DISTRIBUTED LOCALIZATION ALGORITHMS FOR TARGET TRACKING IN WIRELESS SENSOR NETWORKS

by

Tolga TOLGAY

B.S, Computer Engineering, Ege University, 2006

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering
Boğaziçi University

2009

ACKNOWLEDGEMENTS

I would like to thank to my thesis supervisor Professor Cem Ersoy for his guidance, motivation and support during this thesis. Without his encouragement and warnings at the right time, I will not be able to realize this thesis.

I am thankful to Tolga Önel, who always helped me during this thesis. His support and patient answers to all of my question helped me to complete this thesis.

I would like to thank to TUBITAK for their support under the grant number 106E082 and BAP for their support under the grant number 09A101P.

Finally, many special thanks are to my friends, my dear parents and my lovely fiancée for their love, help, patience and endless support.

ABSTRACT

DISTRIBUTED LOCALIZATION ALGORITHMS FOR TARGET TRACKING IN WIRELESS SENSOR NETWORKS

A wireless sensor network (WSN) is a group of tiny wireless sensor nodes which perform sensing and sharing this information in collaboration. WSNs have a wide application area including target tracking.

Target tracking is the process of estimating a target's state with the measurements obtained from the target. Target tracking with WSNs involves the design of energy efficient models that can work with a good trade-off between the energy conservation and the tracking quality.

In this thesis, we focused on the localization methods, which are one of the key parts in the target tracking process. Our motivation has been to provide better target tracking quality and localization accuracy while consuming similar or less energy.

Two new localization algorithms are proposed for an existing distributed collaborative target tracking framework. Our algorithms are easily applicable to this framework. We used simulations to evaluate the performance of these algorithms. The performance results shows that our algorithms perform better than existing localization methods under some conditions.

ÖZET

TELSİZ ALGILAYICI AĞLARDA HEDEF TAKİBİ İÇİN DAĞITIK YER BELİRLEME ALGORİTMALARI

Telsiz algılayıcı ağlar, algılama ve işbirliği içerisinde bilgi paylaşımı işlemlerini gerçekleştirebilen küçük algılayıcı düğümlerden oluşur. Hedef takip etmek de dahil olmak üzere telsiz algılayıcı ağların kullanılabildiği geniş bir uygulama alanı vardır.

Hedef takibi, bir hedefe ait ölçümler yapılarak ve bunlar kullanılarak hedefin bir sonraki durumunun tahmin edilmesi sürecidir. Telsiz algılayıcı ağlarda hedef takibi, enerji korunması ve takip kalitesi arasında iyi bir dengede çalışabilecek enerji etkin modelleri içerir.

Bu çalışma kapsamında hedef takibinin en önemli parçalarından biri olan yer belirleme yöntemlerini inceledik. Motivasyonumuz, daha iyi hedef takip kalitesini ve yer belirleme doğruluğunu aynı ya da daha az enerji harcayarak başarmaktır.

Dağıtık hedef takibi için kullanılacak iki yeni yer belirleme algoritması önerdik. Bu algoritmalar var olan yapıya kolayca uygulanabilir. Sistem başarım değerlendirmesinde kullanılan benzetim yöntemleri, önerilen algoritmaların var olan yer belirleme yöntemlerinden belirli durumlarda daha iyi olduğunu göstermektedir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

D	Mahalanobis Distance
E	Euclidian Distance
F	State Transition Matrix
H	Observation Matrix
i	Information State Denomination
I	Information Matrix Denomination
J	Mutual Information Gain
K	Kalman Gain
L	Sensor Position
N_{max}	Number of Sensor Nodes Allowed to Communicate
P	Covariance Matrix of Observation Estimate
Q	Covariance Matrix of State Transition
R	Covariance Matrix of Observation
TGT_{max}	Number of Targets to be Reported by a Sensor Node
v	Observation Noise
w	Process Noise
W	Weight of a Sensor Node
x	True State of Target
$\hat{x}$	Observation Estimate
y	Information State
Y	Information Matrix
α	Alpha-Beta Filter Constant
φ	Observation
ϖ	Information Received from the Network
ξ	X-axis Coordinate
η	Y-axis Coordinate

ARMA	Auto Regressive Moving Average
BLUE	Best Linear Unbiased Estimator
C4ISR	Command, Control, Communications, Computer, Intelligence, Surveillance and Reconnaissance
CCL	Cooperative Center of Mass Localization
CF	Consensus Filter
CIF-MMI	Cooperative Information Filter with Maximum Mutual Information-based Sensor Selection
DKF	Distributed Kalman Filter
EKF	Extended Kalman filter
GMM	Gaussian Mixture Model
ICCL	Information-based Cooperative Center of Mass Localization
ICTP	Information Controlled Transmission Power
IDSQ	Information Driven Sensor Querying
IF	Information Filter
INS	Iterative Node Selection
JDL	Joint Directors of Laboratories
JPDA	Joint Probabilistic Data Association
KCF	Kalman Consensus Filter
KF	Kalman filter
LSR	limited sensing range
MCO-DKF	Mutually Coupled Oscillators-based Distributed Kalman Filter
MEMS	Micro-Electro-Mechanical Systems
MTT	Multi Target Tracking
P2P	Peer-to-peer
P-MAT	Prediction-based Mobility Adaptive Tracking
RLS	Recursive Least Squares
WSN	Wireless Sensor Network

1. INTRODUCTION

Developments in wireless technology and sensor devices gave rise to Wireless Sensor Networks (WSN). A WSN comprises of a number of small, low-cost and low-power nodes, which sense the environment and send this information to a data collection center. Although WSNs were originally motivated by military applications such as battlefield surveillance, they are used in many industrial and civilian application areas, including process monitoring, machine health monitoring, environment and habitat monitoring, healthcare applications, home automation, robotics and traffic control.

Target tracking is one of the most exciting research areas in wireless sensor networks. Target tracking, in other words, the processing of the measurements obtained from a target in order to maintain an estimate of its current state, has major importance in Command, Control, Communications, Computer, Intelligence, Surveillance and Reconnaissance (C4ISR) applications [1]. Target tracking with WSNs involves the design of energy efficient models that can work with a good trade-off between energy conservation and tracking quality.

Multi target tracking (MTT) differs from the single target tracking and the standard estimation problems in terms of measurement origin uncertainty. Measurement to target association techniques in the literature aims to solve this uncertainty problem. Other than measurement to target association, target initiation and target localization problems are also investigated by the researchers under the title MTT using WSN. Target initiation aims to minimize the false alarms. Target localization deals with minimizing the error between the real and the estimated positions of the target.

1.1. Motivation of the Thesis

A fully-distributed collaborative multi-target tracking framework, which handles initiation, association and estimation problems, is proposed in [2]. The target tracking techniques [2] eliminate the need for a central data associator or a central node. This framework brings the concept of local and network filters (estimators). A local filter is the estimator that works individually on each sensor node. A network filter combines the local estimations of the neighboring sensor nodes, which are shared by the neighboring sensor nodes that are selected according to a collaboration logic. In this thesis, we will work on this framework and aim to decrease the estimation errors while consuming similar or less energy.

We implement new local and network filters for better target location estimation. We also focus on the collaboration logic and implement a new collaboration algorithm that can work with all filters. We compare new filters and collaboration algorithms with the existing components in the target tracking framework [2].

Target location estimation is the combination of the prediction of the target location based on its previous state and the location measurement from the target. Network prediction is the prediction of the target location based on the previous network estimation. We will investigate the effect of network prediction for network filters in different scenarios. We will also look at the contribution of the network prediction to the network estimation.

Dynamic filters can be implemented in two ways. Using different filters in different time slots or using different filters for local and network filters. We discuss these methods and compare these two types of dynamic filters with the existing static filters.

1.2. Structure of the Thesis

In the next chapter, foundations of the target tracking problem in using WSNs are discussed.

The fully-distributed collaborative multi-target tracking framework [2] is introduced in Chapter 3. Existing estimators and collaboration logics in this framework are also examined in detail.

Implemented new estimators, sensor selection algorithms for the collaboration logic and new localization algorithms based on these algorithms are described Chapter 4.

Chapter 5 provides a comparison of existing localization methods with the new implemented localization algorithms in terms of the mean error and energy consumption. The effect of new implementations are also investigated in this chapter.

Finally, we will conclude in Chapter 6 and discuss our future works including the methods to implement a dynamic filter.

2. TARGET TRACKING IN WIRELESS SENSOR NETWORKS

2.1. Wireless Sensor Network Foundations

With the beginning of 21st century, developments in wireless technology, electronics and micro-electro-mechanical systems (MEMS) technology made low-power and low-cost sensor nodes possible. Sensor nodes forming the WSN are capable of sensing the environment, processing data and transmitting information in short ranges. WSN nodes, which send their data to a data-sink in a multi-hop manner is shown in Figure 2.1 [3]. Although this configuration is widely used in WSNs, some application can be implemented using a fully distributed architecture which do not have an information center like data-sink.

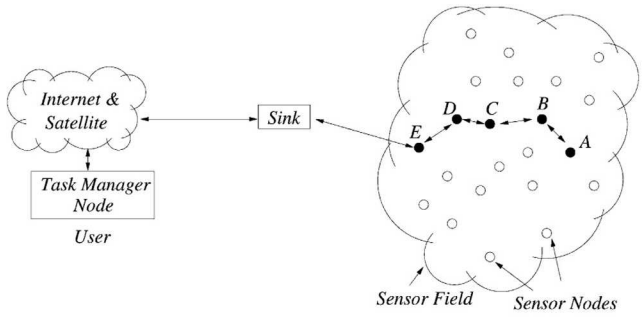

Figure 2.1. Sensor nodes distributed in a field [3]

Realization of the sensor networks require wireless ad-hoc networking techniques. Although there are many protocols and algorithms proposed for traditional wireless ad-hoc networks, they are not applicable to the sensor network applications. There are lots of design issues specific to sensor networks, which are not addressed by wireless ad-hoc networks such as fault tolerance, scalability, production costs, operating environment, topology, hardware constrains, transmission media and power consumption [3]. These design issues guide researchers for developing applications and protocols in WSNs.

In [4], WSN applications are categorized into military, environment, health, home and other commercial areas :

- Military Applications

  The rapid deployment, self-organization and fault tolerance characteristics of sensor networks make them a very promising sensing technique for military C4ISR applications. Some of the military applications of sensor networks are monitoring friendly forces, equipment and ammunition; battlefield surveillance; reconnaissance of opposing forces and terrain; targeting; battle damage assessment; and nuclear, biological and chemical attack detection and reconnaissance.

- Environmental Applications

  Sensors can be used to gather data from environment that will help people to monitor it and pre-inform about natural disasters. Environmental applications include tracking the movements of birds, small animals, and insects; monitoring environmental conditions; Earth monitoring; chemical/biological detection; precision agriculture; biological, Earth, and environmental monitoring in marine, soil, and atmospheric contexts; forest fire detection; meteorological or geophysical research; flood detection; bio-complexity mapping of the environment; and pollution study.

- Health Applications

  Health applications are one of the most interesting application areas of WSNs. These applications aim to provide an easier environment for patient, decrease the risks of faults and improvements on medical science. Health applications, developed using WSNs, are providing interfaces for the disabled; integrated patient monitoring; diagnostics; drug administration in hospitals; monitoring the movements and internal processes of insects or other small animals; telemonitoring of human physiological data; and tracking and monitoring doctors and patients inside a hospital.

- Home Applications

  Home applications aim to provide a smart environment and allow end users to manage home devices locally and remotely more easily. Most common example of these applications is smart houses that monitor all objects in an house such as

vacuum cleaners, micro-wave ovens and refrigerators. Sensor nodes can also interact with outside world via an internet connection to inform the users about any status changes in the house.

- Commercial Applications

  Some of the commercial applications are monitoring material fatigue; building virtual keyboards; managing inventory; monitoring product quality; constructing smart office spaces; environmental control in office buildings; robot control and guidance in automatic manufacturing environments; interactive toys; interactive museums; factory process control and automation; monitoring disaster area; vehicle tracking and detection and detecting and monitoring car thefts.

2.2. Target Tracking Foundations

Target tracking is the process of locating one or more objects using the combination of measured data of these objects and their history. It addresses the problem of combining sensed data and target history to provide accurate and timely knowledge of the location of one or more moving objects [5].

Target tracking is a state estimate problem and its objective is to estimate the target position with the minimum error possible. While trying to minimize the error, it also deals with design challenges such as power consumption, communication and computational complexity. A WSN is ideally suited for tracking moving targets, traversing in a large area with many sensors with its spatial coverage and multiplicity in sensing aspect [6].

In [7], a centralized multi sensor tracking system is used. The advantage of a centralized tracking system is the lower computational load. In centralized networks, all measured data is sent to a central node for processing. This architecture is optimal in terms of decreasing the computation load in the network. However, it has has numerous drawbacks. Sending time series data through the network introduces latency and synchronization issues. It also consumes energy and network bandwidth, while potentially

introducing a single point of failure. Associating sensor readings to tracks suffers from combinatorial explosion when multiple sensors are used [8].

In a distributed target tracking architecture, all sensor nodes are capable of processing the measurement data they have gathered about the targets. After sensor nodes processed their own data, they communicate their results with other nodes. The distributed architecture reduces the communication and eliminates the reliability problem but adds extra computational load to sensor nodes. According to [5], reasons of distributed target tracking architectures to become popular in sensor networks are :

- Large number of sensor nodes can be deployed to achieve a wide area coverage because of the decrease in the cost of sensor nodes.
- Dense sensors enable overlapping coverage, which may result in increased robustness and improved accuracy.
- Diverse sensing modalities provide complementary information; for example, certain types of sensors (e.g., laser range-finders) provide good ranging data, while others (e.g., microphone arrays) provide good directional data, and yet others (e.g., cameras) are ideal for object classification. This diversity in sensing modalities can be exploited to provide accurate and rich information about the target.
- Spatial sensing diversity greatly mitigates the effects of obstructions on line-of-sight sensors.

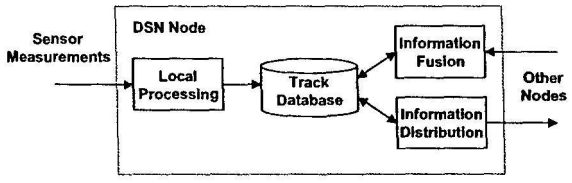

Figure 2.2. Structure of a Distributed Target Tracking Sensor Network Node [9]

A generic distributed tracking architecture is used in [9]. In this architecture, each node has local processing, information fusion and information distribution components as shown in Figure 2.2. Local processing performs local tracking with local sensor data. Information fusion associates the tracks in the incoming information with the local tracks

and updates the state estimation. Information distribution decides when, what and to whom to communicate according to the local information and needs of other nodes.

A distributed target tracking architecture, which performs the tasks of detection, classification and tracking of moving, nonlocal, low observable events, requires nonlocal collaboration among the sensors [6]. Collaboration brings new challenges such as sensor information fusion, communication, sensor management, decision making and sensor selection.

Data fusion is the technique that gather information from multiple resources and combine it in order to achieve better, more efficient and accurate results than a single source. In [10], two applications, self-localization in sensor networks and distributed data association in multiobject tracking, are used to show how sensor network fusion problems can be cast as problems of inference in graphical model.

In [11], multi sensor data fusion architectures are discussed based on JDL (Joint Directors of Laboratories) data fusion process model. The JDL process model is a functionally oriented model of data fusion and is intended to be general and useful across multiple applications. How to distribute the local information is one of the most important implementation criteria in distributed target tracking applications. Three alternatives presented in [11] for fusion of locational information, that includes position and velocity information. These are fusion of raw data, fusion of state vectors and a hybrid approach. Raw data fusion is to distribute the measurements of target location without processing these measurements. This alternative reduces the local computational complexity in a sensor node. Fusion of state vectors is to share information found after processing sensor measurements. However this alternative requires a local computation in sensor node, it ensures to share more reliable information among the network.

Distributed tracking requires the fusion of data among the sensor nodes, which also known as sensor collaboration. In other words, collaboration is to share information

among sensor nodes for the purpose of combining the results of multiple sensors can provide more accurate information than using a single sensor [12]. In sensor collaboration, raw data is not shared among the network. Instead, it is processed and stored locally first, then a smaller part of the processed information is sent to other nodes under certain conditions satisfied.

