tilde{\mathbf{y}}^j_{k''}, \tilde{\mathbf{Y}}^j_{k''})$ from C^j. C^i predicts the information pair of C^j for k via reverse prediction as

$$\begin{aligned}\tilde{\mathbf{Y}}^j_{k(l)} &= \mathbf{F}(k,k'')^\top\left(\tilde{\mathbf{Y}}^j_{k''}{}^{-1} - \mathbf{Q}(k,k'')\right)^{-1}\mathbf{F}(k,k''),\\ \tilde{\mathbf{y}}^j_{k(l)} &= \tilde{\mathbf{Y}}^j_{k(l)}\mathbf{F}(k'',k)\left(\tilde{\mathbf{Y}}^j_{k''}{}^{-1}\tilde{\mathbf{y}}^j_{k''}\right).\end{aligned} \tag{5.13}$$

In the information *fusion* step, the predicted pair $\left(\tilde{\mathbf{y}}^i_{k(l)}, \tilde{\mathbf{Y}}^i_{k(l)}\right)$ and the predicted pairs $\left(\tilde{\mathbf{y}}^j_{k(l)}, \tilde{\mathbf{Y}}^j_{k(l)}\right), \forall C^j \in \mathcal{N}^i$ are fused via the average consensus update as

$$\begin{aligned}\tilde{\mathbf{Y}}^i_{k(l+\gamma)} &= \tilde{\mathbf{Y}}^i_{k(l)} + c\sum_{\forall C^j \in \mathcal{N}^i}\left(\tilde{\mathbf{Y}}^j_{k(l)} - \tilde{\mathbf{Y}}^i_{k(l)}\right),\\ \tilde{\mathbf{y}}^i_{k(l+\gamma)} &= \tilde{\mathbf{y}}^i_{k(l)} + c\sum_{\forall C^j \in \mathcal{N}^i}\left(\tilde{\mathbf{y}}^j_{k(l)} - \tilde{\mathbf{y}}^i_{k(l)}\right).\end{aligned} \tag{5.14}$$

Here, $c \in \left(0, \frac{1}{\Delta_{max}}\right)$, where $\Delta_{max} = \max_{\forall C^i \in \mathbf{C}}\{|\mathcal{N}^i|\}$.

The *correction* step replaces the initial predicted local information pair $\left(\tilde{\mathbf{y}}^i_{k(\tau^i_k)}, \tilde{\mathbf{Y}}^i_{k(\tau^i_k)}\right)$ with

the actual local information pair $\left(\mathbf{y}_k^i, \mathbf{Y}_k^i\right)$ as

$$\begin{aligned}\hat{\mathbf{Y}}_k^i &= \tilde{\mathbf{Y}}_{k(l+\gamma)}^i - \frac{\tilde{\mathbf{Y}}_{k(\tau_k^i)}^i}{N} + \frac{\mathbf{Y}_k^i}{N}, \\ \hat{\mathbf{y}}_k^i &= \tilde{\mathbf{y}}_{k(l+\gamma)}^i - \frac{\tilde{\mathbf{y}}_{k(\tau_k^i)}^i}{N} + \frac{\mathbf{y}_k^i}{N}.\end{aligned} \tag{5.15}$$

The state estimate $\hat{\mathbf{x}}_k^i$ and the corresponding error covariance $\hat{\mathbf{P}}_k^i$ are

$$\hat{\mathbf{x}}_k^i = \hat{\mathbf{Y}}_k^{i\,-1}\hat{\mathbf{y}}_k^i, \text{ and } \hat{\mathbf{P}}_k^i = \hat{\mathbf{Y}}_k^{i\,-1}. \tag{5.16}$$

If the time elapsed since the capturing instant k is larger than $T-\alpha$, then C^i terminates its fusion phase. Otherwise, the same process, (5.12)-(5.15), is repeated using $\left(\tilde{\mathbf{y}}_{k(l+\gamma)}^i, \tilde{\mathbf{Y}}_{k(l+\gamma)}^i\right)$ as input, i.e. $l \leftarrow l+\gamma$.

Note that when $\alpha = 0$ (synchronous case), ICF and ACAF yield the same result but differ in the type of information exchanged: ACAF exchanges predicted information corresponding to $k+l$, whereas ICF exchanges the actual information corresponding to k.

5.3 Simulations

This chapter compares the performance of ACAF, the proposed filter, with (i) ICF [6], which uses average consensus assuming synchronous setting; (ii) aCDTT [33], which uses maximum consensus in asynchronous settings; (iii) the distributed filter that computes the local state estimates without fusion (No fusion); and (iv) a centralised filter (CEN) that assumes the FC is aware of the capturing instants, i.e. the delays are known. CEN performs the proposed information alignment at the FC.

ACAF requires fewer scalar transmissions than ICF and aCDTT. In particular, ACAF has fewer transmissions than ICF because of the early termination of the consensus phase. In aCDTT, each consensus iteration exchanges the instantaneous local estimate, the index of the camera that generated the estimate and the label to distinguish information from subsequent estimation phases. In contrast, only the local estimates are exchanged in ACAF and ICF.

5.3.1 Setup

This chapter uses for a WCN that monitors a 30m $\times$ 20m area using $N = 7$ static cameras whose positions and FoVs are taken from the APIDIS dataset [1]. The validation is conducted for

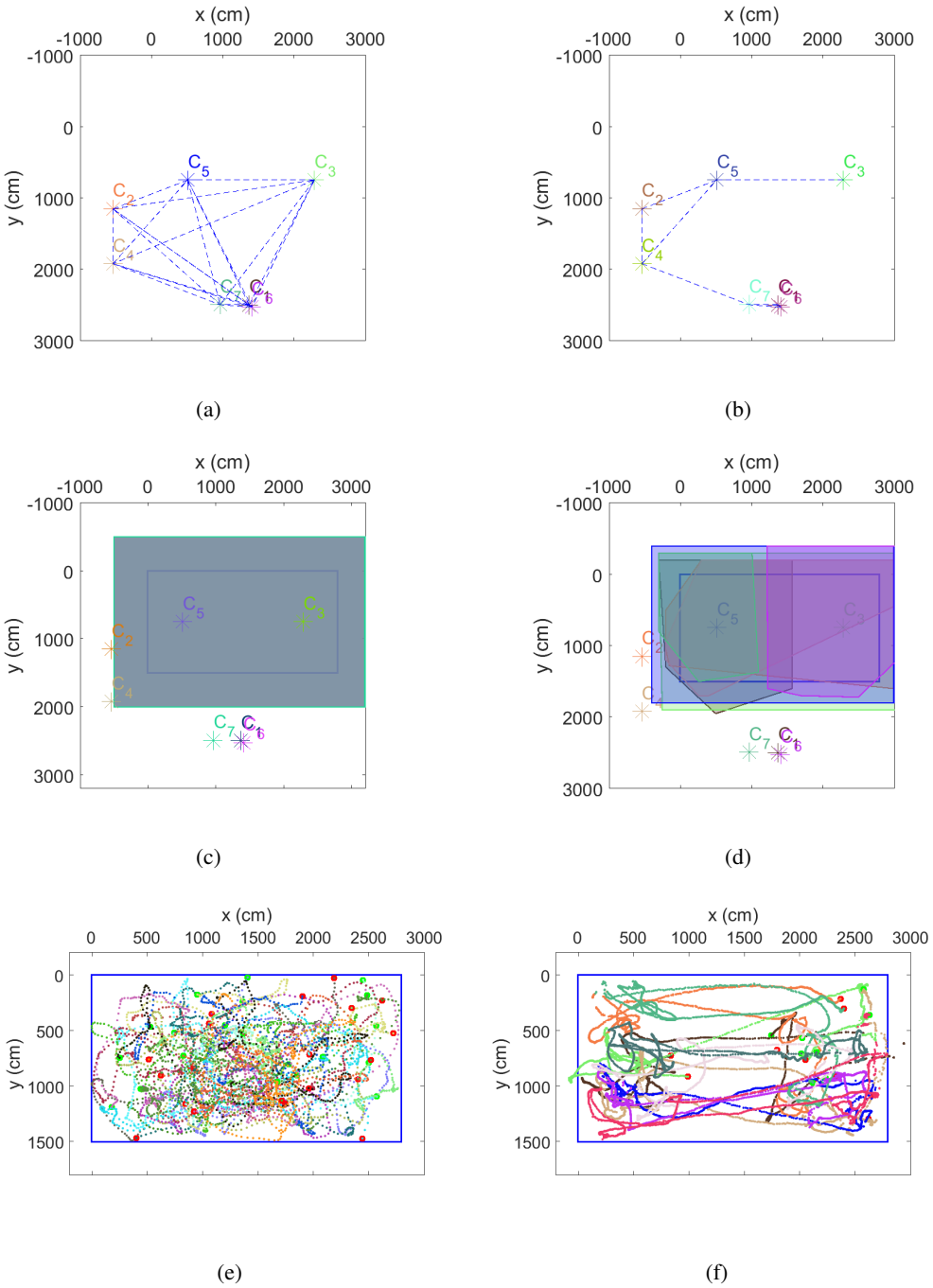


Figure 5.3: Experimental setup. (a) full connectivity, (b) limited connectivity, (b) FoVs of cameras in full observability case, (d) FoVs of cameras in limited observability case (taken from APIDIS), (e) $N_t = 20$ simulated tracks, (f) $N_t = 10$ real tracks (taken from APIDIS). The blue rectangle is the region where the targets move.

full (Figure 5.3(a)) and limited connectivity (Figure 5.3(b)), for full observability of the cameras (Figure 5.3(c)) and limited observability (Figure 5.3(d)), without estimation delays ($\tau_k^i = 0$, $\forall C^i \in \mathbf{C}$) and with random estimation delays ($\tau_k^i \in \{0,1,2,3\}$, $\forall C^i \in \mathbf{C}$), and with known and unknown motion models. Here, $\gamma = 1$ time step, $T = 25$ time steps and one time step $\approx$40ms. Fully

connectivity is generated using the communication range of each node $r_c = 3000cm$ and limited connectivity is generated using $r_c = 1800cm$. The network connectivities are shown using the blue lines in Figures 5.3(a) and 5.3(b). Full observability is generated by assuming that all cameras have the same FoV and it covers beyond the entire court. The FoV is the coloured rectangle in Figure 5.3(c). Limited observability is generated by considering the actual FoVs of APIDIS cameras. The FoVs are the coloured polygons in Figure 5.3(d). To consider trajectories with known motion model, this chapter generates $N_t = 20$ simulated trajectories with a known motion model (Figure 5.3(e)) each 300 time-step long. The considered motion model is the nearly constant velocity model given by (1.2) and (1.3) with $q = 10$. To consider trajectories with unknown motion model, this chapter uses trajectories of $N_t = 10$ players given in the APIDIS dataset each 1500 time-step long (Figure 5.3(f)). The measurement model of each camera C^i follows (1.4). The state to measurement transition function $h^i(\cdot)$ is considered to be linear and is as follows:

$$h^i(\mathbf{x}_k) = \mathbf{H}^i\mathbf{x}_k = [\mathbf{I}_2\ \mathbf{0}_2]\mathbf{x}_k, \tag{5.17}$$

and $\sigma^2 = 60$, i.e the standard deviation of measurement error is $\sigma = 7.7cm$. This chapter analyses the mean tracking error with increasing asynchronism α. To let each camera complete its estimation phase, α should be $\leq T - \tau^{max}$, with $\tau^{max} = \max_{\forall C^i \in \mathbf{C}, \forall k} \left\{\tau_k^i\right\} = 3$ time-steps. If D is the network diameter, aCDTT requires at least D consensus iterations so α should be $\leq T - \tau^{max} - D$. For the limited connectivity (Figure 5.3(b)), $D = 4$ so this chapter chooses $\alpha \in [0, 18]$. This chapter tracks each player $t \in [1, N_t]$ separately using $M = 10$ Monte-Carlo simulations. Each simulation uses a different set of estimation delays and measurements. The mean tracking error (MTE), defined as the mean of the N_t root mean square errors, is considered as the performance measure and is as follows:

$$\text{MTE} = \frac{1}{N_t}\sum_{t=1}^{N_t}\sqrt{\frac{1}{MN}\sum_{r=1}^{M}\sum_{i=1}^{N}\frac{1}{|\mathcal{K}^{i,r,t}|}\sum_{\forall k \in \mathcal{K}^{i,r,t}}||\hat{\mathbf{x}}_k^i(r,t) - \mathbf{x}_k^t||_2^2}. \tag{5.18}$$

Here, $\hat{\mathbf{x}}_k^i(r,t)$ is the estimated location of player t by camera C^i at k during the r^{th} run, $\mathbf{x}_k^t$ is the corresponding ground truth location and $\mathcal{K}^{i,r,t}$ be the set of capturing instants of C^i during r^{th} run of tracking target t.