In [6], the target tracking problem is presented as an information optimization problem. Authors applied this approach to sensor querying and data routing. The idea behind an information-driven approach is to base the sensor collaboration decision on information gain as well as cost and resource consumption. The information gain is represented in Figure 2.3. Dashed ellipsoids represent the updated beliefs after the measurements from either S_1 or S_2 are joined. The solid ellipsoid represents the current belief. Although both S_1 and S_2 reduce the same amount of uncertain area, S_2 has longer principal area in the uncertainty distribution. Therefore S_1 is selected over S_2.

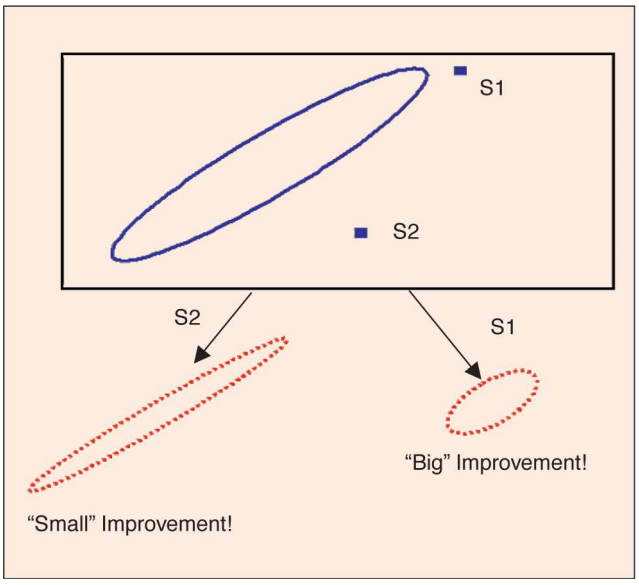

Figure 2.3. Sensor Selection Based on Information Gain [6]

Authors proposed an Information Driven Sensor Querying (IDSQ) tracking algorithm that decides on the sensor nodes which will participate actively in the tracking task. Mahalanobis distance is presented as a suitable measurement for sensor selection

algorithm in information driven architecture as it takes the correlations of the information into account.

Cluster based distributed target tracking solutions are proposed in [13], [14], [15] and [16] for the target tracking problem rather than fully distributed solutions. In [13], a cluster-based approach for collaborative single target tracking is proposed to work in dense wireless networks. Sensor nodes monitor and track the target collectively for detection, clustering and localization of the target. An energy-based target localization algorithm is used to estimate the target's position. Sensor nodes that will participate in information fusion about the target is selected based on the received signal level, the energy level, the sensing range and detected information about the target.

A cluster-based target predicting strategy is proposed in [14]. In this strategy, all active sensor nodes sends their information to the leader of the cluster. Information received by leader is combined by using centroid localization algorithm. Centroid localization algorithm takes the average of cartesian coordinate information received from sensor nodes. Then a prediction algorithm, named Recursive Least Squares (RLS), is used. The prediction result is used to select the new cluster leader and active nodes.

In [15] and [16], Balasubramanian et al. also proposed an energy-aware collaborative target tracking algorithm, which is a combination of Kalman filter and energy-based collaboration algorithm. Distributed Kalman filter runs only on high-end (data gathering) nodes, which are also known as cluster heads. All active sensor nodes share their observations if they detect the target. When the leadership is passed from one cluster to another cluster, the Kalman filter at the next high-end sensor node is initialized with the predicted state obtained from the previous high-end sensor. The data gathering node chooses the set of active sensors at each time instant t so that the total energy spent by the whole network is minimized with the help of energy-based collaboration algorithm. The energy spent by a node is calculated by a mixture of the sensing energy, which depends on the distance between the target and the sensor node, communication energy,

which depends on the distance between sensor node and the data gathering node, and the transmit power.

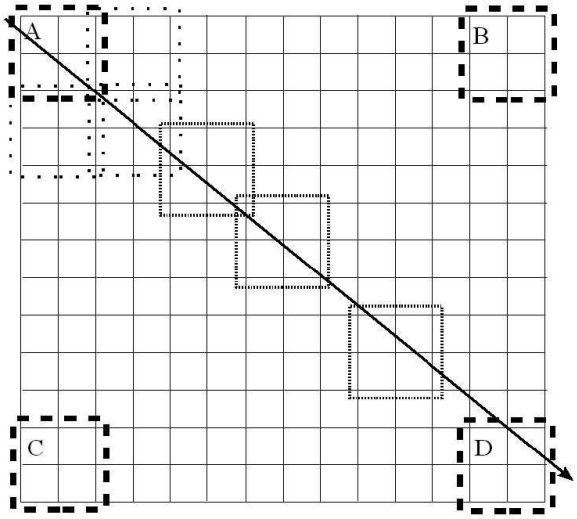

Figure 2.4. Schematic Illustration of Single Target Tracking [17]

Similar to cluster based target tracking, a cell based energy efficient target tracking algorithm is proposed in [17]. Li et al. divided the sensing area into geographic locations, called cells, and predicted the target motion from one cell to another as shown in Figure 2.4. They used an energy-based procedure, shown in Figure 2.5 for estimating the target location by using energy measurements from multiple (4 or more) nodes at a particular instant.

In [18], Zoghi and Kahaei proposed a modified version of *Iterative Node Selection (INS)* algorithm for sensor selection in target tracking applications. INS algorithm, which is proposed in [19], determines the location and the number of active sensor nodes in order to maintain the required distortion in the sink. Authors indicated that the target position is not taken into consideration in this algorithm. They proposed the *Modified INS (MINS)* to resolve this problem by considering target coordinates as one of the codewords in the codebook.

1. Nodes in Cell A are active nodes and Cell A is the active cell. The active nodes report their energy detector outputs to the manager nodes at N successive time instants.
2. At each time instant, the manager nodes determine the location of the target from the energy detector outputs of the active nodes.
3. The manager nodes use locations of the target at the N successive time instants to predict the location of the target at M ($<$ N) future time instants.
4. New cells are created using the target's predicted position if the target is likely to enter. Also a subset of nodes in these cells are activated.
5. Once the target is detected in one of the new cells, it becomes the new active cell and the nodes in the previous active cell may be put in the standby state to conserve energy.

Figure 2.5. Target Tracking Algorithm in [17]

In [20], Zongi and Kahaei proposed two more sensor selection methods, named neighbor selection and information selection, for fully-decentralized data fusion process. Neighbor selection selects the closest sensor nodes to the target. Information selection selects the most informative sensor nodes.

In [21], Pahalawatta et al. developed an optimal sensor selection for the target tracking application in a wireless sensor network using imaging sensors. They formulated the sensor selection problem as maximizing the information utility gained from a set of sensor nodes. They used unscented Kalman Filter for tracking and data fusion. Their sensor selection method, which also takes the initialization time required for motion segmentation by imaging sensor nodes into account, uses a tree-pruning algorithm. In this method, processing node calculates the current target state, and hands over the processing to the next node that will be closest to the target.

Arroyo-Valles et al. used a selective transmission policy for distributed target tracking applications to increase network lifetime without having a significant reduction in the tracking accuracy in [22]. Sensor nodes calculate an importance value of every measure. They gave the decision of transmitting the measure or not using the importance value,

the energy available at the sensor node, the energy cost of transmission and receiving information.

An important part of data fusion is the combination of shared information. Some filters like information filter have their defined processes for the combination of information received from network. A way to combine the information received from network is weighted centroid combination, which is used in [23] not for target localization but for sensor node localization in WSNs.

A target tracking system, based on the auto regressive moving average (ARMA) model in a distributed peer-to-peer (P2P) signal processing framework, is proposed and used in [24], [25], [26] and [27]. Wang et al. indicate that they they have adopted ARMA model to their framework due to its outstanding performance in model fitting and forecasting and its light computational cost. ARMA model is a widely-used model for the prediction of future values. It contains two terms, the auto regressive (AR) term and the moving average (MA) term. The AR term is a linear regression of the current value against one or more prior values. It captures the dependency of the current value and its nearest prior values. The MA term is introduced to capture the influence of random shocks to the future.

In the framework, proposed and used in [24–27], wireless sensor nodes act as peers that perform target detection, feature extraction, classification and tracking, whereas target localization requires the collaboration between wireless sensor nodes for improving the accuracy and robustness. Proposed sensor selection method is a similar one with [6], which tries to balance the information utility and the energy consumption. Information utility is calculated by a mixture of the following facts: Predicted contribution, environment, such as obstacles and noise, and confidence degree, measured by the contribution of the wireless sensor node in previous instants.

In [28], Li et al. proposed a target localization algorithm based on overlap area

boundary of sensor detection (OABSD). This algorithm can localize a target in a boundary of overlap sensing area of neighboring sensor nodes without time synchronization. It establishes a dynamic discovery queue of sensor nodes by using the continuity of target's trajectory. Using this queue, the target's possible position can be bounded to a certain overlap region of those sensor nodes in the discovery queue. This leads a better localization accuracy than traditional centroid localization algorithms.

A new recursive filter, based on best linear unbiased estimator (BLUE), is proposed in [29]. Zhao et al. aimed to implement the recursive BLUE for two reasons. The first one is to minimize the mean error and the second one is to free the filter of the fundamental flows of measurement conversion, which is the process of changing different forms of measurements into the Cartesian coordinates. Most tracking applications perform a measurement conversion and then use a Kalman filter. Authors presented that the proposed algorithm performs better than the Kalman filter with two different measurement-conversion techniques.

In [30], two distributed particle filters with Gaussian Mixer approximation are proposed to localize and track multiple moving targets in a wireless sensor network. These two algorithms differ in how the distributed computing is performed. In the first algorithm, partial results are updated at each sensor clique sequentially based on partial results forwarded from a neighboring clique and local observations. In the second algorithm, all individual cliques compute partial estimates based only on local observations in parallel, and forward their estimates to a fusion center to obtain the final output. In order to conserve bandwidth and power, the local sufficient statistics (belief) is approximated by a low dimensional Gaussian mixture model(GMM) before propagating among sensor cliques.

In [31], a distributed Prediction-based Mobility Adaptive Tracking (P-MAT) algorithm, which provides a very high accuracy of tracking while minimizing the amount of energy used, is proposed. P-MAT uses an adaptive Kalman filter to predict the future

state - location and velocity - of the target. Based on the location prediction, the active tracking region is determined. Sensor nodes in this region are activated for sensing in the next cycle. Based on the velocity prediction, the size of the active region is changed. Information about the target is shared by all active sensors and P-MAT estimates the position of the target at any given time by taking the centroid of the location information received from these sensors.

In [32], Kam and Hodgkiss studied the target tracking problem in a WSN employing proximity sensors, which can provide very little information about the target location, compared to sensors that detect the received power of a target or directional information but increases the power efficiency of the network. In the proposed target tracking algorithm, Kalman filter is used to improve these position measurements to get better estimates of the target position. Kalman filter prediction error circles are used to select the sensor nodes which are going to be active for tracking and collaboration.

In [33], sensor nodes are modeled as coupled oscillators and a distributed Kalman filter is implemented. This mutually coupled oscillators-based method provides a simple and convenient implementation of a collaborative information process in sensor networks, and at the same time, provides the robustness and reliability. This method first spreads the information as a result of local coupling between adjacent nodes. Then dynamic consensus filter is applied to develop mutually coupled oscillators-based distributed Kalman filter (MCO-DKF) algorithm.

Another approach suggested by [34] to implement the DKF is to use bipartite fusion graphs. Khan and Moura employed bipartite fusion graphs in order to fuse the shared observations and shared estimates across the local models. They distributed the information filter form of the Kalman filter.

Yang et al. proposed two fully distributed algorithms based on one-time measurements and the Kalman filter approach in [35] for target tracking problem in mobile sensor

networks. Each agent maintains a target estimate and moves so as to maximize the expected information from its sensor. By communicating and fusing information with nearest neighbors each sensor gets a global estimate of the target and a local velocity vector field which the mobile sensor can follow to maximize its sensory information. The authors also showed that these algorithms perform better than centralized methods.

In [36–41], Olfati-Saber et al. proposed a distributed Kalman filter (DKF) algorithm, improved this algorithm with slight modifications for different scenarios and used it in different type of target tracking scenarios. In [36], Olfati-Saber et al. introduced a distributed filter, named consensus filter (CF), that allows the sensor nodes to track the average of n sensor node measurements. The CF plays an important role in solving the data fusion problem that allows implementation of a scheme for DKF in sensor networks. In [37], DKF is divided into two parts, both are separate dynamic consensus problems in terms of weighted measurements and inverse-covariance matrices. These two data fusion problems are solved is a distributed manner using low-pass and band-pass CFs. In [38], two types of DKF algorithms for sensor networks are introduced. The first type extends the DKF, proposed in [37], by replacing low-pass and band-pass CFs with the high-pass CFs. In other words, high-pass CFs to be used for both sensor data and covariance fusion, which makes the DKF compatible with heterogeneous multi-sensor networks. The second type of DKF algorithms are continuous-time DKF algorithms which share state estimates instead of sensor data among the network. These two algorithms, also named Kalman consensus filter (KCF) algorithms, first apply a local Kalman filter and then share the state estimates using CF.

Olfati-Saber used previously proposed DKF algorithm [37] in mobile sensor networks with a flocking-based mobility model in [39]. In [40], authors used KCF algorithms to solve the data association problem in distributed MTT architecture. Joint probabilistic data association (JPDA) is reformulated and used to perform local measurement to track associations. In [41], the problem of distributed tracking of a target is addressed using sensor networks with nodes that have limited sensing range (LSR). In such networks, the

main problem is that only a small percentage of sensor nodes can observe the target. KCF is an effective distributed estimation algorithm for sensor networks with a peer-to-peer (P2P) architecture.For solving this problem, authors suggested a hierarchical P2P architecture. KCF is modified for this architecture and a message-passing version of the (KCF) for sensor networks with LSR is proposed.

Other than the target localization, there are some hot topics such measurement to target association, target initiation, power control and sensor localization in target tracking. In [42], Songhwai et al. proposed a scalable hierarchical multiple-target tracking algorithm that is focused on false alarm rates and low detection probability. The algorithm can initiate and terminate tracks and requires a small amount of memory. It also robust against transmission failures, communication delays and sensor localization error.

MacErlean and Narayanan developed a framework for detection and tracking of moving objects in a distributed WSN in [43]. Their framework employs collaboration among the sensor nodes, which reduces the false alarms and avoids sending all measurements back to the central processing unit. Events occur in the network are associated together on local basis with a decentralized hypothesis propagation algorithm. Confirmed objects are reported when a suitable confidence criteria is satisfied.

In [44], Tinati and Rezaii developed a distributed MTT algorithm for WSNs. The Monte Carlo (MC) implementation of JPDAF (MC-JPDAF) is applied to the tracking and data association problems of MTT in a cluttered area. With the help of simulations, authors showed that this algorithm could be used dense networks.

3. MULTI-TARGET TRACKING FRAMEWORK

In [2], a fully-distributed collaborative multi-target tracking framework is proposed. This framework has the ability to handle the target tracking problem considering all aspects, including target initiation, association and localization. The distributed MTT framework is presented in Figure 3.1 as in [2]. Modules shown at the upper part of the track list manager module belong to the network track manager part. Modules shown at the lower part of the track list manager module belong to the local track manager part.

On the local track manager part, the sensor receives the targets range and bearing data with the aid of its sensory circuit. Modules on the local track manager part as in [2] are :

- Information extractor converts the range and bearing data related to the targets within the sensors detection range into the information denomination form according to 3.3 and 3.4, where $\boldsymbol{\varphi}$ is the observation, $\mathbf{H}$ is the observation matrix and $\mathbf{R}$ is the covariance matrix of observation. These information denominations are passed to the local plot list manager.
- Local plot list manager holds one plot per track. It is assumed that each target can generate at most one plot in the sensor node. Elements of an entry in the local plot list manager are the information state and the information matrix denomination values.
- Local track associator takes the current track list from the track list manager and associates the denominations in the plot list manager with them. Associated information denomination and track pairs are passed to the information update filter to update the current track state and information denominations are passed to the collaboration logic manager to be sent to the neighboring sensors.
- Collaboration logic manager decides to share or not to share the denomination values with the neighboring sensors.

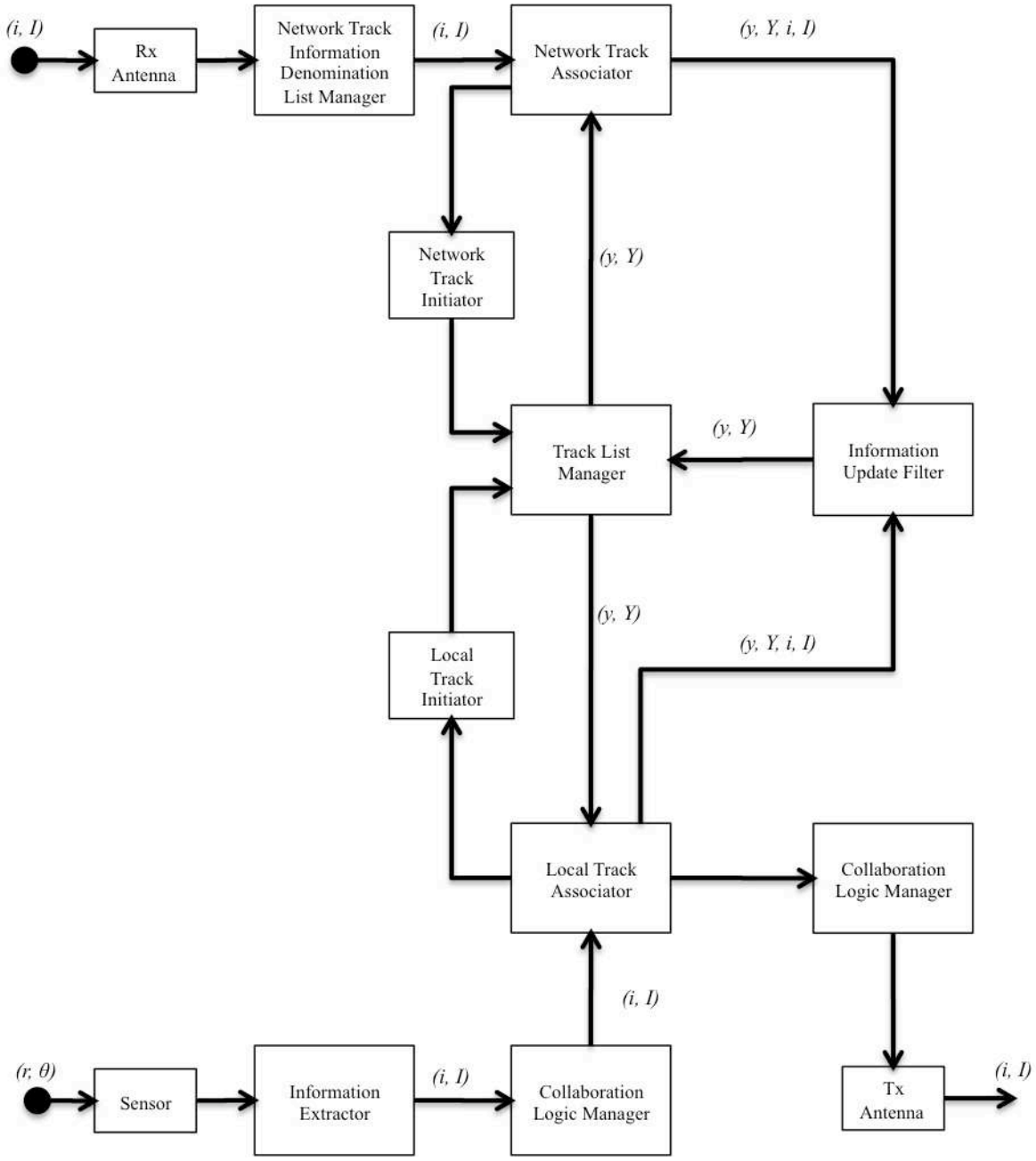

Figure 3.1. Distributed multi-target tracking framework [2]

- Information update filter updates the current track state using the information denominations of each track with the information filter and send the track states to the track list manager.
- Local track initiator receives the denominations that cannot be associated with the current tracks in the track list. Each information denomination is assigned to the track which is most likely to own these information denominations. Each track can be associated with only one information denomination and each information denomination can be associated only one track at a time. The local track initiator initiates a track and sends the track information to the track list manager in order to be added to the track list just if the track initiation criterion is satisfied. e.g., eight continuous detections from the same target have been received.

On the network track manager part; communication circuit of the sensor receives the information state and the information matrix denominations from the neighboring sensors which have been decided to be shared with the sensor network by the sensors that are detecting the target. Modules on the network track manager part as in [2] are :

- Network track information denomination list manager have the same assignment with the local plot list manager except that it may have many plots received from different sensors related with one track.
- Network track associator associates the information denominations in the network track information denomination list manager with the tracks in the track list manager. Shared information between sensors are the denominations related with the observations, not the target state estimations. The associator needs target state estimations in order to associate denominations with them. By putting the burden on computation, the local target state estimates are updated with the received denominations and afterwards the most likely association in a joint manner is performed. The constraint in the network track associator is that each plot can be assigned to at most one target. A sensor receiving an information matrix and the information state, calculates the target state information related with the received information

from the neighboring sensors. For each received information, likelihood values according to the local track states are calculated. As the mean and the covariance values of the local track states, we use the predicted states and the covariances of them. Associated denomination-track pairs are sent to the information update filter.

- Information update filter updates track states using the collaborative information filter and send them to the track list manager.
- Network track initiator receives denominations that cannot be associated with the tracks in the track list manager to initiate a track immediately and to send them to the track list manager. Tracks that have not been detected or reported from the neighbors for the last several (e.g, eight) cycles have been deleted from the track database in the track list manager.

In this framework, two modules, Information Filter Update and Collaboration Logic Manager, have an essential effect on the target localization.

3.1. Information Update Filter

Information update filter module is responsible for updating the track states in the framework. This module estimates the target states locally based on the local target states of the previous time interval and sensor observation about the target. A network estimation is performed based on the previous network estimation result and the information received from the network. This module uses the Information Filter (IF) for both local and network estimations.

In the IF formulation, the information state $\mathbf{y}$ and the information matrix $\mathbf{Y}$ associated with an observation estimate $\hat{\mathbf{x}}$, and the covariance of the observation estimate $\mathbf{P}$ at time instant k are given by [1], [2], [45], [46]

$$\hat{\mathbf{y}}(k) = \mathbf{P}^{-1}(k)\hat{\mathbf{x}}(k), \tag{3.1}$$

$$\mathbf{Y}(k) = \mathbf{P}^{-1}(k) \tag{3.2}$$

According to [1], [2], [45], [46], by means of sufficient statistics, an observation $\boldsymbol{\varphi}$ contributes $\mathbf{i}(k)$ to the information state $\mathbf{y}$ and $\mathbf{I}(k)$ to the information matrix $\mathbf{Y}$ where

$$\mathbf{i}(k) = \mathbf{H}^T\mathbf{R}^{-1}(k)\boldsymbol{\varphi}(k), \tag{3.3}$$

$$\mathbf{I}(k) = \mathbf{H}^T\mathbf{R}^{-1}(k)\mathbf{H} \tag{3.4}$$

where $\boldsymbol{\varphi}$ is the observation, $\mathbf{H}$ is the observation matrix and $\mathbf{R}$ is the covariance matrix of observation. $\mathbf{i}(k)$ is called the information state denomination and $\mathbf{I}(k)$ is called the information matrix denomination value.

Using the information state and information matrix denomination values, sensors update their own belief according to

$$\hat{\mathbf{y}}(k \mid k) = \hat{\mathbf{y}}(k \mid k-1) + \mathbf{i}(k), \tag{3.5}$$

$$\mathbf{Y}(k \mid k) = \mathbf{Y}(k \mid k-1) + \mathbf{I}(k) \tag{3.6}$$

Once the sensor nodes compute their local information denomination values, the collaboration logic manager decides which sensor nodes are going to share their information with the network. Compared to other filters, IF has a big advantage in terms of combining data received from other sensor nodes. IF uses the add operation to combine the information form of local observations received from the network. The distributed data fusion equations are

$$\hat{\mathbf{y}}(k \mid k) = \hat{\mathbf{y}}(k \mid k-1) + \sum_{i=1}^{N} \mathbf{i}_i(k), \tag{3.7}$$

$$\mathbf{Y}(k \mid k) = \mathbf{Y}(k \mid k-1) + \sum_{i=1}^{N} \mathbf{I}_i(k) \tag{3.8}$$

where N is the total number of sensors participating in the fusion process and $\hat{\mathbf{y}}(k \mid k-1)$ represents the information state estimate at time k given the observations up to and

including time $k-1$.

Just before the data at time k are collected, if we are given the observations up to the time $k-1$, the predicted information state and the information matrix at time k can be calculated from 3.9 and 3.10

$$\hat{\mathbf{y}}(k \mid k-1) = \mathbf{Y}(k \mid k-1)\mathbf{F}\mathbf{Y}^{-1}(k-1 \mid k-1)\hat{\mathbf{y}}(k-1 \mid k-1), \tag{3.9}$$

$$\mathbf{Y}(k \mid k-1) = [\mathbf{F}\mathbf{Y}^{-1}(k-1 \mid k-1)\mathbf{F}^T + \mathbf{Q}]^{-1} \tag{3.10}$$

where $\mathbf{Q}$ is the state transition covariance.

State estimate of the target at any time k can be found from

$$\hat{\mathbf{x}}(k \mid k) = \mathbf{Y}^{-1}(k \mid k)\hat{\mathbf{y}}(k \mid k) \tag{3.11}$$

In [46], advantages of using the information filter are listed as :

- Shares the information form of the observations results in a simple additive fusion framework that can be run on each of the tiny sensing devices instead of sharing the measurements related to the target state among the collaborating sensors.
- Saving the limited energy resources by reducing the computational load.
- Improves the numerical accuracy by preserving the symmetry and positive definiteness property of the state covariance matrix.
- Provides a means of the startup of the estimation without an initial estimate.

3.2. Collaboration Logic Manager

Collaboration logic manager is the module that decides if the sensor node is going to share its information or not with the network. After the local estimation is finished,

the sensor node will update its current local belief with the measurements of other sensor nodes. However, not all the sensor nodes provide useful information. The objective in this module is to select an optimal subset of sensor nodes and order the nodes in the subset.

It has to be known that the selection must be performed without any explicit knowledge of local estimation results of other sensor nodes. Knowledge about other sensor nodes' measurements requires communication of extra, unnecessary and less useful information. Thus, the decision should be given alone at each sensor node based on the sensor characteristics such as the sensor position, sensing modality and the predicted contribution of these sensors.

3.2.1. Maximum Mutual Information Based Sensor Selection Algorithm

The mutual information measures how much information one random variable tells about another one [47]. For target tracking applications, these random variables are the target state and the observation from the target. The mutual information between the target state and the observation measurement gives an idea about how much the observation tells about the target state. The algorithm, running in a sensor node for target tracking in a collaborative manner, is shown in Figure 3.2. Participation to the current cycle is decided based on the mutual information gained with the last observation. The mutual information gained with the last observation is formulated as

$$J(k, \boldsymbol{\varphi}(k)) = \frac{1}{2} \log \left[\frac{|\mathbf{Y}(k \mid k)|}{|\mathbf{Y}(k \mid k-1)|} \right] \tag{3.12}$$

where $\mathbf{Y}(k \mid k)$ is the information matrix, calculated by 3.6, at the time k and $\mathbf{Y}(k \mid k-1)$ is the predicted information matrix, denoted in 3.10, at the time k.

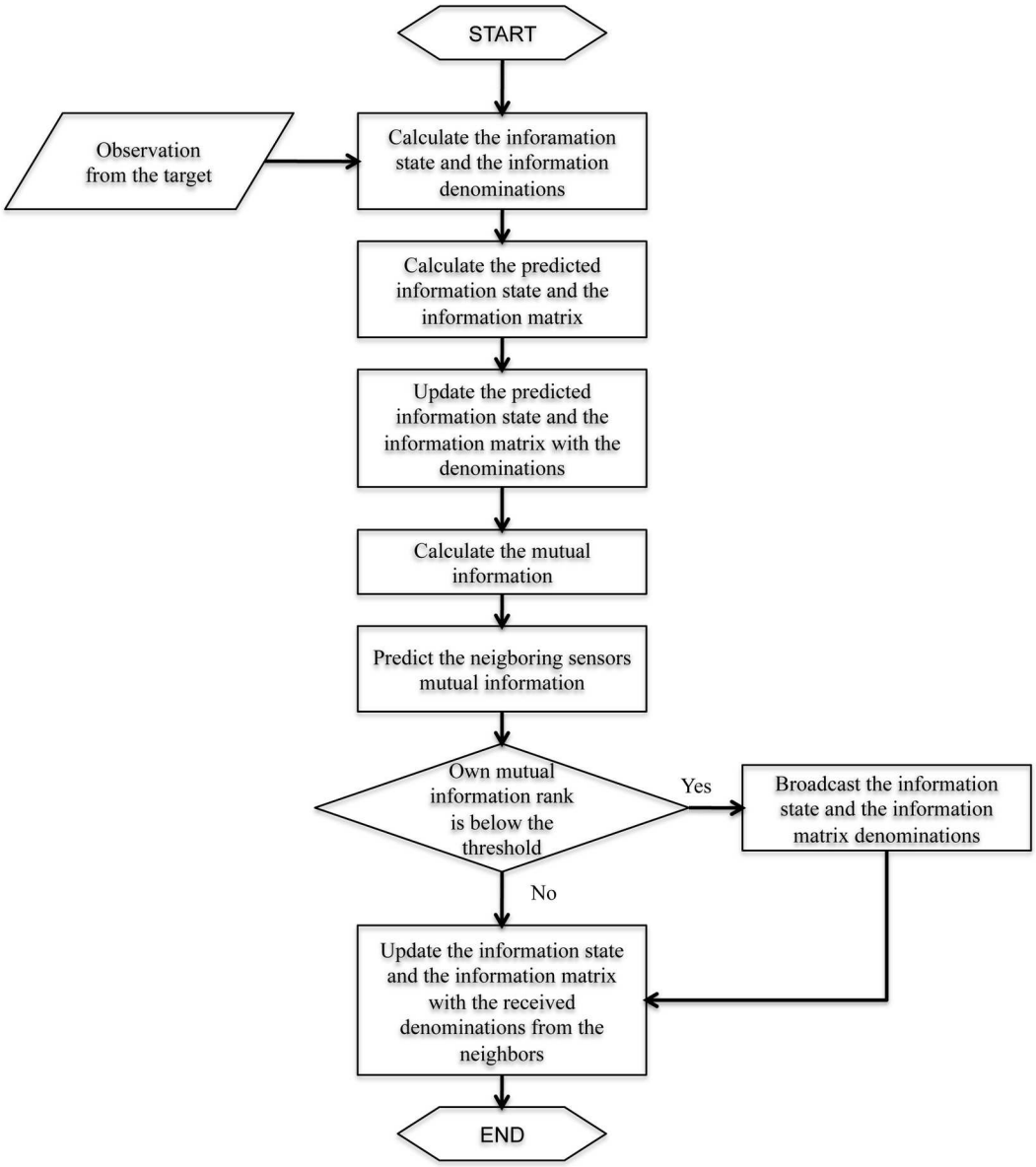

Figure 3.2. Target Tracking Algorithm with Maximum Mutual Information Based Sensor Selection Algorithm [47]

If the mutual information gain **J** is high enough to participate in the current cycle, the sensor node shares its own information about the target with other sensor nodes. Otherwise, it does not transmit its information in that cycle. A sensor node needs to know its neighbor sensor nodes' mutual information in order to know if its own mutual information is high enough to share. This can be estimated from the neighbor sensor nodes' position information and characteristics such as the standard deviation of target range

observations, the standard deviation of target bearing observations, the communication transmission power.

3.2.2. Minimum Mahalanobis Distance Based Sensor Selection Algorithm

The minimum Mahalanobis distance based sensor selection algorithm selects the closest sensors to the target location in terms of the Mahalanobis distance, which takes into account the correlations of the data. Mahalanobis distance of sensor node m at time k is calculated as

$$\mathbf{D}_m(k) = \sqrt{(\boldsymbol{\varphi}_m(k) - \mathbf{L}_m(k))^T P_m(k)^{-1} (\boldsymbol{\varphi}_m(k) - \mathbf{L}_m(k))} \tag{3.13}$$

where $\varphi_m(k)$ is the observation of sensor node m at time k and $\mathbf{P}_m(k)$ is the covariance matrix for sensor nodes observation at time k. $\mathbf{L}_m(k)$ is the sensor position. If the covariance matrix is the identity matrix then Mahalanobis distance is the same as Euclidian distance [47].