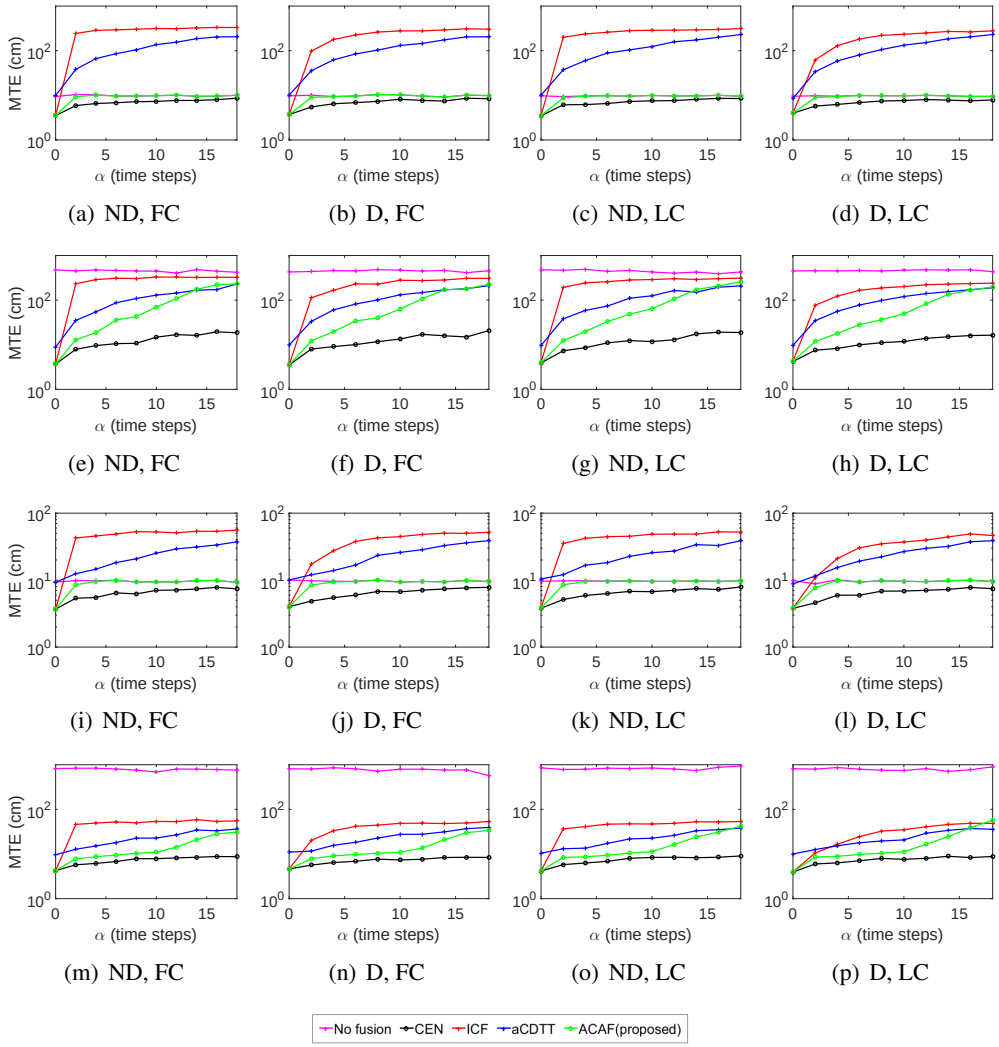


Figure 5.4: Mean tracking error (MTE) with increasing asynchronism α. (a)-(h) Results with simulated tracks. (i)-(p) Results with APIDIS tracks. (a)-(d) and (i)-(l) Results with full observability. (e)-(h) and (m)-(p) Results with limited observability. KEY – D: delay, ND: no delay, FC: full connectivity, LC: limited connectivity. The compared algorithms are the distributed filter that does not fuse (No fusion), the centralised filter (CEN), the Information Consensus Filter (ICF) [6], the asynchronous Consensus-based Distributed Target Tracking method (aCDTT) [33] and the proposed Average Consensus-based Asynchronous Filter (ACAF).

5.3.2 Results

The tracking error increases as the asynchronism increases irrespective of the delays, observability and connectivity (Figure 5.4). In the synchronous case ($\alpha = 0$), the accuracy of ACAF is equivalent to ICF irrespective of the delays, observability and connectivity. In the asynchronous case ($\alpha > 0$), ACAF achieves better tracking accuracy than aCDTT and ICF irrespective of the delays, observability and connectivity. The tracking error of ACAF is upper bounded by the tracking error of the distributed filtering that does not perform fusion. This is because ICF fuses

the information without information alignment. Moreover, there is a risk of fusing the information corresponding to subsequent frames. aCDTT does not perform fusion at all. In addition, aCDTT assigns a local estimate corresponding to a time instant to different other time instants. In the case of full observability, it is better to avoid fusion instead of using ICF and aCDTT irrespective of the delays and connectivity (Figure 5.4(a)- 5.4(d), 5.4(i)- 5.4(l)). This is because ICF fuses asynchronous information without information alignment and aCDTT assigns highly certain information all the times. Both worsen the accuracy. In the case of limited observability, nodes that cannot view the target predict the target information. If there is no fusion, the nodes cannot correct their predicted estimates and result in maximum tracking error irrespective of delays and connectivity (Figure 5.4(e)- 5.4(h), 5.4(m)- 5.4(p)). When asynchronism is high, the tracking error of ACAF increases significantly. This is because the higher the asynchronism, the lower the duration of the fusion phase, thus leading to an insufficient number of consensus iterations for convergence.

5.4 Summary

This chapter proposed an Average Consensus-based Asynchronous tracking Filter which can deal with asynchronous capture and delayed processing that are typical in WCNs. ACAF temporally aligns the data via predictions before fusion using the known states corresponding to the reception instants. Each camera predicts the target information of other cameras at its capturing instant. The proposed method achieves better tracking accuracy and uses less communication bandwidth than state-of-the-art methods in the asynchronous case. The state of the art methods include a maximum consensus-based approach aCDTT and an average consensus-based approach ICF. aCDTT does not perform fusion and selects the estimate with less uncertainty among the asynchronous estimates as the consensus estimate. ICF performs fusion but does not handle the asynchronism. The results show that it is better to avoid fusion instead of fusing asynchronous estimates in the case of full observability. In the case of limited observability, the non-viewing nodes have predicted estimates so fusion is necessary to correct the predictions. ACAF performs fusion using average consensus and handles asynchronism via temporal information alignment. Irrespective of the network connectivity, observability and presence of delays ACAF has always better tracking accuracy compared to aCDTT and ICF.

Chapter 6

Batch Asynchronous Filter[1]

This chapter proposes the Batch Asynchronous Filter (BAF), a distributed fusion scheme for fully connected wireless cameras, that handles non-linearity, benightedness and asynchronism. The chapter assumes that each camera has the knowledge of the maximum relative offset $\alpha^{max} = \max_{i,j}\{\alpha^{ij}\}$, and the maximum and minimum processing delays $\tau^{max} = \max_{i,k}\{\tau_k^i\}$ and $\tau^{min} = \min_{i,k}\{\tau_k^i\}$. The chapter first presents the Bayesian formulation of the idea and then present its implementation under Gaussian assumptions using the IF.

6.1 Bayesian asynchronous fusion

In the *estimation phase*, C^i runs a local Bayesian filter to compute the target probability density function (pdf) $p(\mathbf{x}_k^i|\mathbf{z}_{1:k}^i)$ using the known pdf $p(\hat{\mathbf{x}}_{k'}^i|\mathbf{z}_{1:k'}^i)$ corresponding to its previous capturing instant k', the state transition pdf $p(\mathbf{x}_k^i|\mathbf{x}_{k'}^i)$ and the likelihood function $p(\mathbf{z}_k^i|\mathbf{x}_k^i)$. As the local pdfs are asynchronous, the filter performs two prediction operations, one before the transmission and one after the reception, to temporally align the pdfs. Information alignment is the process of finding the target pdfs of all cameras corresponding to the same instant which is the capturing instant of the fusing camera. If target pdf of a camera corresponds to a different time step, the pdf corresponding to capturing instant is computed by predicting based on its latest pdf that contains target latest location and velocity.

[1]This chapter is completely taken from [C1].

The node C^i predicts the pdf at the transmission instant $k+\tau_k^i$ (first prediction) as

$$p(\tilde{\mathbf{x}}^i_{k+\tau_k^i}|\mathbf{z}^i_{1:k}) = \int p(\mathbf{x}^i_{k+\tau_k^i}|\mathbf{x}^i_k)p(\mathbf{x}^i_k|\mathbf{z}^i_{1:k})d\mathbf{x}^i_k. \tag{6.1}$$

The predicted pdf represents the opinion of C^i about the target state at the transmission instant. The node C^i broadcasts the predicted pdf $p(\tilde{\mathbf{x}}^i_{k+\tau_k^i}|\mathbf{z}^i_{1:k})$ to all other nodes $C^j \in \mathbf{C}\backslash\left\{C^i\right\}$. C^i receives similar pdfs from other nodes. Figure 6.1 shows the timeline of the proposed approach.