Every sensor node makes this calculation for itself and the other sensor nodes based on its prediction about their characteristics. Then the Mahalanobis distances are listed in the descending order. A sensor node determines to send its information if it is one of the first N_{max}, number of sensor nodes allowed to communicate in a time cycle, in the list.

3.2.3. Random Sensor Selection Algorithm

In this selection algorithm, sensor nodes randomly determines if they are going to transmit their measurements. Selecting the participating sensor nodes randomly means that a node detecting the target broadcasts its information immediately if the maximum number of sensor nodes to participate, N_{max}, has not yet been reached [46]. This can be decided by counting the number of target position announcements received from the

neighboring sensor nodes.

3.2.4. Comparison of Sensor Selection Algorithms

In [46], a comparison of these three sensor selection algorithms are given for dense and sparse networks. Sparse scenario has 300 sensor nodes and dense scenario has 800 sensor nodes randomly deployed in a 200 m $\times$ 200 m area. It is shown that for all three sensor selection algorithms, target tracking error is decreasing as the maximum number of sensor nodes allowed to communicate increases.

In both scenarios, mutual information measure results in terms of average tracking quality are better than the Mahalanobis distance based and random selection results. In dense networks, the tracking quality difference between the minimum mutual information based sensor selection algorithm and other two selection algorithms is larger than the corresponding difference in the sparse networks.

4. TARGET LOCALIZATION ALGORITHMS

In the distributed target tracking architecture proposed in [2] and [46], there are three major steps for target localization:

- Local Target Estimation - processes the raw data (usually observation measurements) sensed from target to have a more accurate estimation.
- Sensor Selection for Collaboration - locally determines if that sensor should or should not share its information.
- Network Target Estimation - combines the shared information for a better estimation of target location.

In the architecture, defined in [2], local and network estimators use the Information Filter which is implemented in the *information update filter* module. In this thesis, this module is divided into two parts as the *local track estimator* and the *network track estimator*, and new filters are implemented. Sensor selection decision is given in the *collaboration logic manager* module. In this thesis, this module is also updated with new sensor selection algorithms. Updated framework is shown in Figure 4.1.

Some of the new implementations need to use cartesian coordinates instead of information state and information matrix pairs. Cartesian coordinate and information conversions can be performed based on 3.3, 3.4 and 3.11 at any time in any module.

This chapter will first cover new local and network estimators, and sensor selection algorithms, which are the basis of new target localization algorithms. In Section 4.5, four target localization algorithms, which use new estimators and sensor selection algorithms, will be given.

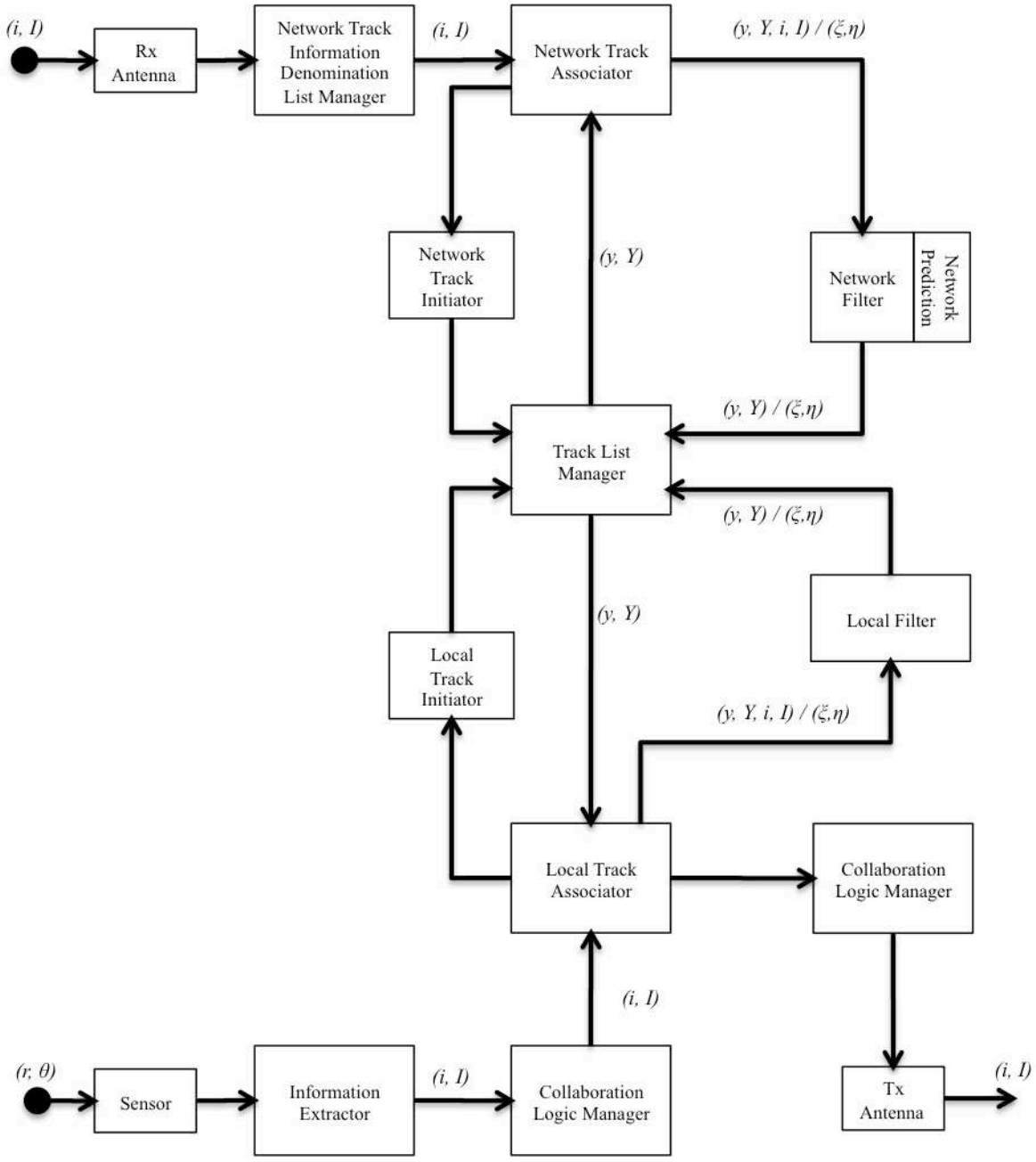

Figure 4.1. Modified distributed multi-target tracking framework

4.1. Local Estimator

4.1.1. Kalman Filter

Kalman filter (KF) is an optimal recursive data processing algorithm. The word recursive means that the KF does not require all of the previous data to be hold in the system. In other words, the KF requires only the estimated state from the previous time step and the current measurement to compute the estimate for the current state. A typical application model for the KF is presented in Figure 4.2 [48].

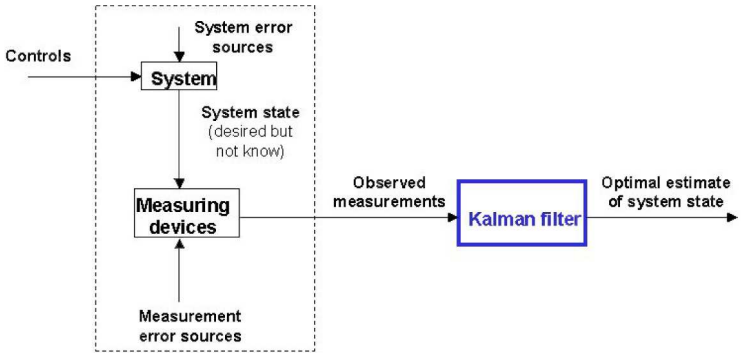

Figure 4.2. Typical Application of a Kalman Filter [48]

Kalman filter has two phases: Prediction (time update) and Update (measurement update). The predict phase uses the estimation from the previous time step and produces a prediction of the current time step. The update phase uses measurements of the current time step and refine prediction to be a more accurate state estimation for the current time step. Equations for running a KF in a sensor node are shown below.

In the prediction phase, state and error covariance prediction equations are

$$\hat{\mathbf{x}}(k \mid k-1) = \mathbf{F}\hat{\mathbf{x}}(k-1 \mid k-1) \tag{4.1}$$

$$\mathbf{P}(k \mid k-1) = \mathbf{F}\mathbf{P}(k-1 \mid k-1)\mathbf{F}^T + \mathbf{Q}(k-1) \tag{4.2}$$

where $\mathbf{F}$ is the state transition matrix and $\mathbf{Q}(k)$ is the state transition covariance.

The update phase has three steps. The first step is to calculate the error between the observation and the prediction using the following equations.

$$\mathbf{e}(k) = \boldsymbol{\varphi}(k) - \mathbf{H}\hat{\mathbf{x}}(k \mid k-1) \tag{4.3}$$

Then Kalman gain is found based on the predicted error covariance. Equations for the Kalman gain are

$$\mathbf{S}(k) = \mathbf{H}\mathbf{P}(k \mid k-1)\mathbf{H}^T + \mathbf{R}(k) \tag{4.4}$$

$$\mathbf{K}(k) = \mathbf{P}(k \mid k-1)\mathbf{H}^T\mathbf{S}(k)^{-1} \tag{4.5}$$

where $\mathbf{K}(k)$ is the Kalman gain for time k.

At the end, filtered estimation results for the state and the error covariance is calculated using 4.6 and 4.7.

$$\hat{\mathbf{x}}(k \mid k) = \hat{\mathbf{x}}(k \mid k-1) + \mathbf{K}(k)\mathbf{e}(k) \tag{4.6}$$

$$\mathbf{P}(k \mid k) = \mathbf{P}(k \mid k-1) - \mathbf{K}(k)\mathbf{S}(k)\mathbf{K}(k)^T \tag{4.7}$$

Kalman filter is used as a local estimator of target localization algorithms for linear targets. When non-linear targets are tracked, a non-linear version of KF, extended Kalman filter (EKF) is used. The next section explains EKF.

4.1.2. Extended Kalman Filter

The extended Kalman filter is the non-linear version of the Kalman filter. EKF linearizes the current mean and covariance. In EKF, the state transition and observation

models do not need to be linear. Instead they can be differentiable functions.

$$\mathbf{x}(k) = f(\mathbf{x}(k-1)) + \mathbf{w}(k) \tag{4.8}$$

$$\boldsymbol{\varphi}(k) = h(\mathbf{x}(k)) + \mathbf{v}(k) \tag{4.9}$$

where $\mathbf{x}(k)$ is the true state of the target at time k, $\mathbf{w}(k)$ and $\mathbf{v}(k)$ are process and observation noises which are both assumed to be zero mean Gaussian noise with covariance $\mathbf{Q}(k)$ and $\mathbf{R}(k)$ respectively.

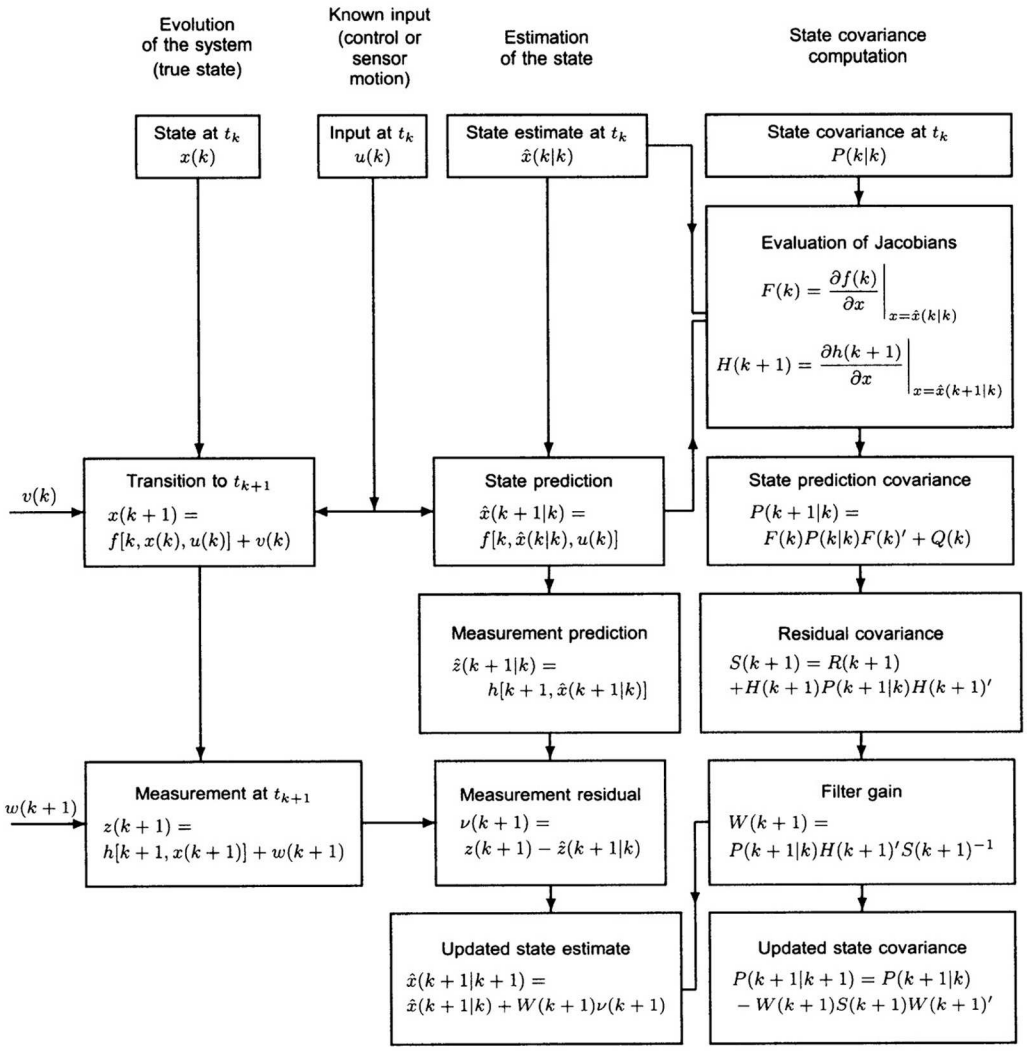

Figure 4.3. One-cycle Flowchart of Extended Kalman Filter [49]

As presented in the flowchart of EKF, Figure 4.3 [49], EKF also has same two phases, prediction and update, just like KF. The prediction phase equations are

$$\hat{\mathbf{x}}(k \mid k-1) = f(\hat{\mathbf{x}}(k-1 \mid k-1)) \tag{4.10}$$

$$\mathbf{P}(k \mid k-1) = \mathbf{F}(k)\mathbf{P}(k-1 \mid k-1)\mathbf{F}(k)^T + \mathbf{Q}(k-1) \tag{4.11}$$

Update phase equations are

$$\mathbf{e}(k) = \boldsymbol{\varphi}(k) - h(\hat{\mathbf{x}}(k \mid k-1)) \tag{4.12}$$

$$\mathbf{S}(k) = \mathbf{H}(k)\mathbf{P}(k \mid k-1)\mathbf{H}(k)^T + \mathbf{R}(k) \tag{4.13}$$

$$\mathbf{K}(k) = \mathbf{P}(k \mid k-1)\mathbf{H}(k)^T\mathbf{S}(k)^{-1} \tag{4.14}$$

$$\hat{\mathbf{x}}(k \mid k) = \hat{\mathbf{x}}(k \mid k-1) + \mathbf{K}(k)\mathbf{e}(k) \tag{4.15}$$

$$\mathbf{P}(k \mid k) = \mathbf{P}(k \mid k-1) - \mathbf{K}(k)\mathbf{S}(k)\mathbf{K}(k)^T \tag{4.16}$$

The main difference compared to KF, $\mathbf{F}$ and $\mathbf{H}$ are not static in the extended Kalman filter. $\mathbf{F}(k)$ and $\mathbf{H}(k)$ are calculated by the Jacobian of the state transition and the observation model at each time cycle with the following equations.

$$F(k) = \frac{\partial f}{\partial x}\Big|_{\hat{\mathbf{x}}(k-1|k-1)} \tag{4.17}$$

$$H(k) = \frac{\partial h}{\partial x}\Big|_{\hat{\mathbf{x}}(k|k-1)} \tag{4.18}$$

The linearization (evaluation of Jacobians) of the state transition and observation model can be done at the latest state estimate for $\mathbf{F}$ and the predicted state for $\mathbf{H}$.

EKF is used as a local estimator in the target localization algorithms for non-linear moving targets. It can also used for linear moving targets as EKF transforms into KF when the state transition and observation models are linear.

4.2. Network Prediction

Prediction is an essential part of estimation. As shown in Information, Kalman and extended Kalman filters, filters combine prediction and measurement to form a better, meaningful estimation. In some target tracking applications, local estimations are performed using filters but filters are not used in the data fusion process. This result in a basic combination of the information received from the network using simple algorithms such as the centroid localization.

In the updated framework, the network prediction is a sub-module of the network estimation module. It predicts the state of the target based on the previous network estimation with the same prediction method of the local filter in use.

Network prediction results are passed to the network estimators which decide to use or not to use these prediction results. If the network prediction is used, the network estimator module is also the decider for how to use the network prediction in estimation.

4.3. Network Estimator

The network estimator module receives the associated information from the network track associator module. This information could be either in the form of denominated information or cartesian coordinates based on the filters used in the system.

In the local estimator, filters combine the measurements from the target and their prediction about the target state. In the network estimator, prediction is performed in the network prediction sub-module. As there are no new measurements belonging to the target, information received from the network is used instead of the measurements.

4.3.1. Cooperative Adaptive Alpha-Beta Filter

Alpha-beta filter is a simplified form of the Kalman filter. Alpha-beta filter does not require a historical knowledge of the state and measurement noise covariances. Unlike the Kalman filter, the alpha-beta filter equations are derived with the assumption that measurements are made at regular intervals in time T. The main advantage of the alpha-beta filter is that the equations are simpler to implement than the Kalman filter. Alpha-beta filter equations are

$$\hat{\mathbf{x}}(k \mid k-1) = \hat{\mathbf{x}}(k-1 \mid k-1) + \Delta T \hat{\mathbf{v}}(k-1 \mid k-1) \tag{4.19}$$

$$\mathbf{e}(k) = \boldsymbol{\varphi}(k) - \hat{\mathbf{x}}(k \mid k-1)) \tag{4.20}$$

$$\hat{\mathbf{x}}(k \mid k) = \hat{\mathbf{x}}(k \mid k-1) + \alpha \mathbf{e}(k) \tag{4.21}$$

$$\hat{\mathbf{v}}(k \mid k) = \hat{\mathbf{v}}(k-1 \mid k-1) + \frac{\beta}{\Delta T}\mathbf{e}(k) \tag{4.22}$$

Cooperative adaptive alpha-beta filter is a modified version of the alpha-beta filter which is used for the network estimation. Prediction in 4.19 is already performed by the network prediction sub-module. The word collaborative means that the network filter uses the information received from the network instead of observation measurements (φ). Since the prediction phase is not performed by the filter, velocity equations are not needed. Collaborative adaptive alpha-beta filter equations are

$$\mathbf{e}(k) = \boldsymbol{\varpi}(k) - \hat{\mathbf{x}}(k \mid k-1) \tag{4.23}$$

$$\hat{\mathbf{x}}(k \mid k) = \hat{\mathbf{x}}(k \mid k-1) + \alpha \mathbf{e}(k) \tag{4.24}$$

where $\boldsymbol{\varpi}(k)$ is the information received from the network and $\hat{\mathbf{x}}(k \mid k-1)$ is the network prediction result.

The reason for choosing the cooperative adaptive alpha-beta filter instead of the cooperative Kalman filter is to keep the network prediction simple and communication

needs low. The Kalman filter uses the Kalman gain ($\mathbf{K}$) instead of the static α parameter. The Kalman gain calculation requires knowledge about the state and measurement covariances, $\mathbf{Q}$ and $\mathbf{R}$ respectively. This will bring extra communication cost to the network since $\mathbf{Q}$ and $\mathbf{R}$ will be shared with the local estimation.

4.4. Collaboration Logic Manager

A new Euclidian distance based sensor selection algorithm is added to the collaboration logic manager. In this algorithm, nearest sensor nodes to the target are selected to share their information.

4.4.1. Minimum Euclidian Distance Based Sensor Selection

Euclidian distance based sensor selection algorithm selects the closest sensor nodes to the target. Euclidian distance of sensor node m at time k is calculated as

$$E_m(k) = \sqrt{(\boldsymbol{\varphi}_m^{\xi}(k) - \mathbf{L}_m^{\xi}(k))(\boldsymbol{\varphi}_m^{\eta}(k) - \mathbf{L}_m^{\eta}(k))} \tag{4.25}$$

where $\boldsymbol{\varphi}_m^{\xi}(k)$ is x-axis and $\varphi_m^{\eta}(k)$ is y-axis of observation belonging to the sensor node m at time k. $\mathbf{L}_m^{\xi}(k)$ is the x-axis and $\mathbf{L}_m^{\eta}(k)$ is the y-axis position of the same sensor node m.

Every sensor node make this calculation for itself and the other sensor nodes based on its prediction about their characteristics. Then the Euclidian distances are listed in descending order. A sensor node determines to send its information if it is one of the first N_{max} in the list. N_{max} is the number of sensor nodes allowed to communicate in a time cycle.

4.5. Target Localization Algorithms

We propose two new target localization algorithms, which use a combination of the newly suggested and previously implemented filters and sensor selection algorithms. These algorithms are

1. Cooperative Center of Mass Localization
2. Information-Based Cooperative Center of Mass Localization

A general block diagram for the target localization algorithms is shown in Figure 4.4. In this block diagram, connections between modules are shown. Each algorithm has its own block diagram with the used algorithms, filters and information transfer forms. There is only one module, which is not explained in the framework. This module is called the information combiner. Its aim is to combine the information received from the network. This module can just simply add the received information if it is an information denomination form or use a centroid localization if cartesian coordinates are used by the algorithm. How to combine data is in the localization algorithms responsibility according to the filters and sensor selection algorithms used.

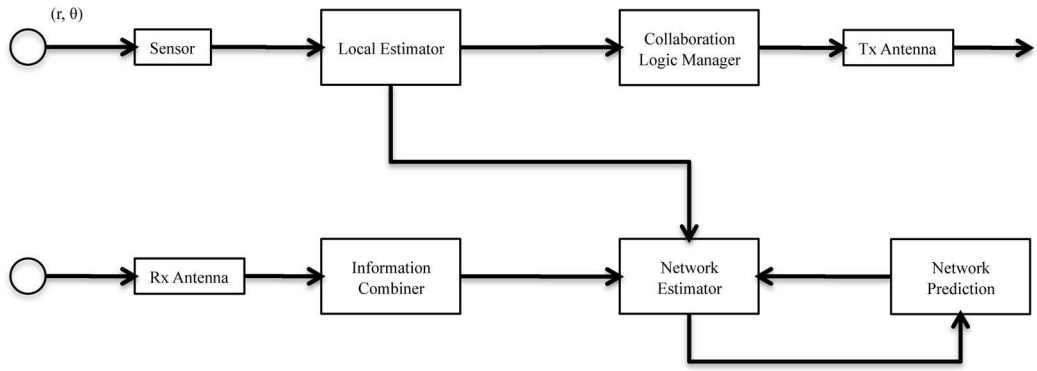

Figure 4.4. Block Diagram of a Sensor Node with Localization Algorithms

A generic form of the localization procedure is shown in Figure 4.5. Depending on the algorithm requirements, cartesian coordinates or information denominations are shared among the modules. On the local tracking side, first, target's measurements are

sensed as (r,θ) pair. This measurement is converted to either the cartesian coordinates or the information denominations and sent to the local estimator module. This module uses a local filter to calculate sensor node's local estimation results. They are passed to the collaboration logic manager which employs a sensor selection algorithm, which decides whether the sensor node shares or does not share its local results.

1. Calculate local estimation results using local filter.
2. Decide to send or not to send local estimation results using sensor selection algorithm
3. Combine network information using information combiner algorithm
4. Predict target's state based on previous network filtered state
5. Calculate network estimation results using network filter.

Figure 4.5. Target Localization Algorithm

On the network tracking side, local results of other sensor nodes are received. These results are combined in the information combiner module. After local estimation results received from the network, they are combined. The network filter is used to estimate the targets position using network prediction and combined results.

4.5.1. Cooperative Center of Mass Localization

This algorithm employs the extended Kalman filter as the local estimator, euclidian distance based sensor selection algorithm and cooperative adaptive alpha-beta filter as the network estimator. The block diagram of this algorithm, employed in a sensor node, is shown in Figure 4.6

Local results, received from the network, are combined in the information combiner module using a weighted centroid combination method. Every sensor node has a weight based on its order in the list of sensor selection algorithm. This weight is in inverse proportion with the euclidian distance of the sensor node to the target. Weighted centroid

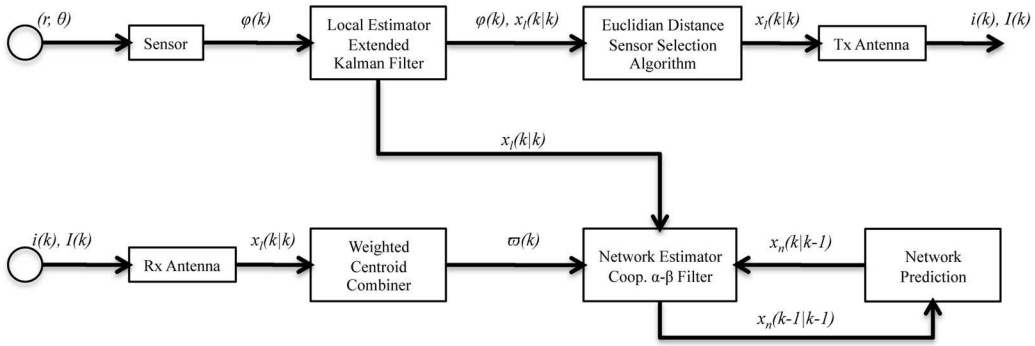

Figure 4.6. Block Diagram of a Sensor Node with Cooperative Center of Mass Localization Algorithm

combination equations are

$$\boldsymbol{\xi}_n(k) = \frac{\sum_i^N \boldsymbol{\xi}_i(k)\mathbf{W}_i(k)}{\sum_i^N \mathbf{W}_i(k)} \tag{4.26}$$

$$\boldsymbol{\eta}_n(k) = \frac{\sum_i^N \boldsymbol{\eta}_i(k)\mathbf{W}_i(k)}{\sum_i^N \mathbf{W}_i(k)} \tag{4.27}$$

where $\boldsymbol{\xi}_n(k)$ is the x-axis and $\boldsymbol{\eta}_n(k)$ is the y-axis of cartesian coordinates. Together they form $\boldsymbol{\varpi}(k)$, information received from network, used in 4.23. $\mathbf{W}_i(k)$ is the weight of sensor node i at the time step k and calculated as

$$\mathbf{W}_i(k) = \frac{1}{\mathbf{E}_i(k)} \tag{4.28}$$

Cartesian coordinates are shared among the modules in this localization algorithm. However information denominations are shared between the sensor nodes because other modules in the framework, presented in [2], work with information denominations.

4.5.2. Information-Based Cooperative Center of Mass Localization

This algorithm employs the information filter as local and network estimators, and the euclidian distance based sensor selection algorithm. The block diagram of this algo-

rithm, employed in a sensor node, is shown in Figure 4.7

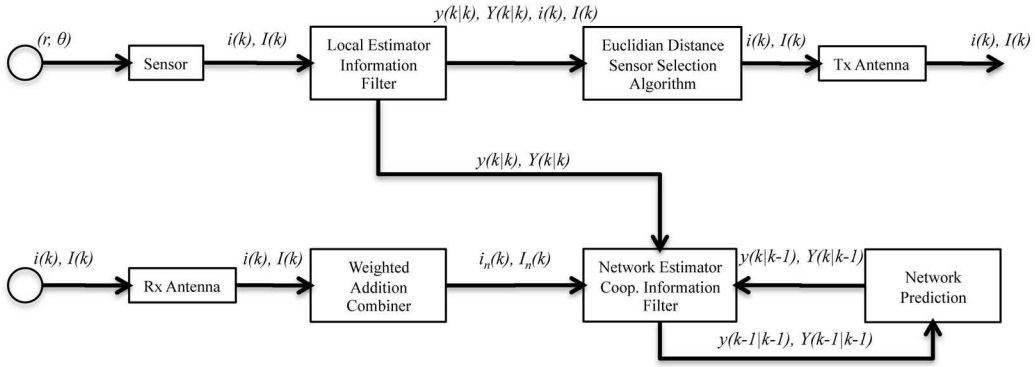

Figure 4.7. Block Diagram of a Sensor Node with Information-Based Cooperative Center of Mass Localization Algorithm

In the information combiner module, just a simple add operation is used. After the local estimation results received from the network, they are combined. The cooperative information filter is used to estimate the targets position using the network prediction and combined results. Since the information denominations are combined with add operations, the center of mass logic is provided by multiplying the information denominations with the weights of each sensor node. After this modification, data fusion equations become:

$$\hat{\mathbf{y}}(k \mid k) = \hat{\mathbf{y}}(k \mid k-1) + \sum_{i=1}^{N} \mathbf{i}_i(k)\mathbf{W}_i(k), \tag{4.29}$$

$$\mathbf{Y}(k \mid k) = \mathbf{Y}(k \mid k-1) + \sum_{i=1}^{N} \mathbf{I}_i(k)\mathbf{W}_i(k) \tag{4.30}$$

where $\mathbf{W}_i(k)$ is calculated by 4.28.

5. PERFORMANCE EVALUATION OF LOCALIZATION ALGORITHMS

This chapter aims to analyze the suggested target localization algorithms. Following the information about the simulation environment, we comment on different test scenarios and simulation results using the mean error and the energy consumption.

5.1. Simulation Environment and Parameters

We run Monte Carlo simulations to evaluate the performance of the proposed target localization algorithms, described in Section 4.5, in terms of power consumption and localization accuracy, based on the methodology and the work in [1, 2, 46].

Table 5.1. Parameters used in the simulations

Parameter	**Description**	**Values**
S	Number of sensor nodes	300*, 550, 800
Field	Sensing field dimensions	200m × 200m
TGT	Number of targets	1* to 5
MOV	Target Trajectory	Sin*, Lin
α	Alpha-Beta Filter parm.	0.5
T	Time	100s
N_{max}	# of sensors allowed to communicate	1 - 20
TGT_{max}	# of targets to be reported by a sensor	1
SR	Sensing Range	18m
CR	Communication Range	110m
DM	Detection Model	Binary Detection
CM	Communication Model	Shadow Fading
PC	Power Control	ICTP

Parameters used in our simulations are shown in Table 5.1. Values, used during simulations, are also given in this table. Stared values are used as default unless otherwise specified in simulations. We use the parameters of the TWR-ISM-002-I micro-power impulse radar (MIR) with pseudo-random signaling. Typical detection range of a TWR-ISM-002-I is 18 meters [50] and communication range is 110m. The simulations are run for a flat, rural setting where the radio signal propagation is characterized by the shadow-fading model with parameters given in Table 5.2 [1, 2, 46]. Information-controlled transmission power (ICTP) adjustment scheme [45], based on the information content of the sensor about the target state, is used as power control mechanism. All data points in the result graphs and tables represent the means of 20 runs, each one is 100 seconds long and has a distinct random seed.

Table 5.2. Shadow fading communication model parameters. [1, 2, 46]

Carrier frequency	1.8 GHz
Path loss exponent	2
TX & RX antenna height	0.1 m
Shadow fading standard deviation	4
Sensor transmission power	-30 dB

In our simulations, we used three different values for the sensor node count which provide us sparse, medium dense and dense network configurations within the sensing area of 200m $\times$ 200m. Although we used one target in most of the simulations, we also analyzed the mean error in multiple target environments in Section 5.4. Sinusoidal target trajectory is used as the default motion model. Section 5.5 investigates the effect of the target trajectory to the localization performance. Section 5.6 gives a comparison of design alternatives of the framework modules used in the target localization. N_{max} is the number of sensor nodes that are allowed to share their information with the other sensor nodes in a time cycle. TGT_{max} is the maximum number of targets to be reported by a sensor node. List of all simulation performed in the thesis are given in Table 5.3 with the sections they are explained in.