In the *fusion phase*, the cameras perform a batch fusion in which each camera fuses the received pdfs from other cameras to update its local pdf. The filter defines a time window $\mathbf{K}^i_k = [k-\alpha_1, k+\alpha_2]$ where $\alpha_1 = \alpha^{max} - \tau^{min}$ and $\alpha_2 = \alpha^{max} + \tau^{max}$. This is because $k-\alpha_1$ and $k+\alpha_2$ are the earliest and the latest possible reception instants respectively. Hence, the camera C^i enters the fusion phase at $k+\alpha_2$ and considers only the pdfs received in the time window $\mathbf{K}^i_k$ for fusion. Let $k'' \in \mathbf{K}^i_k$ be a reception instant of pdf $p(\tilde{\mathbf{x}}^j_{k+\tau_k^j}|\mathbf{z}^j_{1:k})$ from C^j. C^i considers the pdf as $p(\tilde{\mathbf{x}}^i_{k''}|\mathbf{z}^j_{1:k})$. C^i predicts the pdf of C^j at its capturing instant k (second prediction) based on the received pdf $p(\tilde{\mathbf{x}}^i_{k''}|\mathbf{z}^j_{1:k})$ as

$$p(\tilde{\mathbf{x}}^i_k|\mathbf{z}^j_{1:k}) = \int p(\mathbf{x}^i_k|\tilde{\mathbf{x}}^i_{k''})p(\tilde{\mathbf{x}}^i_{k''}|\mathbf{z}^j_{1:k})d\tilde{\mathbf{x}}^i_{k''}. \tag{6.2}$$

This predicted pdf $p(\tilde{\mathbf{x}}^i_k|\mathbf{z}^j_{1:k})$ is considered as the opinion of C^j at capturing instant k of C^i. As the local pdf $p(\mathbf{x}^i_k|\mathbf{z}^i_{1:k})$ and the predicted pdfs $p(\tilde{\mathbf{x}}^i_k|\mathbf{z}^j_{1:k}), \forall C^j \in \mathbf{C}\backslash\left\{C^i\right\}$ correspond to the same time instant (capturing instant k of C^i), C^i now fuses the pdfs by computing their Kullback-Leibler Average (KLA) [6] as

$$p(\hat{\mathbf{x}}^i_k|\mathbf{z}^i_{1:k}) = \frac{p(\mathbf{x}^i_k|\mathbf{z}^i_{1:k})^{\frac{1}{N}} \prod\limits_{j=1,j\neq i}^{N} p(\tilde{\mathbf{x}}^i_k|\mathbf{z}^j_{1:k})^{\frac{1}{N}}}{\int p(\mathbf{x}^i_k|\mathbf{z}^i_{1:k})^{\frac{1}{N}} \prod\limits_{j=1,j\neq i}^{N} p(\tilde{\mathbf{x}}^i_k|\mathbf{z}^j_{1:k})^{\frac{1}{N}} d\mathbf{x}}. \tag{6.3}$$

Figure 6.2 illustrates the information of the nodes after the information alignment and before the fusion. Note that the duration of the frame capture and the fusion phase are negligible compared to the processing delay τ^i_k.

To avoid fusing information from subsequent estimation phases, the consequent captures

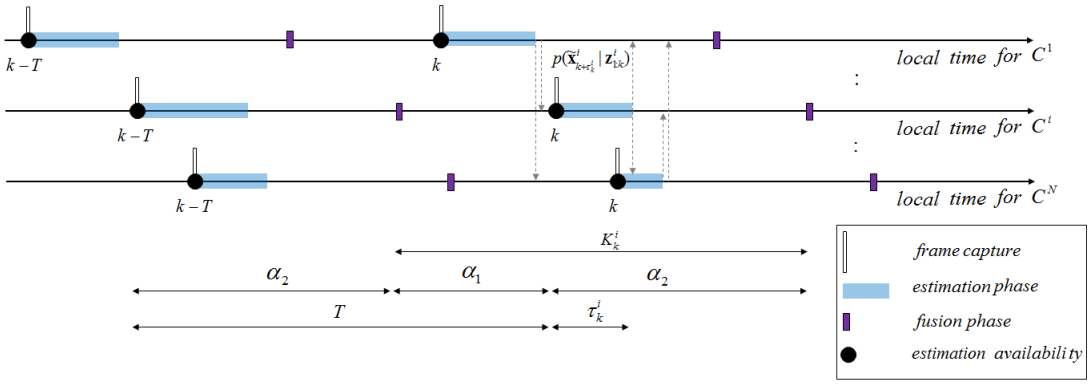


Figure 6.1: The timeline of the batch asynchronous fusion. KEY – C^i: camera i, τ_k^i: processing delay of C^i at k, τ^{max}, τ^{min}: maximum, minimum processing delays in the network, α^{max}: maximum relative offset, $\alpha_1 = \alpha^{max} - \tau^{min}$, $\alpha_2 = \alpha^{max} + \tau^{max}$, $T = \alpha_1 + \alpha_2$: inter-capturing period, $\mathbf{K}_k^i = [k-\alpha_1, k+\alpha_2]$: time window corresponding to the capturing instant k of C^i, $p(\tilde{\mathbf{x}}_{k+\tau_k^i}^i | \mathbf{z}_{1:k}^i)$: transmitted target pdf of C^i. The grey arrows indicate the communication.

must be well separated. To achieve this, the cameras capture their next frames after

$$T = \underbrace{(\alpha^{max} - \tau^{min})}_{\alpha_1} + \underbrace{(\alpha^{max} + \tau^{max})}_{\alpha_2}. \tag{6.4}$$

Note that the inter-capturing period increases with the relative offsets and the processing delays. If τ^{min}, τ^{max} and α^{max} are unknown, selection of α_1 and α_2 creates trade-off between accuracy and inter-capturing period.

Each node C^i can compute the minimum mean square error estimate of the pdf $\hat{\mathbf{x}}_k^i$ and the corresponding error covariance $\hat{\mathbf{P}}_k^i$ as

$$\begin{aligned} \hat{\mathbf{x}}_k^i &= \int_{-\infty}^{\infty} \mathbf{x}_k^i p(\hat{\mathbf{x}}_k^i | \mathbf{z}_{1:k}^i) d\mathbf{x}_k^i, \\ \hat{\mathbf{P}}_k^i &= \int_{-\infty}^{\infty} (\mathbf{x}_k^i - \hat{\mathbf{x}}_k^i)(\mathbf{x}_k^i - \hat{\mathbf{x}}_k^i)^T p(\hat{\mathbf{x}}_k^i | \mathbf{z}_{1:k}^i) d\mathbf{x}_k^i. \end{aligned} \tag{6.5}$$

The cameras compute the image plane location $\hat{\mathbf{z}}_k^i$ of the target using the $\hat{\mathbf{x}}_k^i$ and the known calibration data.

6.2 Information Filter based batch asynchronous fusion

Based on the Bayesian asynchronous fusion described above, the *Batch Asynchronous Filter* (BAF), is derived for a linear and Gaussian system using the Information filter. The information

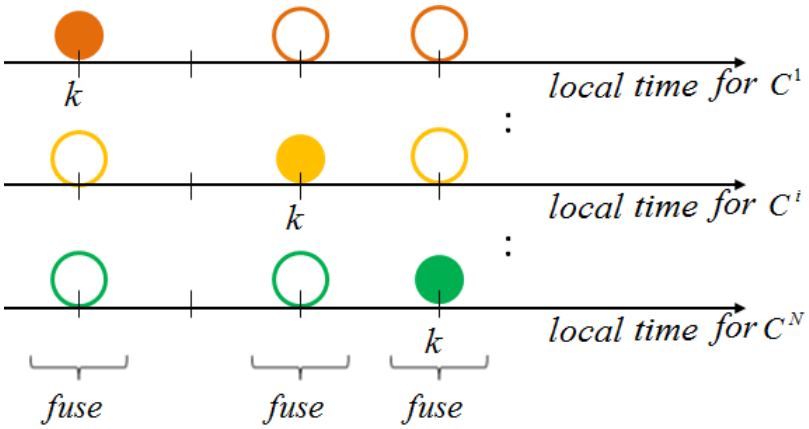


Figure 6.2: Camera information flow with the proposed batch-based asynchronous fusion. Filled circles indicate local pdf of C^i at its capturing instant k ($p(\mathbf{x}_k^i|\mathbf{z}_{1:k}^i)$). Hollow circles with the same colour indicate the opinion of C^i at the capturing instants of other cameras. Hollow circles in the same column indicate the opinion of other cameras $C^j \in \mathbf{C}\backslash C^i$ (the corresponding predicted pdfs) at the capturing instant k of C^i ($p(\tilde{\mathbf{x}}_k^i|\mathbf{z}_{1:k}^j)$). The nodes fuse the temporally aligned information, i.e. column wise.

form of the estimate $(\hat{\mathbf{x}}_k^i, \hat{\mathbf{P}}_k^i)$ is

$$\begin{aligned} \hat{\mathbf{y}}_k^i &= \hat{\mathbf{P}}_k^{i\,-1} \hat{\mathbf{x}}_k^i, \\ \hat{\mathbf{Y}}_k^i &= \hat{\mathbf{P}}_k^{i\,-1}. \end{aligned} \tag{6.6}$$

The Information filter assumes that the target follows a constant velocity model given by (1.2) and (1.3) with $q = 10$.

In the estimation phase, each camera C^i computes the information pair $(\mathbf{y}_k^i(i), \mathbf{Y}_k^i(i))$ using the Information Filter [15] that uses the previous known information pair $(\hat{\mathbf{y}}_{k-T}^i, \hat{\mathbf{Y}}_{k-T}^i)$ and the current local measurement $\mathbf{z}_k^i$. C^i predicts the target information at the transmission instant $k+\tau_k^i$ as

$$\begin{aligned} \tilde{\mathbf{Y}}_{k+\tau_k^i}^i(i) &= (\mathbf{F}(k,k+\tau_k^i)\mathbf{Y}_k^i(i)^{-1}\mathbf{F}(k,k+\tau_k^i)^\top + \mathbf{Q}(k,k+\tau_k^i))^{-1}, \\ \tilde{\mathbf{y}}_{k+\tau_k^i}^i(i) &= \tilde{\mathbf{Y}}_{k+\tau_k^i}^i \mathbf{F}(k,k+\tau_k^i) \left(\mathbf{Y}_k^i(i)^{-1}\mathbf{y}_k^i(i)\right). \end{aligned} \tag{6.7}$$

Then, C^i sends the predicted information $\left(\tilde{\mathbf{y}}_{k+\tau_k^i}^i(i), \tilde{\mathbf{Y}}_{k+\tau_k^i}^i(i)\right)$ to all cameras at $k+\tau_k^i$.

Let $k'' \in \mathbf{K}_k^i$ be the time instant when C^i received the information pair $\left(\tilde{\mathbf{y}}_{k+\tau_k^i}^j(j), \tilde{\mathbf{Y}}_{k+\tau_k^i}^j(j)\right)$ from C^j. C^i considers the information as $\left(\tilde{\mathbf{y}}_{k''}^i(j), \tilde{\mathbf{Y}}_{k''}^i(j)\right)$.