Table 5.3. List of Simulations

Section	**S**	**TGT**	**MOV**	N_{max}	**Runs**	**Algorithm**
5.2	300	1	Sin.	1 to 8	20	All
5.3	800	1	Sin.	1 to 20	20	All
5.4	550	1 to 5	Sin.	20	20	All
5.5	300	1	Lin.	1 to 8	20	All
5.6.1	300	1	Sin.	1 to 8	20	CCL
5.6.2	300	1	Sin.	1 to 8	20	ICCL
5.6.3	300	1	Sin.	1 to 8	20	CCL
5.6.4	300	1	Sin.	1 to 8	20	CCL
5.6.5	300	1	Sin.	3	20	CCL

As mentioned in Section 3.2.4, in the target tracking framework, presented in [2] and [46], the localization method using the information filter with the maximum mutual information based sensor selection algorithm gives the minimum localization error. In our simulations, we compared the new localization algorithms with this method.

5.1.1. Process Models

We have used two types of target trajectories in our simulations. These are

1. Linear target trajectory,
2. Sinusoidal target trajectory.

The target process is a four dimensional vector that consists of the two dimensional position of the target, (ξ, η), and the velocity of the target, $(\dot{\xi}, \dot{\eta})$, at each of these dimensions.The target process state vector is defined by

$$\mathbf{x} = [\xi \, \eta \, \dot{\xi} \, \dot{\eta}]^T, \tag{5.1}$$

Linear process model of a target evolves in time according to

$$\mathbf{x}(k+1) = \mathbf{F}\mathbf{x}(k) + \mathbf{w}(k) \tag{5.2}$$

where $\mathbf{x}(k)$ is the real target state vector at time k and $\mathbf{w}$ is the process transition noise. $\mathbf{F}$ is the process transition matrix and used in all simulations as

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.3}$$

Sinusoidal process model of a target evolves in time according to

$$\mathbf{x}(k) = \mathbf{f}(\mathbf{x}(k-1)) + \mathbf{w}(k) \tag{5.4}$$

where $\mathbf{f}$ is the process function and used in all simulations as

$$\mathbf{f}(k) = \begin{cases} \xi(k) = \xi(k-1) + \dot{\xi}(k-1) \\ \eta(k) = \eta(k-1) + \dot{\eta}(k-1) \\ \dot{\xi}(k) = \dot{\xi}(k-1) + 1 \\ \dot{\eta}(k) = 10 * (sin((k+1)/8) - sin(k/8)) \end{cases} \tag{5.5}$$

5.2. Analysis of Localization Error in Sparse Networks

This scenario consists of 300 sensor nodes in a 200m × 200m sensing area. Target moves in a sinusoidal movement model with independent, Gaussian distributed process noise. An illustration of the simulation scenario is presented in Figure 5.1, which shows that when the target moves away from the sensor node, observation errors are increasing

and the localization algorithms reduce these observation errors.

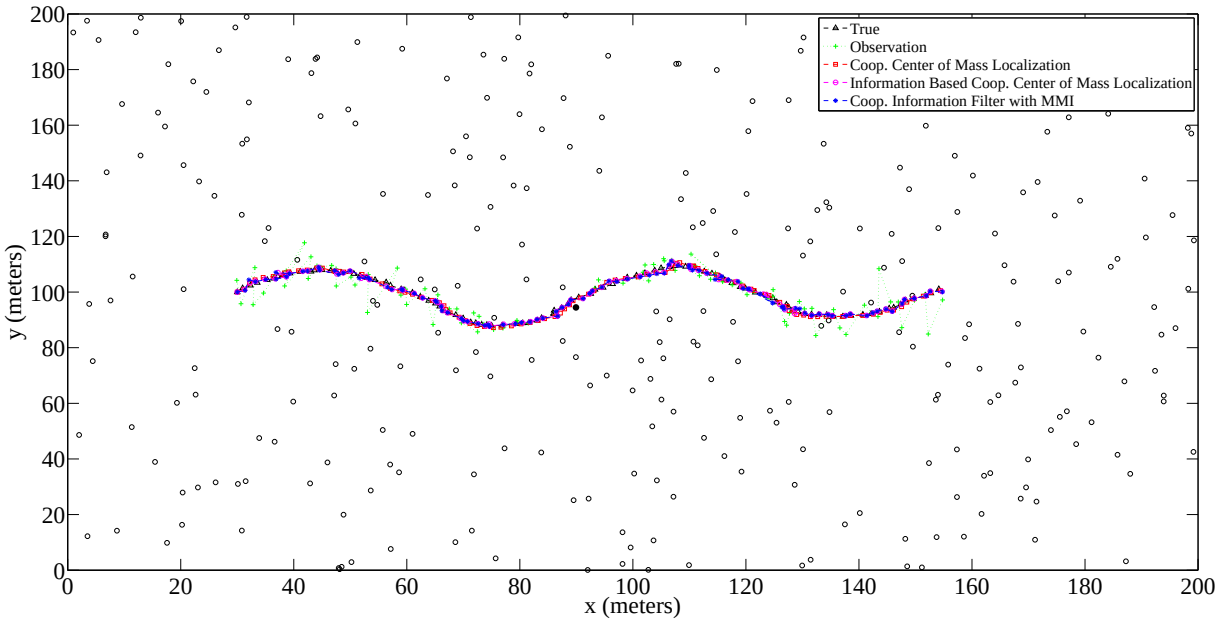

Figure 5.1. Illustration of the sparse network scenario

Figure 5.2 shows that as the number of sensor nodes participating to the collaboration increases, localization errors are decreasing. In the test cases with low number of participating sensor nodes, both cooperative center of mass localization (CCL) and information based cooperative center of mass localization (ICCL) algorithms perform better than the localization method using the cooperative information filter with the maximum mutual information based sensor selection (CIF-MMI). In the test cases with higher number of the participating sensor nodes, the mean error of CCL is constant where the ICCL and the CIF-MMI, which have similar decreasing localization errors.

Energy consumption increases as the number of sensor nodes allowed to communicate increases. This is an expected condition since the number of packets shared among the network inrease. All localization algorithms have same levels of energy consumption as shown in Figure 5.3.

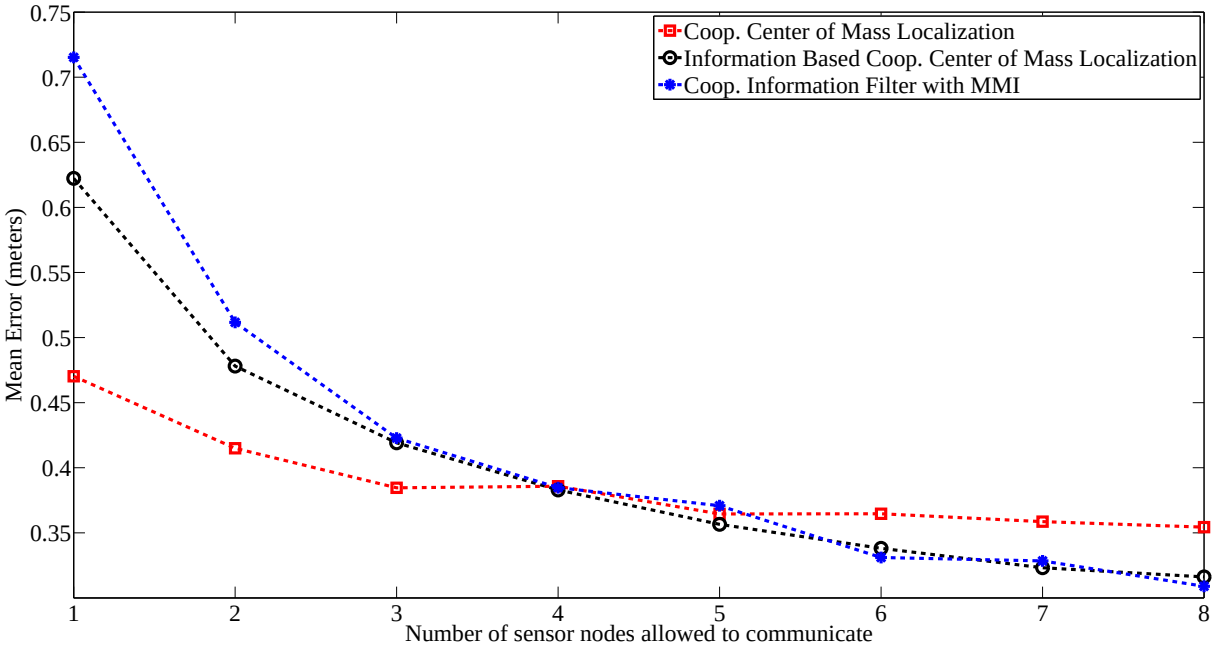

Figure 5.2. Mean error comparison for the sparse network

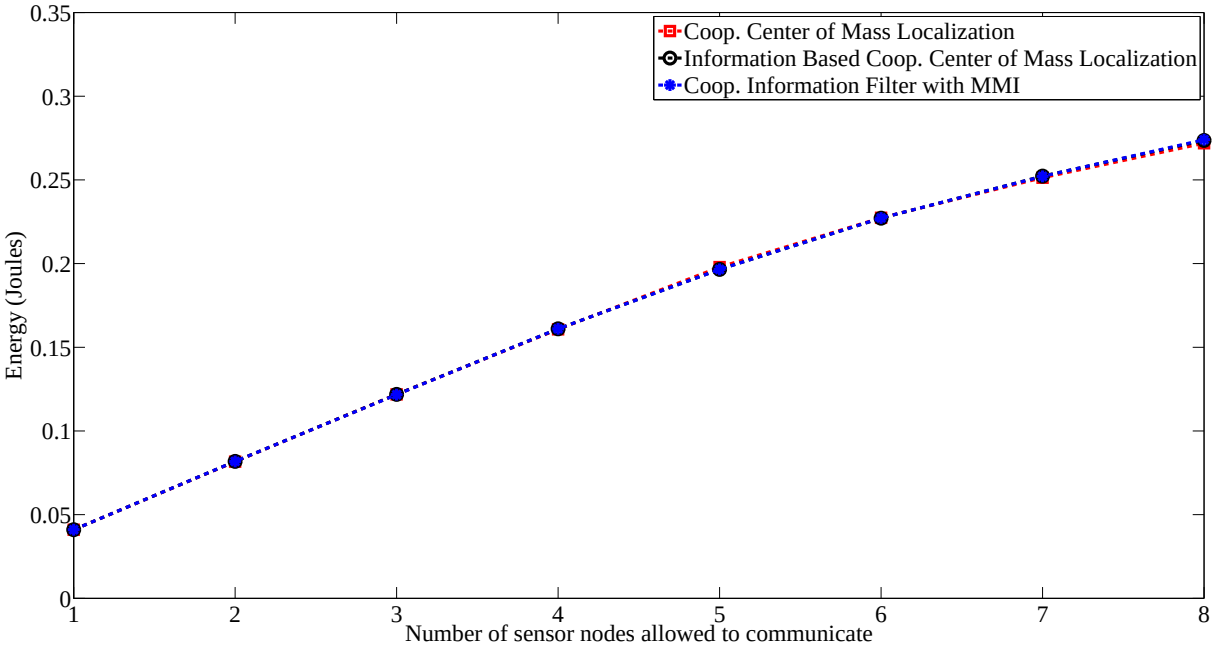

Figure 5.3. Comparison of the consumed energy for the sparse network

In Figure 5.4, the change of the mean localization error as the energy consumption increases is shown. For a given maximum energy level, we can examine the amount of localization error differences between the algorithms. If we select 0.15 joules as the maximum energy consumption level, we can have up to 34.2 per cent using CCL and 12.9 per cent using ICCL better localization accuracy than CIF-MMI. When there is no energy constraint, we can use still use ICCL because it has similar localization accuracy for higher energy levels with CIF-MMI.

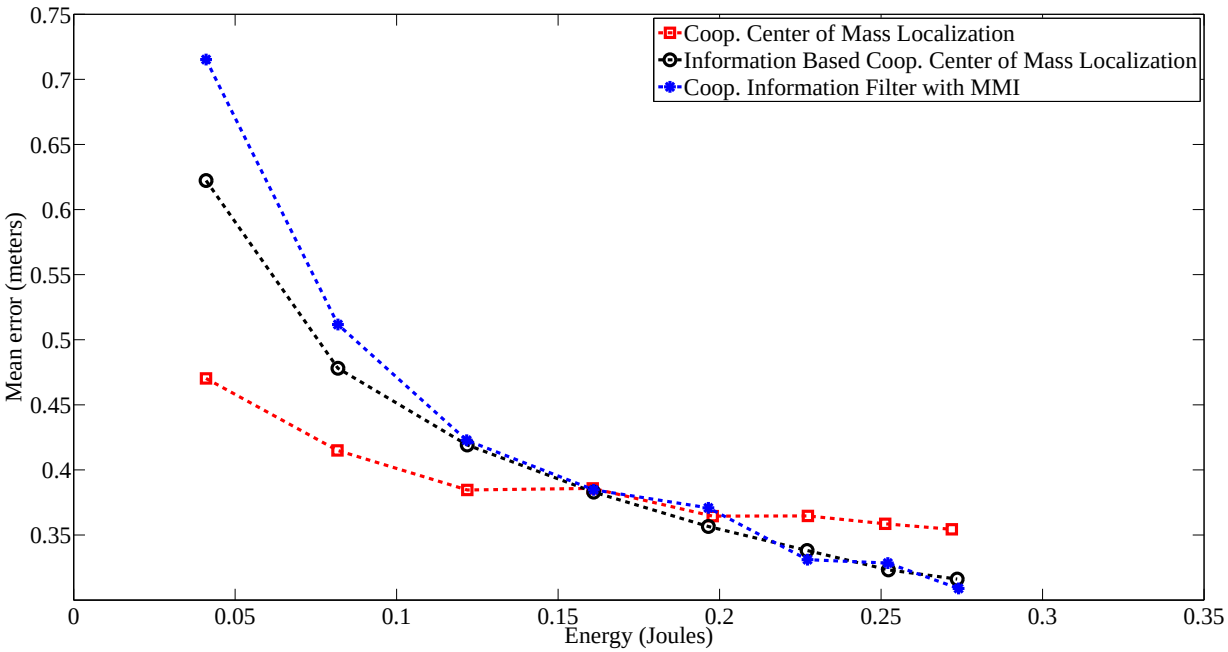

Figure 5.4. Comparison of the mean error for the desired energy consumption for the sparse network

CCL performs better on low energy levels since it selects the closest sensor nodes to the target and weights the information from these sensor nodes. It is expected that sensor nodes closer to the target have better information. However, it is not always true, especially for the sensor nodes contributing in the fourth or fifth place. As the number of sensor nodes participating to the collaboration increases, CCL selects sensor nodes closer to the target instead of sensor nodes which have better information but are farther from the target. In the lowest energy levels, CCL also performs better than ICCL although

they both use the same selection algorithm because the CCL uses a non-linear estimator in the local estimator module and the network prediction sub-module, it is able to track the sinusoidal target better for the low contribution levels.

5.3. Analysis of Localization Error in Dense Networks

This scenario presents a comparison of the localization algorithms in dense networks, consists of 800 sensor nodes in a 200m × 200m sensing area. Target moves in a sinusoidal movement model with independent, Gaussian distributed process noise. Illustration of the network is shown in Figure 5.5 from the solid dotted sensor node's view located at (89,92).

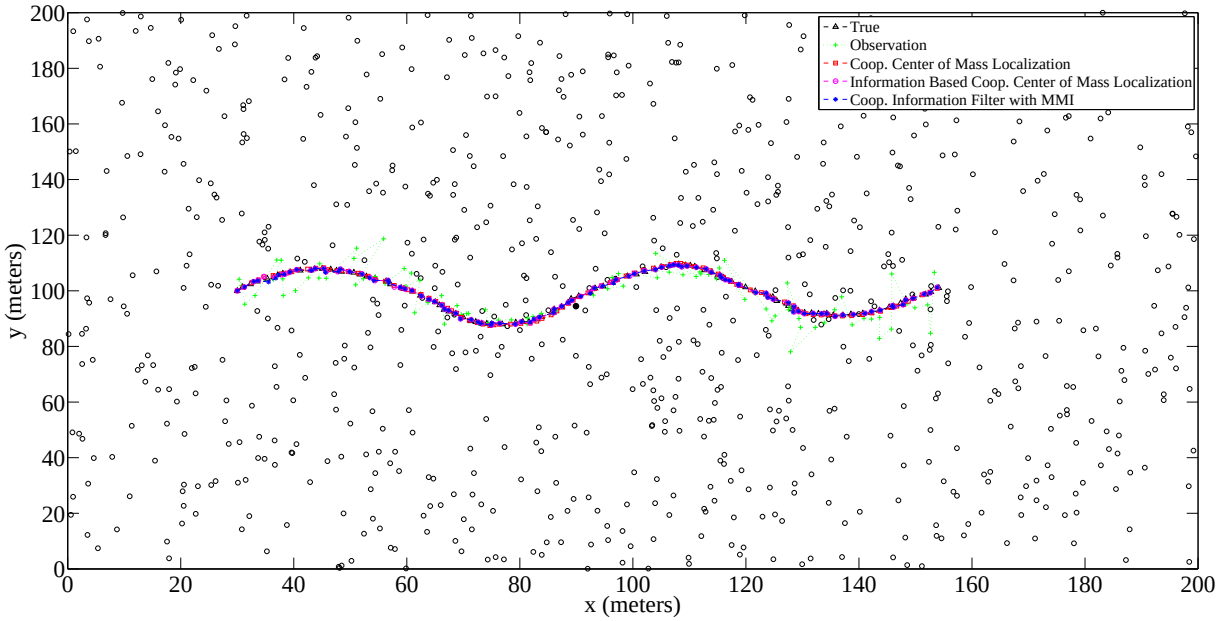

Figure 5.5. Illustration of the dense network scenario

The mean error has a similar trend for dense networks compared to sparse networks. In both networks, CCL and ICCL algorithms have better localization accuracy than CIF-MMI for low level of contribution as shown in Figure 5.6. The energy consumption is shown in Figure 5.7. All algorithms have the same level of energy consumption, that is

increasing as the contribution increases.

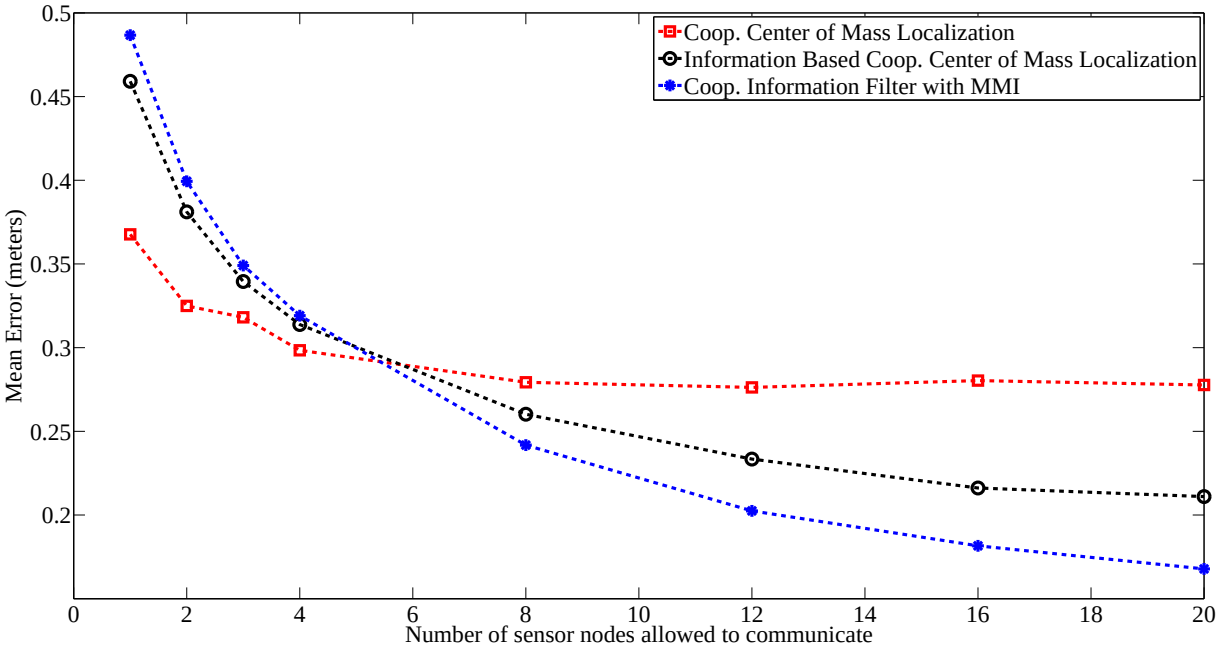

Figure 5.6. Mean error comparison for the dense network

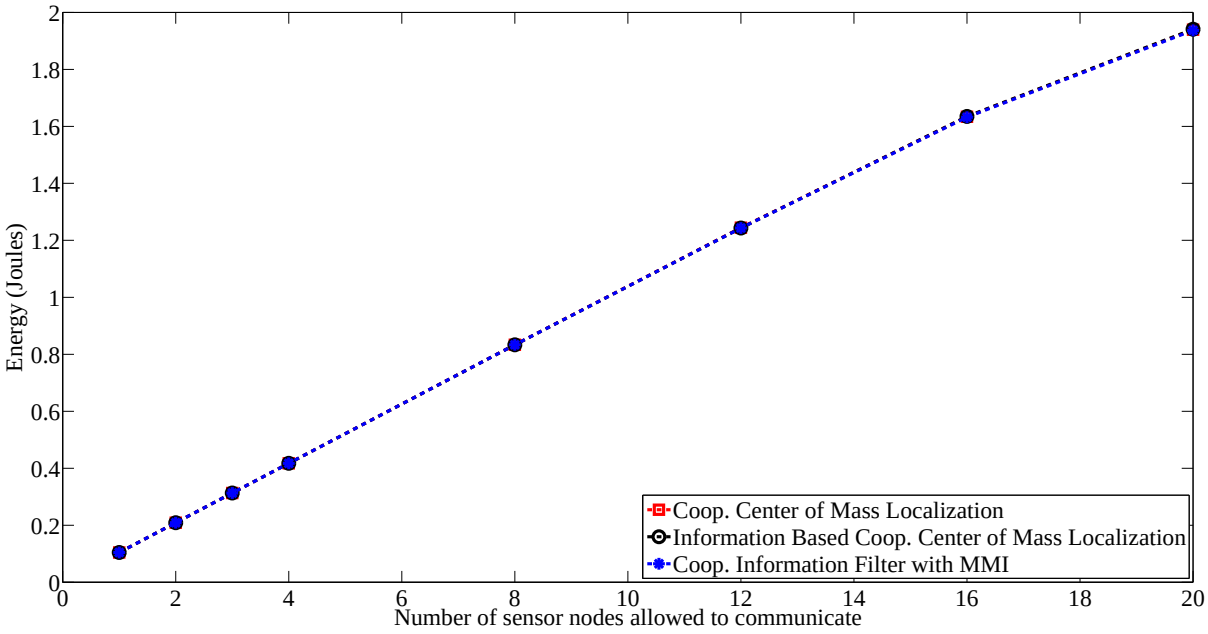

Figure 5.7. Comparison of the consumed energy for the dense network

In Figure 5.8, the change in the mean localization error as the energy consumption increases is shown. For a given maximum energy level, we can examine the amount of localization error differences between the algorithms. If we select 0.6 joules as the maximum energy consumption level, we can have up to 24.4 per cent (using CCL) and 5.6 per cent (using ICCL) better localization accuracy than CIF-MMI.

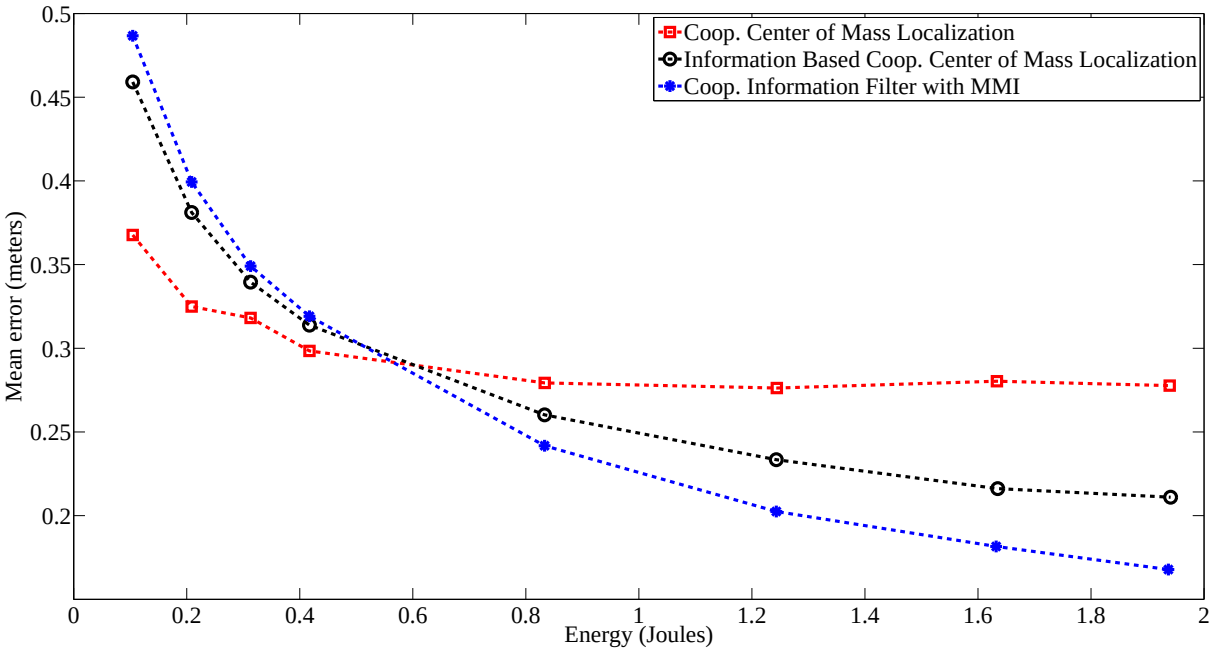

Figure 5.8. Comparison of the mean error for the desired energy consumption for the dense network

Although CCL and ICCL is still better than CIF-MMI in terms of mean error, there is a decrease in the error difference compared to sparse networks. Since there are more sensor nodes detecting the target in dense networks, CIF-MMI is able to find sensor nodes with better information in its first three participating sensor nodes. Also the information quality difference between the closest sensor node and other detecting sensor nodes will not be as much as sparse networks.

Figure 5.8 also clearly shows that CIF-MMI is able to select sensor nodes with better information in higher N_{max} values. On the other hand, high level of energy consumption

does not make a big difference on CCL algorithm's mean localization error. Opposite to the sparse networks, ICCL algorithm is not as good as CIF-MMI in dense networks when the energy consumption is high. This also proves that closer sensor node does not have to be the one with better information, as mentioned in the previous section.

5.4. Analysis of Localization Error with Multiple Targets

Tracking multiple targets requires more energy consumption than the single target tracking in order to keep the localization accuracy in the same level. However, a network may require to keep the energy consumption below a threshold value even there are increasing number of targets. In this section, we analyzed a network that allows at most 20 sensor nodes to communicate regardless of the target count. In Figure 5.9, the scenario belonging to multiple targets in a sensing area of 200m $\times$ 200m with 550 sensor nodes is shown.

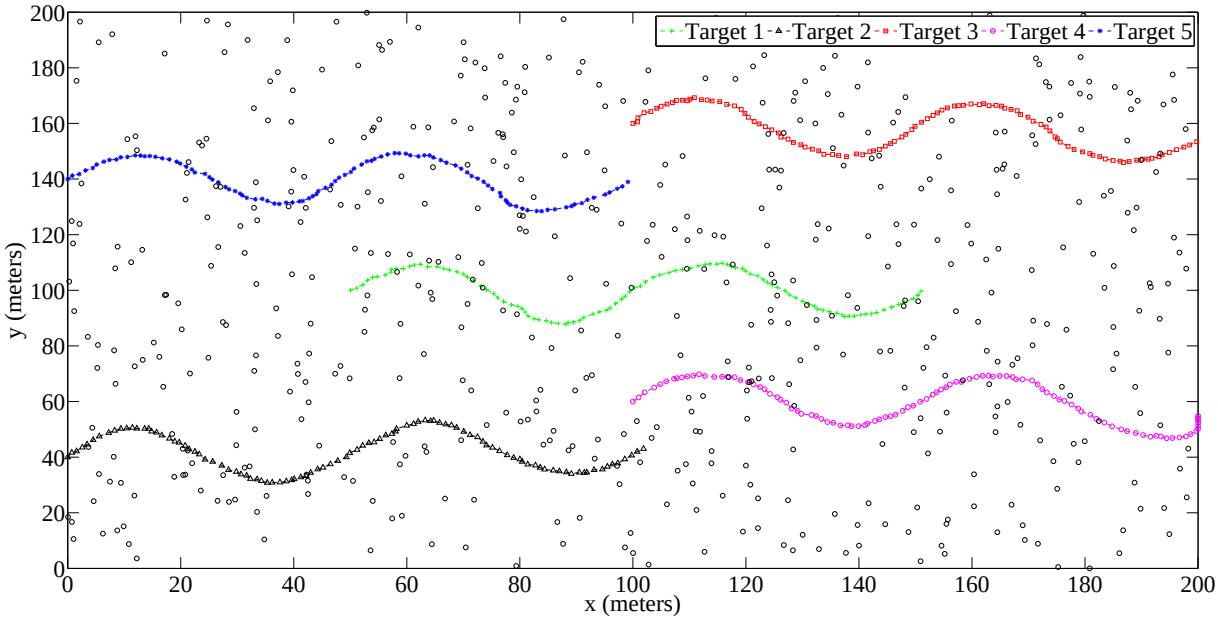

Figure 5.9. Illustration of five targets in a 200m $\times$ 200m area with 550 sensor nodes

Figure 5.10 shows that as the number of targets are increasing, the mean error of the network is increasing. This is an expected condition because tracking each new target brings a new localization error of that target to the network. There is also an additional error introduced with a new target as a result of the number of sensor nodes contributing to collaboration per target decreases. In the case that the system is tracking only one target, all 20 of the allowed sensor nodes contribute to the same target. When there are two targets, this number decreases to 10 sensor nodes per target and for the five targets scenarios, it is 4 per target. In other words, for a network, which allows only a limited number of sensor nodes to contribute in the network estimation, the level of contribution per target decreases as the number of targets are increasing.

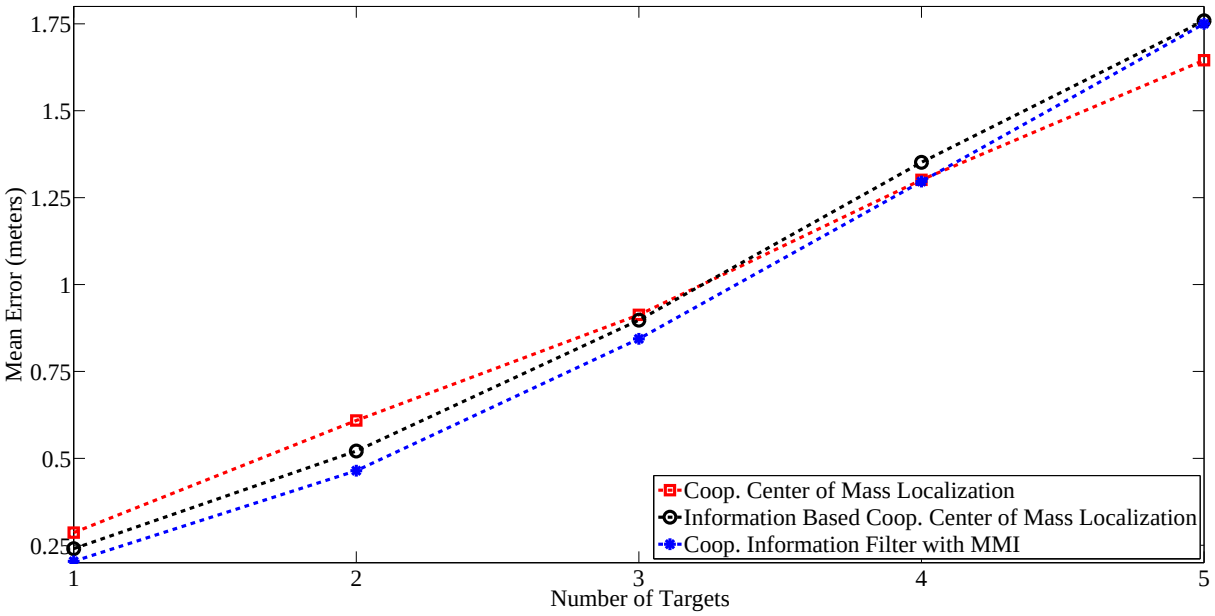

Figure 5.10. Mean error comparison for the increasing number of targets

Under the given conditions of this scenario, using the CIF-MMI gives better localization accuracy than the CCL up to three targets. When the number of targets are four, these two algorithms cause almost the same localization error. Tracking five targets with CCL is better than CIF-MMI in terms of localization accuracy. If there were lower number of contributing sensor nodes, we may see that CCL have lower mean error than

CIF-MMI with the addition of third or fourth target. The reason for CCL to perform better than other algorithms when the number of targets are increasing is the decrease in the contributing sensor nodes per target. As explained in the previous section, CCL algorithm performs better than other algorithms for low energy consumption requirements and low contribution levels.