In the fusion phase (at $k+\alpha_2$), C^i predicts the information of C^j, $\forall C^j \in \mathbf{C}$, corresponding to

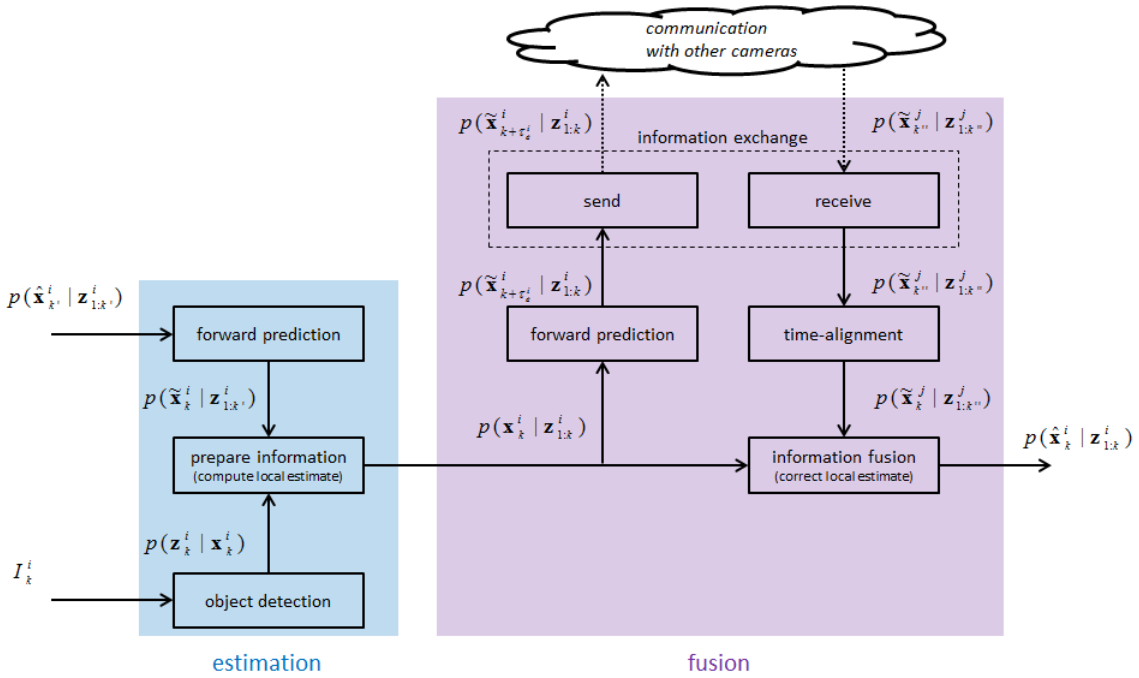


Figure 6.3: Block diagram of BAF running in each camera C^i. Key: $p(\hat{\mathbf{x}}^i_{k'}|\mathbf{z}^i_{1:k'})$ - posterior pdf of C^i corresponding to k'. $\mathcal{I}^i_k$ - frame captured by C^i at k. $p(\mathbf{x}^i_{k'}|\mathbf{z}^i_{1:k'})$ - local posterior pdf of C^i corresponding to k. $p(\hat{\mathbf{x}}^i_k|\mathbf{z}^i_{1:k})$ - posterior pdf of C^i corresponding to k after fusion phase. τ^i_k - processing delay of the frame captured by C^i at k.

its capturing instant k as

$$\begin{aligned}\tilde{\mathbf{Y}}^i_k(j) &= \left(\mathbf{F}(k'',k)\tilde{\mathbf{Y}}^i_{k''}(j)^{-1}\mathbf{F}(k'',k)^\top + \mathbf{Q}(k'',k)\right)^{-1},\\ \tilde{\mathbf{y}}^i_k(j) &= \tilde{\mathbf{Y}}^i_k(j)\mathbf{F}(k'',k)\left(\tilde{\mathbf{Y}}^i_{k''}(j)^{-1}\tilde{\mathbf{y}}^i_{k''}(j)\right).\end{aligned} \tag{6.8}$$

The KLA of the Gaussian pdfs can indeed be obtained by the average of the information terms [6]. Hence, the next step is for C^i to update its local information corresponding to its capturing instant k as

$$\begin{aligned}\hat{\mathbf{Y}}^i_k &= \frac{1}{N}\left(\mathbf{Y}^i_k(i) + \sum_{j=1,j\neq i}^{N}\tilde{\mathbf{Y}}^i_k(j)\right),\\ \hat{\mathbf{y}}^i_k &= \frac{1}{N}\left(\mathbf{y}^i_k(i) + \sum_{j=1,j\neq i}^{N}\tilde{\mathbf{y}}^i_k(j)\right).\end{aligned} \tag{6.9}$$

Figure 6.3 shows the block diagram of BAF.

Each camera knows it local processing delay τ^i_k. If this knowledge is transmitted to the other nodes, they can know when the sender has captured the frame. By exploiting this information, this chapter also proposes an alternative that transmits the local information $\left(\mathbf{y}^i_k(i), \mathbf{Y}^i_k(i)\right)$ to all cameras at $k+\tau^i_k$ without prediction and also the local processing delay τ^i_k.

Let $k'' \in \mathbf{K}_k^i$ be the time instant when C^i received the information pair $\left(\mathbf{y}_k^j(j), \mathbf{Y}_k^j(j)\right)$ and τ_k^j from C^j. As the filter assumes no communication delay, C^i considers the information as $\left(\tilde{\mathbf{y}}^i_{k''-\tau_k^j}(j), \tilde{\mathbf{Y}}^i_{k''-\tau_k^j}(j)\right)$.

In the fusion phase (at $k+\alpha_2$), C^i predicts the information of C^j, $\forall C^j \in \mathbf{C}$, corresponding to its capturing instant k as

$$\begin{aligned}
\tilde{\mathbf{Y}}_k^i(j) &= \left(\mathbf{F}(k''-\tau_k^j,k)\tilde{\mathbf{Y}}^i_{k''-\tau_k^j}(j)^{-1}\mathbf{F}(k''-\tau_k^j,k)^\top + \mathbf{Q}(k''-\tau_k^j,k)\right)^{-1}, \\
\tilde{\mathbf{y}}_k^i(j) &= \tilde{\mathbf{Y}}_k^i(j)\mathbf{F}(k''-\tau_k^j,k)\left(\tilde{\mathbf{Y}}^i_{k''-\tau_k^j}(j)^{-1}\tilde{\mathbf{y}}^i_{k''-\tau_k^j}(j)\right).
\end{aligned} \tag{6.10}$$

The next step is for C^i to update its local information corresponding to its capturing instant k as

$$\begin{aligned}
\hat{\mathbf{Y}}_k^i &= \frac{1}{N}\left(\mathbf{Y}_k^i(i) + \sum_{j=1, j\neq i}^{N} \tilde{\mathbf{Y}}_k^i(j)\right), \\
\hat{\mathbf{y}}_k^i &= \frac{1}{N}\left(\mathbf{y}_k^i(i) + \sum_{j=1, j\neq i}^{N} \tilde{\mathbf{y}}_k^i(j)\right).
\end{aligned} \tag{6.11}$$

The KLA of the Gaussian pdfs can indeed be obtained by the average of the information terms [6]. Note that due to the additional transmission of local delay knowledge the communication cost is 1 scalar higher compared to the previous approach. As this approach avoids one prediction, the estimation accuracy is higher. Figure 6.4 shows the block diagram of BAF with delay transmission.

The comparisons of BAF with the sequential methods SAF [92] and SAF-ED [40] are as follows. In SAF, the nodes exchange measurements and are aware of the measurement models of all the nodes. In BAF and SAF-ED, the nodes exchange their local estimates so the nodes need not know the measurement models of all the nodes. This provides network scalability. In SAF and SAF-ED, the global estimate of a node corresponds to the instant of the last measurement reception. In BAF, the global estimate of a node corresponds to its capturing instant. Unlike SAF, BAF and SAF-ED consider processing delays.

6.3 Simulations

6.3.1 Setup

This chapter compares the proposed Batch Asynchronous Filter (BAF) with six methods. They are the Maximum Consensus-based Asynchronous Filter (MCAF) [33], the Average Consensus-

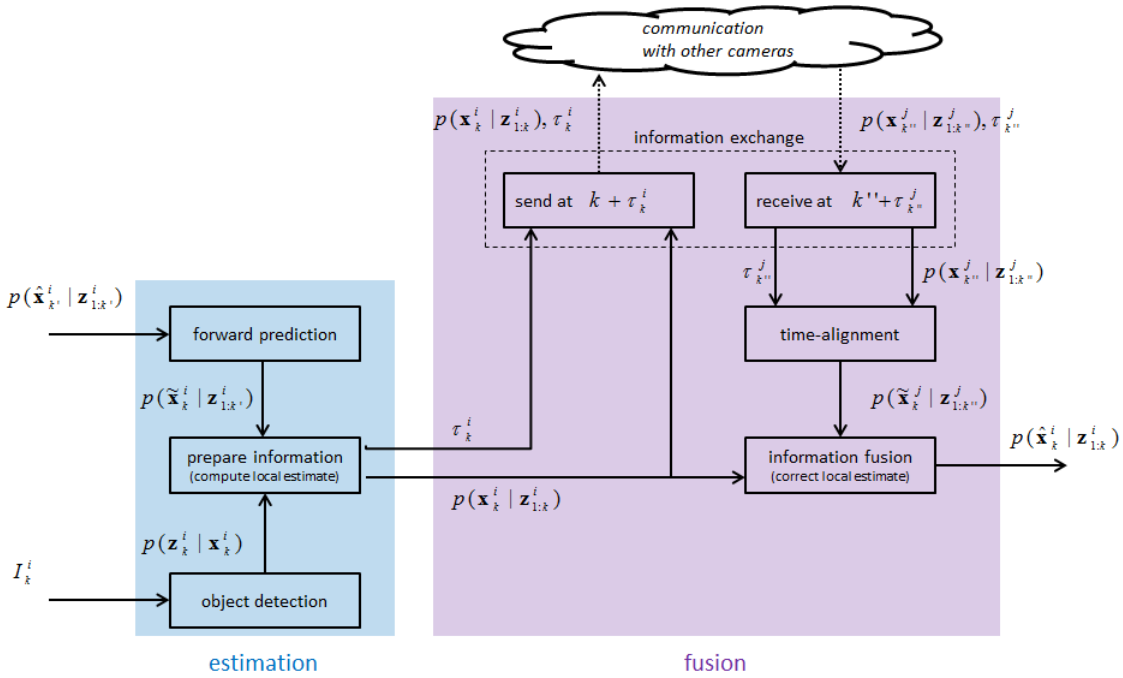
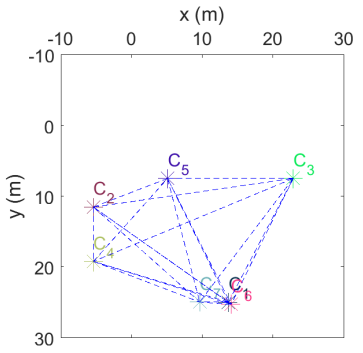


Figure 6.4: Block diagram of BAF (with delay transmission) running in each camera C^i. Key: $p(\hat{\mathbf{x}}^i_{k'}|\mathbf{z}^i_{1:k'})$ - posterior pdf of C^i corresponding to k'. $\mathcal{I}^i_k$ - frame captured by C^i at k. $p(\mathbf{x}^i_{k'}|\mathbf{z}^i_{1:k'})$ - local posterior pdf of C^i corresponding to k. $p(\hat{\mathbf{x}}^i_k|\mathbf{z}^i_{1:k})$ - posterior pdf of C^i corresponding to k after fusion phase. τ^i_k - processing delay of the frame captured by C^i at k.

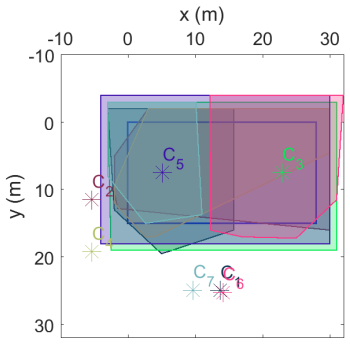
based Asynchronous Filter (ACAF) [C2], the Sequential Asynchronous Filter that does not consider delays (SAF) [92], the Sequential Asynchronous Filter with estimated delays (SAF-ED) [40], the Batch Asynchronous Filter (BAF) [9] and the distributed filter without fusion (No fusion).

For a fair comparison, this chapter uses the same inter-capturing period T, defined by (6.4), for all the algorithms. SAF-ED uses the ground truth delay knowledge for the estimated delays. This chapter uses the APIDIS dataset [1], which is captured with $N = 7$ cameras whose positions and FoVs are known. The FoVs are limited (Figure 6.5(b)) and only a subset of the cameras can view a target at a given time. The camera network monitors a $30m \times 20m$ basketball court. This chapter considers the trajectories of $N_t = 10$ players whose ground plane ground truth is known for a duration of 1500 time steps (Figure 6.5(c)). Each time step corresponds to $40ms$. This chapter simulates the asynchronous captures and processing delays by skipping some frames. As the motion models of the targets are unknown, this chapter uses the nearly constant velocity model given by (1.2) and (1.3) with $q = 10$.

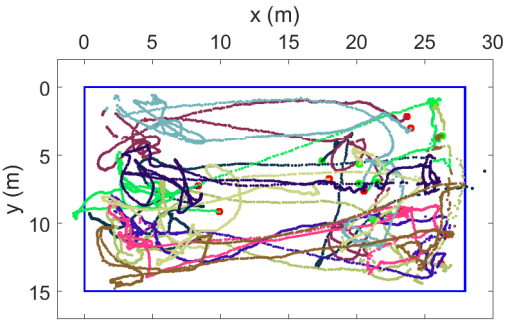
The measurement model of each camera C^i follows (1.4). The state to measurement transition



(a) Fully connected $N = 7$ cameras



(b) Positions and field of views



(c) $N_t = 10$ APIDIS Trajectories

Figure 6.5: Illustration of the validation setup in terms of camera viewpoints and target trajectory on a common ground plane.

function $h^i(\cdot)$ is considered to be linear and is as follows:

$$h^i(\mathbf{x}_k) = \mathbf{H}^i \mathbf{x}_k = [\mathbf{I}_2 \ \mathbf{0}_2]\mathbf{x}_k, \tag{6.12}$$

and $\sigma^2 = 60$, i.e the standard deviation of measurement error is $\sigma = 7.7$ pixels.

The random processing delays of each camera are

$$\tau_k^i \in \mathcal{U}\left\{\tau^{min}, \tau^{max}\right\}, \ \forall C^i \in \mathbf{C}. \tag{6.13}$$

Similarly, the random relative offsets among camera C^i and C^j are:

$$\alpha^{ij} \in \mathcal{U}\left\{0, \alpha^{max}\right\}, \ \forall C^i, C^j \in \mathbf{C}, i \neq j. \tag{6.14}$$

This chapter uses $\tau^{min} = 0$ and single iteration of information exchange per measurement.

Table 6.1: Comparison of tracking errors on the image plane for the distributed filter that never fuses (No fusion), the Maximum Consensus-based Asynchronous Filter (MCAF) [33], the Average Consensus-based Asynchronous Filter (ACAF) [C2], the Sequential Asynchronous Kalman Filter (SAF) [92], the Sequential Asynchronous Filter with estimated delays (SAF-ED) [40] and the proposed Batch Asynchronous Filter (BAF). The data refer to the error in pixels when $\tau^{max} = 4$ and $\alpha^{max} = 4$ using $M = 100$ Monte-Carlo runs.

Algorithm	**mean**	**std**	**max**	**min**
No fusion	20.6	17.2	247.9	0.016
MCAF	20.6	17.2	247.9	0.016
ACAF	20.4	17.0	234.9	0.012
SAF	16.2	13.7	242.7	**0.002**
SAF-ED	13.8	11.9	178.4	0.005
BAF (proposed)	**12.1**	**9.9**	**127.4**	**0.002**

Assuming the inter-camera target association information to be known, this chapter tracks the $N_t = 10$ players using $M = 10$ independent Monte-Carlo simulation runs. Each run uses a different set of processing delays $\{\tau_k^i\}$, relative offsets α^{ij} and measurements $\{\mathbf{z}_k^i\}$.

Let $\mathcal{K}^{i,r} \subset [1,1500]$ be the set of capturing instants of C^i during r^{th} run and $\mathcal{P}_k^i \subset [1,N_t]$ be the set of players in the FoV of C^i at k. This chapter defines as performance measure the *average root mean square error* (ε) of N_t players' locations on the image planes of the cameras:

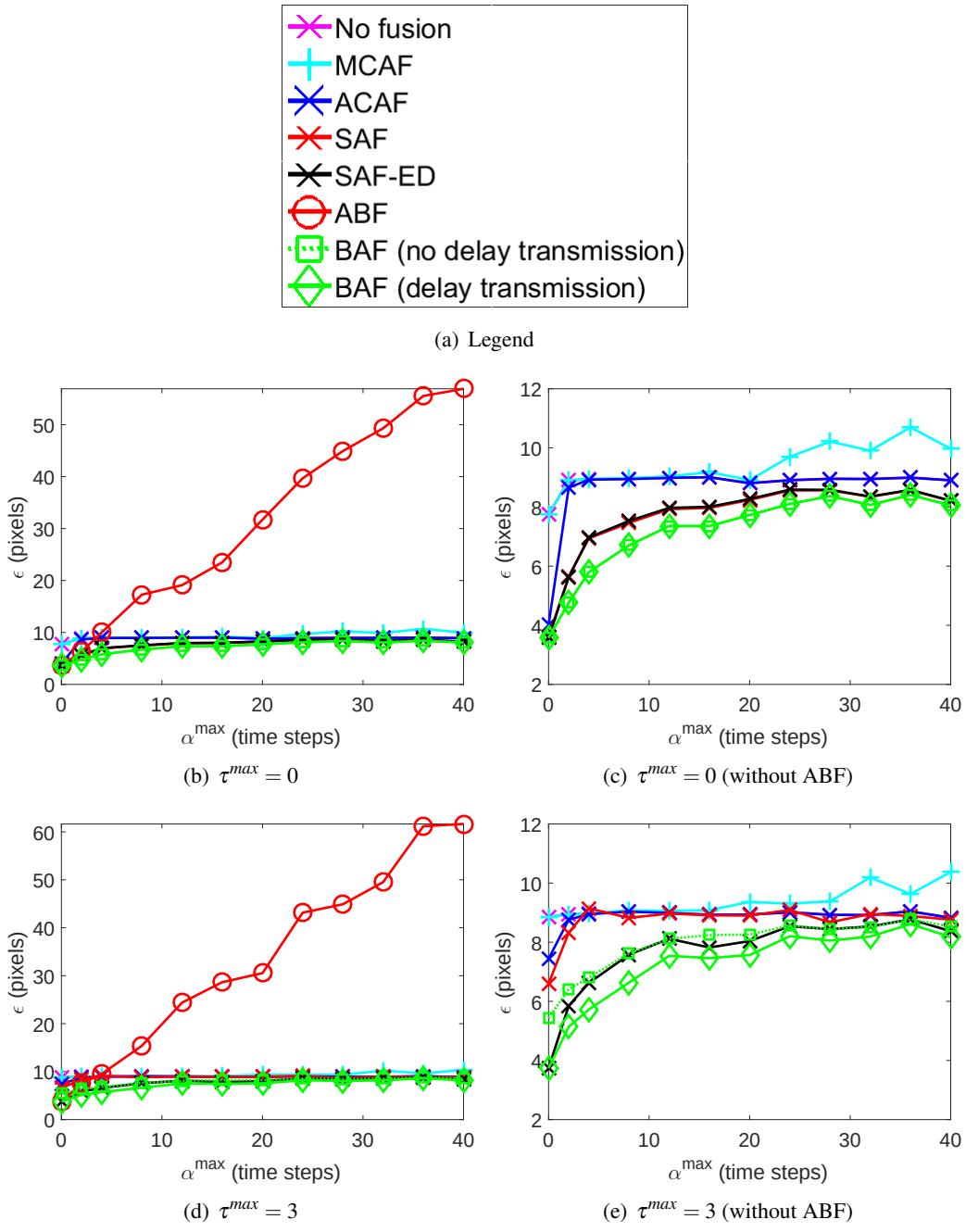
$$\varepsilon = \sqrt{\frac{1}{MN}\sum_{r=1}^{M}\sum_{i=1}^{N}\frac{1}{|\mathcal{K}^{i,r}|}\sum_{\forall k\in\mathcal{K}^{i,r}}\frac{1}{|\mathcal{P}_k^i|}\sum_{\forall t\in\mathcal{P}_k^i}||\hat{\mathbf{z}}_k^i(r,t)-\zeta_k^{i,t}||_2^2}. \tag{6.15}$$

Here, $\hat{\mathbf{z}}_k^i(r,t)$ is the estimated location of player t in the image plane of camera C^i at k during the r^{th} run and $\zeta_k^{i,t}$ is the corresponding image plane ground truth. This chapter analyses the error ε with increasing level of relative offset α^{max} for fixed processing delays. This chapter considers with zero delay ($\tau^{max} = 0$) and with non-zero delay $\tau^{max} = 3$. This chapter also analyses with increasing level of processing delay τ^{max} for fixed relative offset levels. This chapter considers a synchronous clocks case $\alpha^{max} = 0$ and an asynchronous clocks case $\alpha^{max} = 6$.

6.3.2 Results

Without processing delays ($\tau^{max} = 0$) and asynchronism ($\alpha^{max} = 0$), this chapter identifies two groups with similar performance: BAF and SAF-ED; and ACAF and SAF (Figure 6.6(b), 6.7(b)).

In the asynchronous case ($\alpha^{max} > 0$) with ($\tau^{max} > 0$) and without ($\tau^{max} = 0$) processing delays, BAF outperforms all other methods irrespective of the delays (Figure 6.6(b), 6.7(d) and Figure 6.6(d), 6.7(d)). Unlike BAF, the sequential methods (SAF and SAF-ED) do not correct



(a) Legend

(b) $\tau^{max} = 0$

(c) $\tau^{max} = 0$ (without ABF)

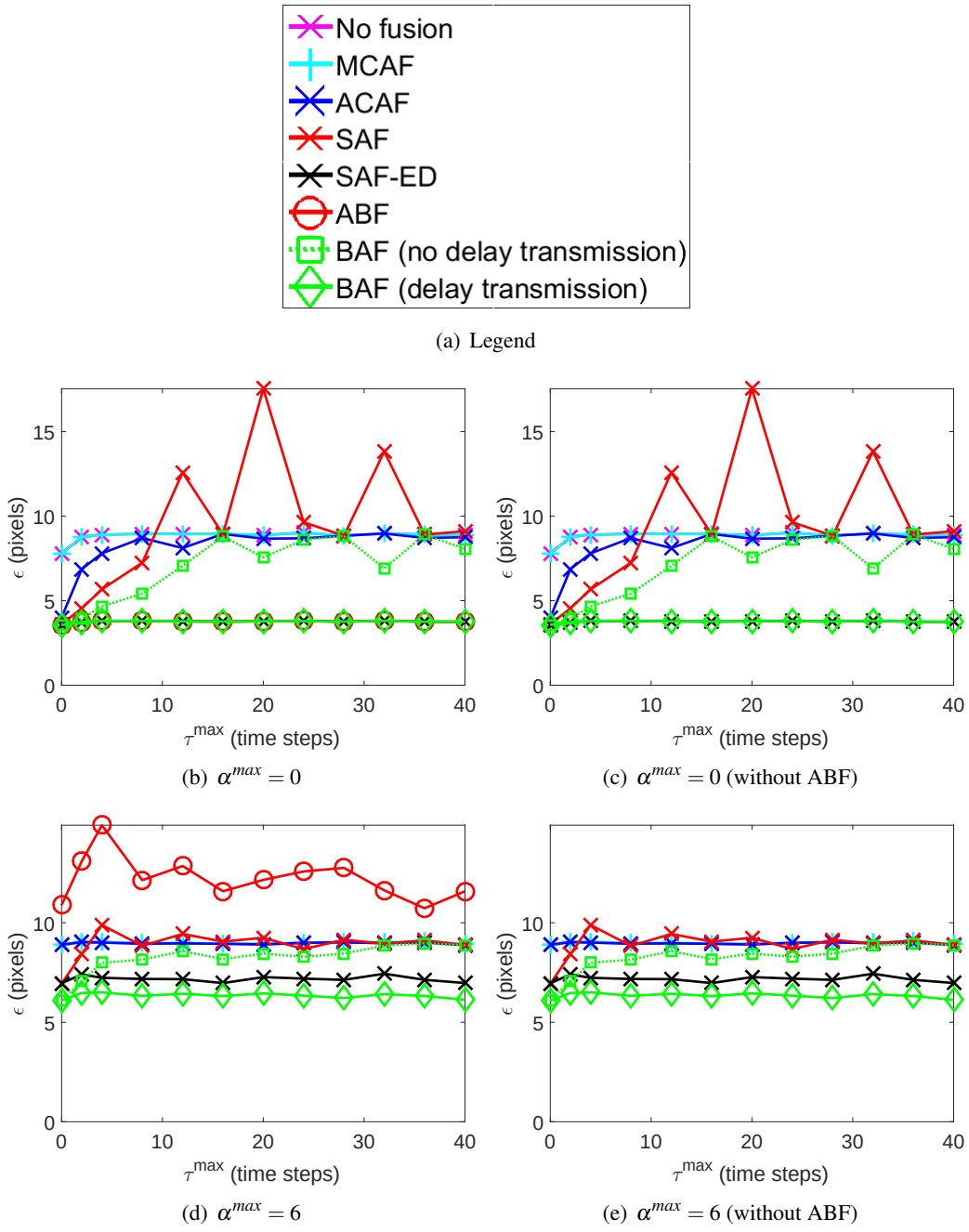
(d) $\tau^{max} = 3$

(e) $\tau^{max} = 3$ (without ABF)

Figure 6.6: Average Root Mean Square tracking Error (ε) with varying degrees of relative offset α^{max} and processing delay τ^{max}. The algorithms under analysis are the Maximum Consensus-based Asynchronous Filter (MCAF) [33], the Average Consensus-based Asynchronous Filter (ACAF) [C2], the Sequential Asynchronous Kalman Filter (SAF) [92], the Sequential Asynchronous Filter with estimated delays (SAF-ED) [40], the Asynchronous Batch Filter (ABF), the proposed Batch Asynchronous Filter (BAF) [9] and the distributed filter without fusion (No fusion).

the estimates corresponding to the capturing instants using the delayed information.

In the synchronous case ($\alpha^{max} = 0$) with processing delays ($\tau^{max} > 0$), BAF outperforms all other methods except SAF-ED (Figure 6.7(b)). This is because the accuracy of the KLA



(a) Legend

(b) $\alpha^{max} = 0$

(c) $\alpha^{max} = 0$ (without ABF)

(d) $\alpha^{max} = 6$

(e) $\alpha^{max} = 6$ (without ABF)

Figure 6.7: Average Root Mean Square tracking Error (ε) with varying degrees of relative offset α^{max} and processing delay τ^{max}. The algorithms under analysis are the Maximum Consensus-based Asynchronous Filter (MCAF) [33], the Average Consensus-based Asynchronous Filter (ACAF) [C2], the Sequential Asynchronous Kalman Filter (SAF) [92], the Sequential Asynchronous Filter with estimated delays (SAF-ED) [40], the Asynchronous Batch Filter (ABF) [9], the proposed Batch Asynchronous Filter (BAF) and the distributed filter without fusion (No fusion).

estimate decreases if the uncertainty of the pdfs increase. In BAF, the information of other cameras undergoes two predictions thereby reducing the uncertainty. One by the sender C^j from its capturing instant k to its transmission instant $k+\tau_k^j$ and the second my the receiver C^i from

the reception instant $k+\tau_k^j$ to its capturing instant k. In SAF-ED, based on the known delay, the receiving node realises that the captures are synchronous so there is no prediction just the fusion. However, the performance of SAF-ED depends on the correctness of the estimated delays. If a node has no view of the target because of its limited FoV, in ACAF and MCAF the node sends predicted target information. However, in SAF, SAF-ED and BAF, nodes send information only when they have the target observations, otherwise, they do not send any information. This avoids the presence of redundant priors and hence the tracking accuracy of SAF, SAF-ED and BAF is not affected by the limited FoVs. In the case of synchronous captures 6.7(b), the performance of ABF and BAF is the same.

In all the asynchronous cases ($\alpha^{max} > 0$), BAF outperforms ACAF and MCAF, because in ACAF the nodes do not consider the information received before their capturing instants and in MCAF nodes do not fuse the information but select the most certain information that might correspond to a different time step. In the case of asynchronous captures 6.6(b) and 6.6(d), ABF has the highest error among all because ABF fuses asynchronous information without temporal alignment.

In the case of delays ($\tau^{max} > 0$), it is better to skip fusion instead of using MCAF (Figure 6.6 (b)-(e)). Note that the errors are always upper-bounded by the error with no fusion.

As in camera networks processing delays are always present, BAF is a good choice when the network is fully connected.

When the network is with limited connectivity ACAF is the best choice but in ACAF the time window of the nodes start from their capturing instants so the information received prior to the capture is discarded. To handle the problem, in ACAF each node sends its information multiple times between consequent captures such that the neighbours receive the information after their capturing instants. This increases not only the communication cost but also the inter-capturing period.

Table 6.1 show the results of the specific case ($M = 100$ runs) when both processing delays and relative offsets are present, and $\tau^{min} = 0$, $\tau^{max} = 4$, $\alpha^{max} = 4$. The inter-capturing period is $T = 12$. The proposed approach has the smallest error in terms of mean, standard deviation, maximum and minimum values. The errors are high in the captured frames if the targets change their direction in the 12 time steps time window of the capturing instant. The prediction of target information is done based on the previously known target velocity. If the target changes

its direction during the time window, considering the velocity corresponding to the previous capturing instant reduces the prediction accuracy and thereby the tracking accuracy. Hence, the errors are high if the target changes direction in the window.

6.4 Summary

This chapter presented a batch method for asynchronous tracking in fully connected WCNs where each camera predicts the target information of other cameras to temporally align it with its local information. The cameras consider and process only the information received in time windows around their local capturing instants. Importantly the proposed method considers the information received both before and after its capturing instants. Moreover, the proposed method corrects the local estimates with respect to their capturing instants by fusing the temporally aligned information. The inter-capturing period is affected by the upper bounds of the relative offsets and the processing delay. In asynchronous cases, the proposed tracking approach outperforms state-of-the-art methods in terms of image-plane error at the cost of additional communication of local processing delay knowledge. The state of the art algorithms include two sequential methods, two consensus methods and a batch method. The sequential methods do not correct the local estimates based on the received information. The consensus methods require multiple iterations of information exchange to converge to the KLA of the temporally aligned local estimates. The state of the art batch method does not consider processing delays and assumes that the received target information corresponds to the reception instant. Hence, as the asynchronism increases the tracking accuracy of the state of the art batch method decreases.

Chapter 7

Conclusions

7.1 Summary of achievements

This thesis focussed on distributed tracking in wireless camera networks (WCNs) and addressed the challenges of non-linearity, limited connectivity, limited observability that cause benightedness, redundant information in the network, and varying and unknown processing delays that produce asynchronous information. The thesis proposed five distributed tracking algorithms for WCNs, namely the Extended Information Consensus Filter (EICF), the Extended Information Weighted Consensus Filter (EIWCF), Neighbour consensus filter (N-consensus), the Average Consensus-based Asynchronous Filter (ACAF) and the Batch Asynchronous Filter (BAF). Each filter handles a subset of the challenges. EICF, EIWCF, N-consensus and ACAF use consensus framework so they work with any type of network connectivity whereas BAF works only with fully connected networks. In all the algorithms, each node iteratively performs two phases, namely estimation and fusion. The estimation phase computes the measurement (pixel coordinates of the target) from the captured frame. The Information Filter (IF) or the Extended Information Filter (EIF) estimates the target state and the corresponding error covariance (that represents the uncertainty of the state estimate) using the measurement and the previous estimate. The fusion phase performs information exchange and information fusion.

EICF performs average consensus-based (A-consensus) fusion. EICF weights the estimates of each node based on its uncertainty and solves the problem of benightedness. Nodes viewing the target get higher weights as their error covariance is lower than that of non-viewing

nodes. This concept is called weighted averaging [71, 84] and is used in ICF [15]. EICF handles non-linearity by using the EIF for local state estimation. The concept of weighted averaging is applied to EIF to handle benightedness along with non-linearity. The redundant prior knowledge is handled by weighting the redundant information less than the obtained measurement information [50]. The EIWCF is also an A-consensus approach that extends EICF by adding such weighting. EICF solves the problems of non-linearity and benightedness without requiring the knowledge of the number of nodes in the network. However, EIWCF requires the knowledge of the number because the redundancy is proportional to the number of nodes in the network. When the number of nodes is available, EIWCF can be used to solve the redundancy problem along with non-linearity and benightedness at almost the same communication and computation cost as EICF. Both algorithms use EIF as the underlying filter to deal with non-linearity and employ A-consensus as the fusion scheme to support the limited network connectivity. Experimental results show that the two algorithms outperform the accuracy of the Extended Kalman Consensus Filter (EKCF) (that do not perform information weighting) at the expense of higher communication and computational costs, especially when more consensus iterations are used.

A-consensus such as EICF and EIWCF approaches require the knowledge of the maximum degree of the network to achieve convergence. Moreover, all the nodes in the network participate in consensus. N-consensus dynamically identifies a reduced set of nodes in the neighbourhood of the viewing nodes and achieves consensus only among these nodes. The set includes all nodes that are viewing the target at the current time step and the nodes that might view the target at the next time step are included. N-consensus uses covariance information and achieves better consensus estimates in the selected subnetwork. To handle non-linearities in the system, the thesis employed the EIF at each node for local state information. In addition to handling the benightedness and non-linearity, the N-consensus reduces the number of nodes involved in the consensus process, thereby reducing the energy consumption of the network without compromising on the tracking accuracy.

ACAF is an A-consensus approach that handles asynchronous measurements. In ACAF nodes perform information-alignment with respect to their capturing instants by predicting the information of their neighbours. The information of the sending node is predicted by a receiving node with a time-reversed operation. The nodes discard the information received before the frame capture and terminate their fusion phase before any other node captures the next frame.

The termination criterion is decided based on its local processing time.

BAF tackles the problem of target tracking accounting for processing delays in fully connected WCNs when the cameras exchange data asynchronously. Similar to ACAF, in BAF, the nodes predict the target state estimates corresponding to the end time instant of their estimation phases and the predicted local estimates are transmitted to all other nodes. If a node receives the target state information from any node, it stores the value in a buffer along with the reception time. A time window is defined around the capturing instant. At the end of the window, the node enters the fusion phase. In the fusion phase, based on the received information in the window, the node predicts the target state estimate corresponding to its capturing instant. This predicted information is considered as the opinion of the sender about the target. After predicting the information of all the senders, the node fuses the temporally aligned local target information and the predicted target information of the senders and updates its estimate corresponding to the capturing instant. The thesis used IF in both ACAF and BAF so they do not handle non-linearity.

EICF and EIWCF are preferred in the case of small WCNs where the diameter of the network is in the order of 10 hops. Note that if the number of cameras is not available, EIWCF can not be applied EICF is the only option. N-consensus is preferred in the case of large WCNs where the network diameter is in the order of more than or equal to 100 hops where many cameras do not need the target information all the time. If the processing delays are so high that the cameras might not capture synchronously, ACAF and BAF should be used instead of EICF, EIWCF and N-consensus. If the network is fully connected, BAF produces better accuracy than all the algorithms. In the case of limited network connectivity, BAF cannot be used whereas others can be used and ACAF produces better tracking accuracy than EICF, EIWCF and N-consensus.

7.2 Future work

ACAF and BAF can be used in the non-linear systems by replacing the IF with the EIF. All the proposed algorithms can be extended by using other advanced non-linear filters such as UKF [45], CubKF [3] and PF [24].

The optimal set of future viewing nodes includes only the nodes having overlapping FoVs with all the current viewing nodes. The proposed N-consensus considers more nodes as future viewing nodes than the optimal case because the selection process considered only the viewing

range but not the viewing direction. As a future work, N-Nodes could be selected based on a distributively computed vision graph [28] to generate the optimal to save resources. Vision graph contains all the cameras as vertices. There exists an edge between two cameras only if their FoVs are overlapped. They are called vision neighbours. If the target is in the FoV of a camera, it is possible that it will enter the FoV of the camera with overlapping FoV so the vision neighbours of the current viewing nodes are the only future viewing nodes. Another possible improvement is to replace the hop distances by relative physical distances between nodes. The distances can be computed based on the signal strength of the messages received [91]. With this approach, the value of threshold becomes a physical distance which is twice the visual sensing range. The proposal distances are computed by incrementing by the relative physical distance between the sender and the receiver computed using received signal strength (instead of incrementing by one). As the threshold is independent of communication range, the algorithm will support heterogeneous communication ranges.

The thesis assumed that there are no communication delays in the channel. In reality, there could be significant communication delays (comparable to processing delays). Future work could involve modelling and estimating the delays to aid predictions in batch and consensus frameworks. This was already done in a sequential framework [40]. Moreover, packet losses in wireless channel are common. Future work could study the impact of packet losses on distributed target tracking.

The work of this thesis covers single target tracking and assumed that there are no false positive detections. Handling false detections and multiple targets can be done using data association techniques. The Joint Probabilistic Data Association-based distributed target tracking algorithm [47] performs intra-camera association to deal with multiple targets and false detections. However, it does not perform inter-camera association and does not handle asynchronism. Moreover, in reality, the number of targets is unknown and also varying so a mechanism to support dynamic track initialisation and termination should be added. Future work could focus on these limitations.

Bibliography

[1] Basketball dataset from the European project APIDIS. `http://sites.uclouvain.be/ispgroup/index.php/Softwares/APIDIS`, 2009. Last accessed: 2017-03-16.

[2] B. D. O. Anderson and J. B. Moore. *Optimal Filtering*. Prentice-Hall, 1979.

[3] I. Arasaratnam and S. Haykin. Cubature Kalman filters. *IEEE Trans. on Automatic Control*, 54(6):1254–1269, Jun 2009.

[4] N. Assimakis, M. Adam, and A. Douladiris. Information filter and Kalman filter comparison: selection of the faster filter. *Int. Jour. of Information Engineering*, 2(1):1–5, 2012.

[5] T. C. Aysal, M. E. Yildiz, A. D. Sarwate, and A. Scaglione. Broadcast gossip algorithms for consensus. *IEEE Trans. on Signal Processing*, 57(7):2748–2761, 2009.

[6] G. Battistelli and L. Chisci. Kullback-leibler average, consensus on probability densities, and distributed state estimation with guaranteed stability. *Automatica*, 50(3):707–718, Mar 2014.

[7] G. Battistelli, L. Chisci, and C. Fantacci. Parallel consensus on likelihoods and priors for networked nonlinear filtering. *IEEE Signal Processing Let.*, 21(7):787–791, Jul 2014.

[8] G. Battistelli, L. Chisci, G. Mugnai, A. Farina, and A. Graziano. Consensus-based linear and nonlinear filtering. *IEEE Trans. on Automatic Control*, 60(5):1410–1415, May 2015.

[9] J. Beaudeau, M. F. Bugallo, and P. M. Djurić. Target tracking with asynchronous measurements by a network of distributed mobile agents. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pages 3857–3860, Mar 2012.

[10] F. Bénézit, V. Blondel, P. Thiran, J. Tsitsiklis, and M. Vetterli. Weighted gossip: Distributed averaging using non-doubly stochastic matrices. In *Proc. of the IEEE Int. Symp. on Information Theory*, pages 1753–1757, Jun 2010.

[11] V. P. Bhuvana, M. Schranz, M. Huemer, and B. Rinner. Distributed object tracking based on cubature Kalman filter. In *Proc. of the Asilomar Conf. on Signals, Systems and Computers*, pages 423–427, Pacific Grove, CA, USA, 2013.

[12] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Trans. on Information Theory*, 52(6):2508–2530, 2006.

[13] R. Carli, G. Notarstefano, L. Schenato, and D. Varagnolo. Distributed quadratic programming under asynchronous and lossy communications via newton-raphson consensus. In *Proc. of the European Control Conf.*, pages 2514–2520, Jul 2015.

[14] A. Carron, M. Todescato, R. Carli, and L. Schenato. An asynchronous consensus-based algorithm for estimation from noisy relative measurements. *IEEE Trans. on Control of Network Systems*, 1(3):283–295, Sep 2014.

[15] D. W. Casbeer and R. Beard. Distributed information filtering using consensus filters. In *Proc. of the American Control Conf.*, pages 1882–1887, Jun 2009.

[16] D. W. Casbeer, Y. Cao, E. Garcia, and D. Milutinovic. Bridge consensus: Ignoring initial inessentials. *CoRR*, abs/1501.02713, 2015.

[17] M. Cetin, L. Chen, J. W. Fisher, A. T. Ihler, R. L. Moses, M. J. Wainwright, and A. S. Willsky. Distributed fusion in sensor networks. *IEEE Signal Processing Mag.*, 23(4):42–55, 2006.

[18] G. Chen, P. L. St-Charles, W. Bouachir, G. A. Bilodeau, and R. Bergevin. Reproducible evaluation of pan-tilt-zoom tracking. In *Proc. of the IEEE Int. Conf. on Image Processing*, pages 2055–2059, Sep 2015.

[19] Y. Chen and Q. Zhao. A novel square-root cubature information weighted consensus filter algorithm for multi-target tracking in distributed camera networks. *Sensors*, 15(5):10526–10546, May 2015.

[20] Z. Chen. Bayesian filtering: From Kalman filters to particle filters, and beyond. Technical report, Adaptive Systems Lab, McMaster University, 2003.

[21] C. Ding, B. Song, A. Morye, J. A. Farrell, and A. K. Roy-Chowdhury. Collaborative sensing in a distributed PTZ camera network. *IEEE Trans. on Image Processing*, 21(7):3282–3295, 2012.

[22] P. M. Djurić, J. Beaudeau, and M. F. Bugallo. Non-centralized target tracking with mobile agents. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pages 5928–5931, May 2011.

[23] P. M. Djurić and L. Geng. Non-centralized target tracking in networks of directional sensors. In *Proc. of the IEEE Aerospace Conf.*, pages 1–6, Mar 2011.

[24] P. M. Djuric, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Miguez. Particle filtering. *IEEE Signal Processing Mag.*, 20(5):19–38, Sep 2003.

[25] S. L. Dockstader and A. M. Tekalp. Multiple camera tracking of interacting and occluded human motion. *Proc. of the IEEE*, 89(10):1441–1455, Oct 2001.

[26] X. Dong and M. C. Vuran. Vision graph construction in wireless multimedia sensor networks. In *Proc. of the IEEE Global Telecommunications Conf.*, Miami, FL, USA, Dec 2010.

[27] J. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. In *Proc. of the Symp. on Operating Systems Design and Implementation*, volume 36, pages 147–163, Dec 2002.

[28] L. Esterle, P. R. Lewis, X. Yao, and B. Rinner. Socio-economic vision graph generation and handover in distributed smart camera networks. *ACM Trans. on Sensor Networks*, 10(2):20:1–20:24, 2014.

[29] L. Fang and P. J. Antsaklis. Information consensus of asynchronous discrete-time multi-agent systems. In *Proc. of the American Control Conf.*, pages 1883–1888, Jun 2005.

[30] S. Farahmand, S. I. Roumeliotis, and G. B. Giannakis. Set-membership constrained particle filter: Distributed adaptation for sensor networks. *IEEE Trans. on Signal Processing*, 59(9):4122–4138, Sep 2011.

[31] D. Franken and A. Hupper. Improved fast covariance intersection for distributed data fusion. In *Proc. of the 8th Int. Conf. on Information Fusion*, Jul 2005.

[32] Á. F. García-Fernández and J. Grajal. Asynchronous particle filter for tracking using non-synchronous sensor networks. *Signal Processing*, 91(10):2304–2313, Oct 2011.

[33] S. Giannini, A. Petitti, D. Di Paola, and A. Rizzo. Asynchronous consensus-based distributed target tracking. In *Proc. of the IEEE Conf. on Decision and Control*, pages 2006–2011, Dec 2013.

[34] R. Goshorn, J. Goshorn, D. Goshorn, and H. Aghajan. Architecture for cluster-based automated surveillance network for detecting and tracking multiple persons. In *Proc. of the ACM/IEEE Int. Conf. on Distributed Smart Cameras*, pages 219–226, 2007.

[35] D. Gu, J. Sun, Z. Hu, and H. Li. Consensus based distributed particle filter in sensor networks. In *Proc. of the Int. Conf. on Information and Automation*, pages 302–307, Changsha, China, 2008.

[36] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[37] O. Hlinka, P. M. Djuric, and F. Hlawatsch. Time-space-sequential distributed particle filtering with low-rate communications. In *Proc. of the Asilomar Conf. on Signals, Systems and Computers*, pages 196–200, 2009.

[38] O. Hlinka and F. Hlawatsch. Time-space-sequential algorithms for distributed Bayesian state estimation in serial sensor networks. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pages 2057–2060, 2009.

[39] O. Hlinka, F. Hlawatsch, and P. M. Djuric. Distributed particle filtering in agent networks: A survey, classification, and comparison. *IEEE Signal Processing Mag.*, 30(1):61–81, 2013.

[40] O. Hlinka, F. Hlawatsch, and P. M. Djurić. Distributed sequential estimation in asynchronous wireless sensor networks. *IEEE Signal Processing Let.*, 22(11):1965–1969, Nov 2015.

[41] O. Hlinka, O. Sluciak, F. Hlawatsch, and M. Rupp. Distributed data fusion using iterative covariance intersection. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pages 1861–1865, Florence, Italy, 2014.

[42] Y. Hu, Z. Duan, and D. Zhou. Estimation fusion with general asynchronous multi-rate sensors. *IEEE Trans. on Aerospace and Electronic Systems*, 46(4):2090–2102, Oct 2010.

[43] B. Jia, K. D. Pham, E. Blasch, D. Shen, Z. Wang, and G. Chen. Cooperative space object tracking using consensus-based filters. In *Proc. of the 17th Int. Conf. on Information Fusion*, pages 1–8, Jul 2014.

[44] S. Joo and Q. Zheng. A temporal variance-based moving target detector. In *Proc. of the IEEE Int. workshop Performance Evaluation of Tracking and Surveillance*, Jan 2005.

[45] S. J. Julier and J. K. Uhlmann. Unscented filtering and nonlinear estimation. *Proc. of the IEEE*, 92(3):401–422, Mar 2004.

[46] R. E. Kalman. A new approach to linear filtering and prediction problems. *Trans. of the ASME Jour. of Basic Engineering*, 82:35–45, 1960.

[47] A. T. Kamal, J. H. Bappy, J. A. Farrell, and A. K. Roy-Chowdhury. Distributed multi-target tracking and data association in vision networks. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 38(7):1397–1410, Jul 2016.

[48] A. T. Kamal, C. Ding, B. Song, J. A. Farrell, and A. K. Roy-Chowdhury. A generalized Kalman consensus filter for wide-area video networks. In *Proc. of IEEE Conf. on Decision and Control and European Control Conf.*, pages 7863–7869, 2011.

[49] A. T. Kamal, J. A. Farrell, and A. K. Roy-Chowdhury. Information consensus for distributed multi-target tracking. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pages 2403–2410, 2013.

[50] A. T. Kamal, J. A. Farrell, and A. K. Roy-Chowdhury. Information weighted consensus filters and their application in distributed camera networks. *IEEE Trans. on Automatic Control*, 58(12):3112–3125, Dec 2013.

[51] M. Kriegleder, R. Oung, and R. D'Andrea. Asynchronous implementation of a distributed average consensus algorithm. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1836–1841, Nov 2013.

[52] W. Li. Camera sensor activation scheme for target tracking in wireless visual sensor networks. *Int. Jour. of Distributed Sensor Networks*, 2013(397537):1–11, 2013.

[53] W. Li and Y. Jia. Distributed consensus filtering for discrete-time nonlinear systems with non-gaussian noise. *Signal Processing*, 92(10):2464–2470, 2012.

[54] G. Liu and G. Tian. Distributed object tracking using a derivative free nonlinear information consensus filter. In *Proc. of IEEE Int. Conf. on Robotics and Biomimetics*, pages 1732–1735, Dec 2016.

[55] T. Matsuyama and N. Ukita. Real-time multitarget tracking by a cooperative distributed vision system. *Proc. of the IEEE*, 90(7):1136–1150, Jul 2002.

[56] H. Medeiros, J. Park, and A. C. Kak. Distributed object tracking using a cluster-based Kalman filter in wireless camera networks. *IEEE Jour. of Selected Topics in Signal Proc.*, 2(4):448–463, 2008.

[57] M. Mehyar, D. Spanos, J. Pongsajapan, S. H. Low, and R. M. Murray. Asynchronous distributed averaging on communication networks. *IEEE/ACM Trans. on Networking*, 15(3):512–520, Jun 2007.

[58] C. Micheloni, G. L. Foresti, and L. Snidaro. A network of co-operative cameras for visual surveillance. *IEE Proc. of the Vision, Image and Signal Processing*, 152(2):205–212, Apr 2005.

[59] A. Mittal and L. Davis. M2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene. 51(3):189–203, 2003.

[60] A. G. O. Mutambara. *Decentralized Estimation and Control for Multisensor Systems*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1998.

[61] R. Nagpal, H. Shrobe, and J. Bachrach. Organizing a global coordinate system from local information on an ad hoc sensor network. In *Proc. of Int. Conf. on Information Processing in Sensor Networks*, pages 333–348, Palo Alto, CA, USA, 2003.

[62] C. Nastasi and A. Cavallaro. Distributed target tracking under realistic network conditions. In *Proc. of the Sensor Signal Processing for Defence*, pages 1–5, 2011.

[63] D. Niculescu and B. Nath. DV based positioning in ad hoc networks. *Telecommunication Systems*, 22(1-4):267–280, 2003.

[64] W. Niehsen. Information fusion based on fast covariance intersection filtering. In *Proc. of the 5th Int. Conf. on Information Fusion*, volume 2, pages 901–904, Annapolis, MD, USA, Jul 2002.

[65] R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multi-agent systems. *Proc. of the IEEE*, 95(1):215–233, 2007.

[66] R. Olfati-Saber and N. F. Sandell. Distributed tracking in sensor networks with limited sensing range. *Proc. of the American Control Conf.*, pages 3157– 3162, Jun 2008.

[67] S. Persa and P. P. Jonker. Real-time image processing architecture for robot vision. *Proc. SPIE, Intelligent Robots and Computer Vision XIX: Algorithms, Techniques, and Active Vision*, 4197:221–228, 2000.

[68] J. C. SanMiguel and A. Cavallaro. Energy consumption models for smart-camera networks. *IEEE Trans. on Circuits and Systems for Video Technology*, 2017 (In press). DOI: 10.1109/TCSVT.2016.2593598.

[69] V. Savic, H. Wymeersch, and S. Zazo. Belief consensus algorithms for fast distributed target tracking in wireless sensor networks. *Signal Processing*, 95:149–160, Feb 2014.

[70] X. Sheng, Y. H. Hu, and P. Ramanathan. Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor network. In *Proc. of the Int. Symp. on Information Processing in Sensor Networks*, pages 181–188, 2005.

[71] A. Simonetto, T. Keviczky, and R. Babuska. Distributed nonlinear estimation for robot localization using weighted consensus. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 3026–3031, 2010.

[72] F. Sivrikaya and B. Yener. Time synchronization in sensor networks: a survey. *IEEE Network*, 18(4):45–50, Jul 2004.

[73] B. Song, A. T. Kamal, C. Soto, C. Ding, J. A. Farrell, and A. K. Roy-Chowdhury. Tracking and activity recognition through consensus in distributed camera networks. *IEEE Trans. on Image Processing*, 19(10):2564–2579, 2010.

[74] S. F. Stefánsson, B. Jónsson, and K. R. Thórisson. A YARP-based architectural framework for robotic vision applications. In *Proc. of the 4th Int. Conf. on Computer Vision Theory and Applications*, pages 65–68, Feb 2009.

[75] T. Sun, M. Xin, and B. Jia. Distributed estimation in general directed sensor networks based on batch covariance intersection. In *Proc of American Control Conf.*, pages 5492–5497, Jul 2016.

[76] M. Taj and A. Cavallaro. Distributed and decentralized multicamera tracking. *IEEE Signal Processing Mag.*, 28(3):46–58, May 2011.

[77] M. Vemula, J. Miguez, and A. Artes-Rodriguez. A sequential monte carlo method for target tracking in an asynchronous wireless sensor network. In *Proc. of the Workshop on Positioning, Navigation and Communication*, pages 49–54, Mar 2007.

[78] A Wang and A Chandrakasan. Energy-efficient DSPs for wireless sensor networks. *IEEE Signal Processing Mag.*, 19(4):68–78, 2002.

[79] S. Wang and W. Ren. On the consistency and confidence of distributed dynamic state estimation in wireless sensor networks. In *Proc. of the 54th IEEE Conf. on Decision and Control*, pages 3069–3074, Dec 2015.

[80] Y. Wang, M. Casares, and S. Velipasalar. Cooperative object tracking and event detection with wireless smart cameras. In *Proc. of the 6th IEEE Int. Conf. on Advanced Video and Signal Based Surveillance*, pages 394–399, Sep 2009.

[81] Y. Wang and A. Cavallaro. Prioritized target tracking with active collaborative cameras. In *Proc. of the IEEE Int. Conf. on Advanced Video and Signal Based Surveillance*, pages 131–137, Aug 2016.

[82] H. D. Whyte. *Multi Sensor Data Fusion*. Technical report, Australian Centre for Field Robotics, The University of Sydney, Australia, 2001.

[83] D. Wu, S. Ci, H. Luo, Y. Ye, and H. Wang. Video surveillance over wireless sensor and actuator networks using active cameras. *IEEE Trans. on Automatic Control*, 56(10):2467–2472, Oct 2011.

[84] L. Xiao. A scheme for robust distributed sensor fusion based on average consensus. In *Proc. of the Int. Conf. on Information Proc. in Sensor Networks*, pages 63–70, Los Angeles, CA, USA, 2005.

[85] L. Xiao and S. Boyd. Fast linear iterations for distributed averaging. *Systems and Control Let.*, 53:65–78, 2004.

[86] G. Xing, R. Tan, B. Liu, J. Wang, X. Jia, and C. Yi. Data fusion improves the coverage of wireless sensor networks. In *Proc. of the Int. Conf. on Mobile Computing and Networking*, pages 157–168, 2009.

[87] F. G. Yap and H. H. Yen. A survey on sensor coverage and visual data capturing/processing/transmission in wireless visual sensor networks. *Sensors*, 14(2), 2014.

[88] A. Yilmaz, Xin Li, and M. Shah. Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(11):1531–1536, Nov 2004.

[89] F. Zanella, D. Varagnolo, A. Cenedese, G. Pillonetto, and L. Schenato. Asynchronous newton-raphson consensus for distributed convex optimization. In *Proc. of the IFAC Workshop on Distributed Estimation and Control in Networked Systems*, volume 45, pages 133 – 138, Sep 2012.

[90] Q. Zhou and J. K. Aggarwal. Object tracking in an outdoor environment using fusion of features and cameras. *Image and Vision Computing*, 24(11):1244–1255, 2006.

[91] Y. Zhou, L. Lamont, and L. Li. Wormhole attack detection based on distance verification and the use of hypothesis testing for wireless ad hoc networks. In *Proc. of the IEEE Military Communications Conf.*, 2009.

[92] G. Zhu, F. Zhou, L. Xie, R. Jiang, and Y. Chen. Sequential asynchronous filters for target tracking in wireless sensor networks. *IEEE Sensors Jour.*, 14(9):3174–3182, Sep 2014.