From the previous sections, we also know that the ICCL algorithm also performs better than CIF-MMI for low level of contribution. In this scenario, we can see that the gap of the mean error between these two algorithms are closed as the number of targets are increased and the mean error for ICCL is the same with CIF-MMI for tracking five targets. We may be able to see that ICCL has better localization accuracy if the number of targets are increased to more than five.

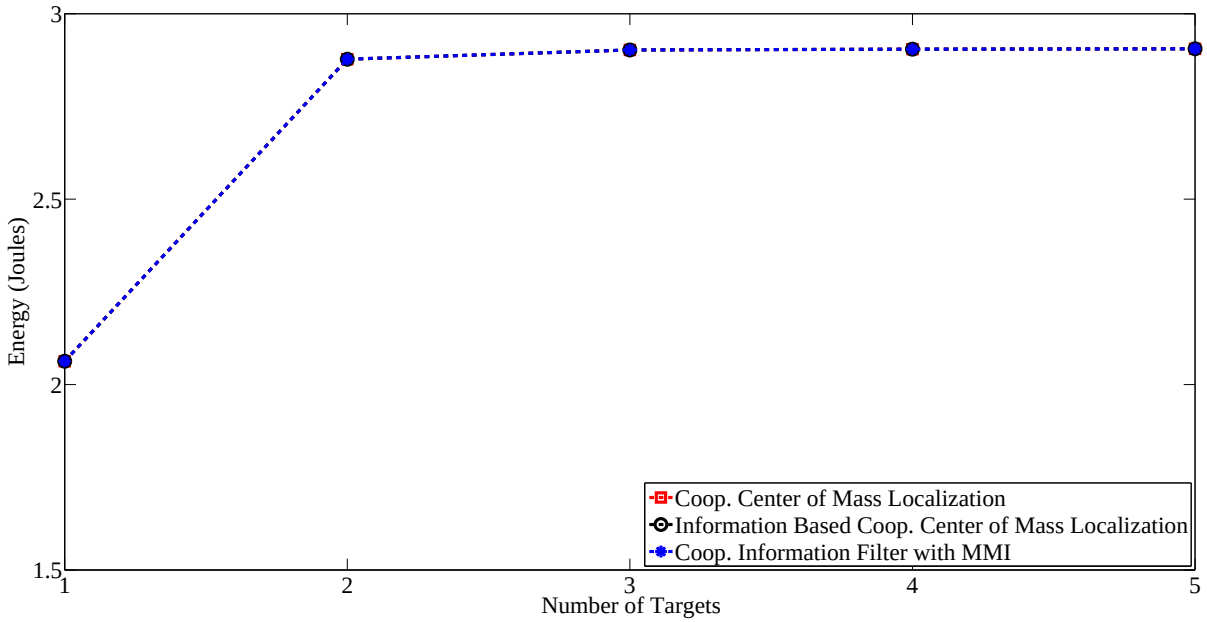

Figure 5.11. Comparison of the consumed energy for the increasing number of targets

Figure 5.11 shows that the consumed energy increases as the number of targets changed from one to two. The reason for this change is that the maximum number of contributing sensor nodes are not reached with one target. When the network is tracking

only one target, there can be time steps that the number of sensor nodes detecting the target is lower than 20, N_{max}. After two targets, the energy consumption is the same. When we examine the Figures 5.10 and 5.11 together, we can see that tracking five targets with CCL has 5.98 per cent better localization accuracy with the same energy consumption.

5.5. Analysis of Localization Error with a Different Target Trajectory

In the previous sections, we have simulated scenarios using sinusoidal moving targets. This gives a start-up advantage to CCL algorithm against other two localization methods since it uses a non-linear estimator. In this section, we will investigate the sparse scenario using a different target trajectory which has a linear movement.

Figure 5.12 illustrates a sample scenario for tracking a target with a linear trajectory in a sparse network. Same as Section 5.2, sparse network consists of 300 sensor nodes randomly deployed into a sensing area of 200m × 200m.

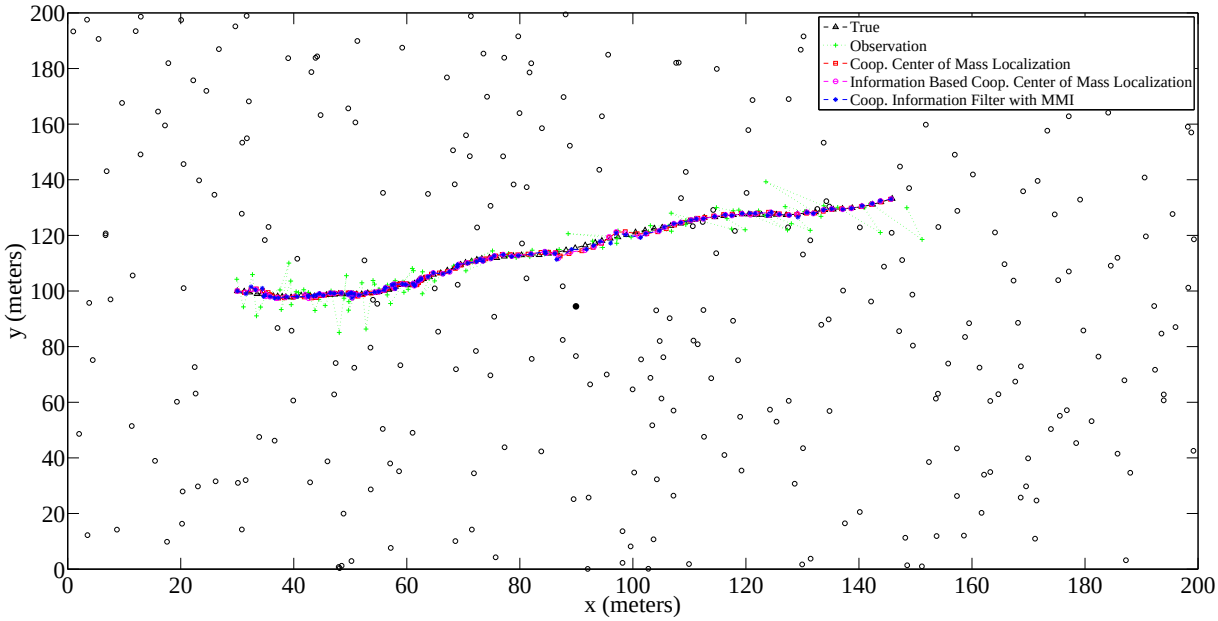

Figure 5.12. Illustration of the sparse network scenario with linear target

In Figure 5.13, the mean localization error is shown for the sparse network tracking a linear moving target. It is seen that CIF-MMI has the highest mean error like the sinusoidal moving target for the low sensor contributions. However, while tracking the linear target, information based localization algorithms converge faster than the case for tracking the sinusoidal target. ICCL and CIF-MMI provide significantly better results than CCL with at least six sensor contributing for the sinusoidal target. For the linear trajectory, these algorithms provide a lower localization error using just four sensor nodes in collaboration. Figure 5.14 shows that there is no difference in energy consumption depending on the target trajectory.

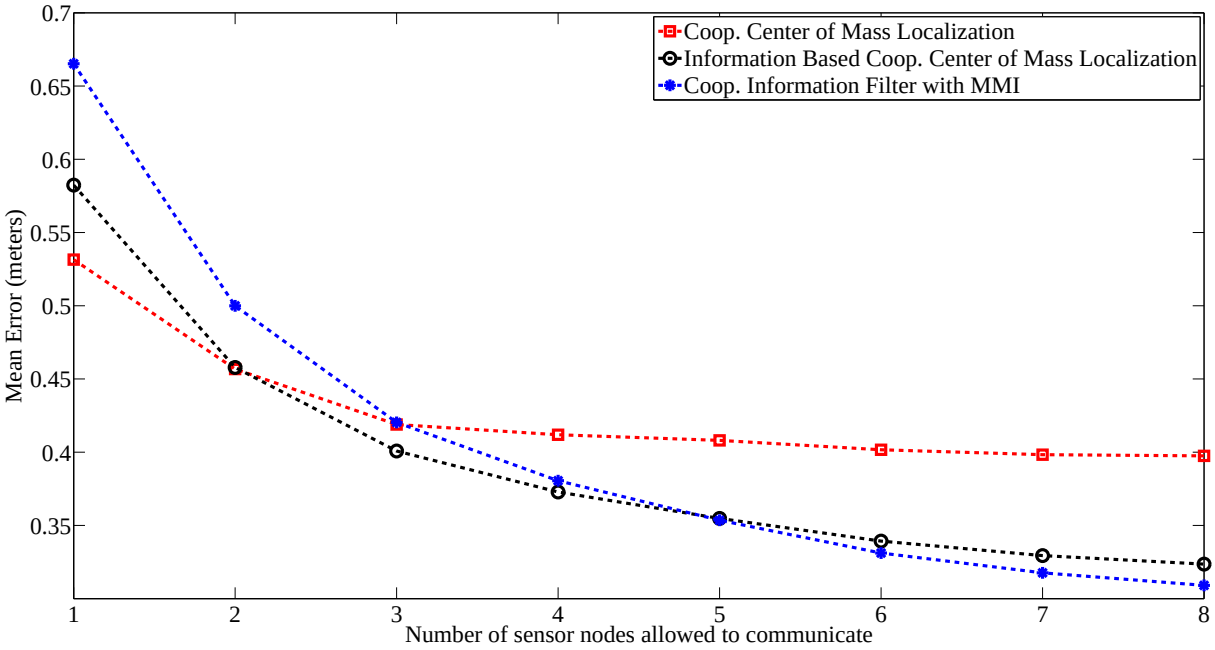

Figure 5.13. Mean error comparison for the linear target

The change of mean error with the energy consumption is shown in Figure 5.15. This figure shows us that CCL has a lower mean error than CIF-MMI for the network which have 0.12 joules energy consumption limit. This threshold is 0.16 joules for tracking the sinusoidal target. The reason for the decrease of the threshold value is CIF-MMI, which employs the information filter - a linear estimator -, performs better in tracking linear moving objects.

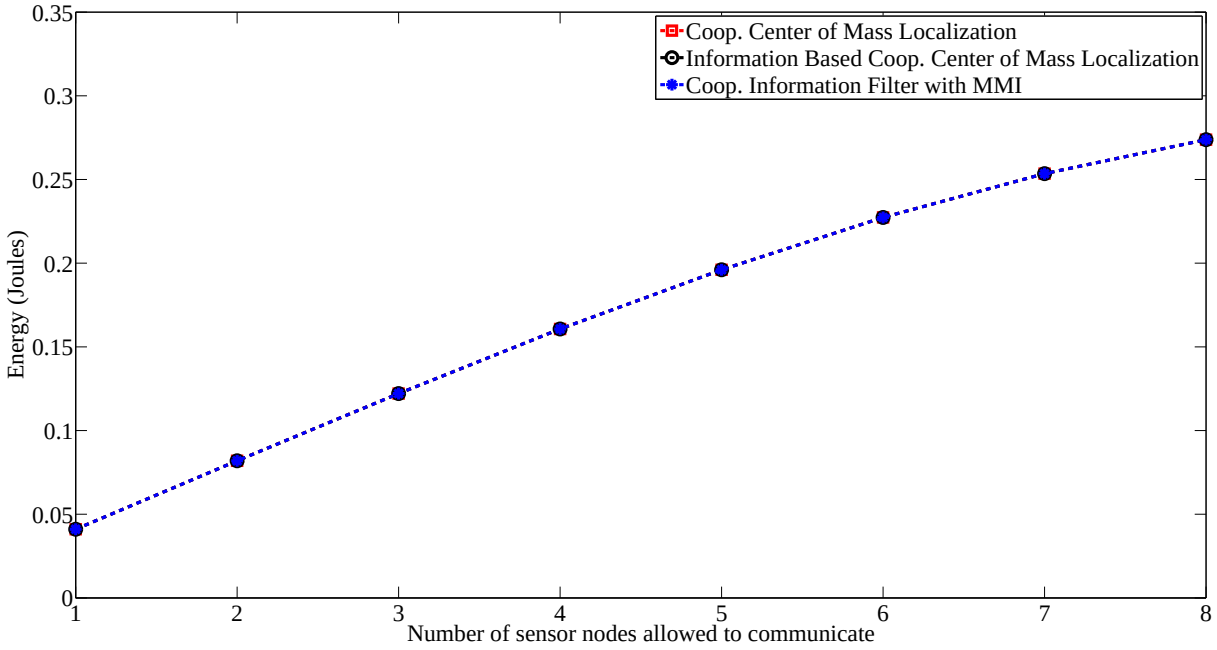

Figure 5.14. Comparison of the consumed energy for the linear target

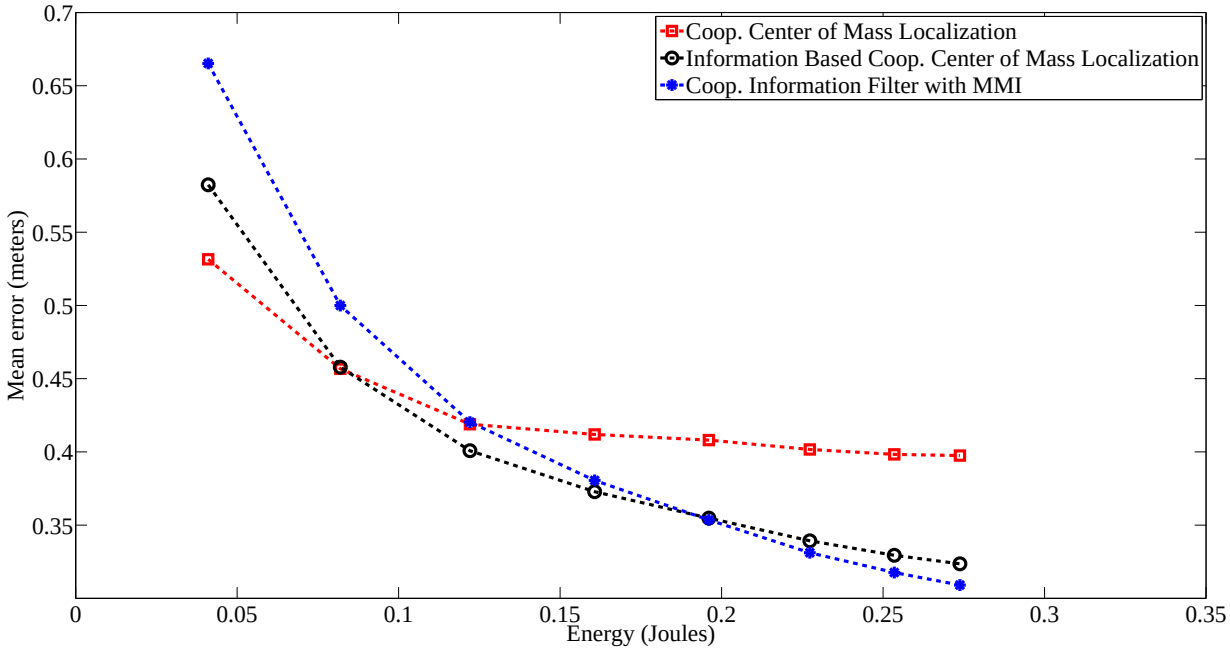

Figure 5.15. Comparison of the mean error for the desired energy consumption for the linear target

In this scenario, ICCL perform better than CIF-MMI in terms of the localization accuracy until the 0.2 joules of energy consumption. For networks exhaust more than 0.2 joules, ICCL almost performs similar to the CIF-MMI. In the lowest energy level shown in Figure 5.15, CCL and ICCL perform 20.1 per cent and 12.4 per cent better than CIF-MMI respectively. On the other hand, for the highest energy level in the same figure, CCL has 28.6 per cent bigger error than CIF-MMI where ICCL has only 4.6 per cent worse localization. Under the light of these differences, we can say that CCL is the best algorithm to use for low energy consumption levels and ICCL is an ideal localization method to use for the scenarios that we can have both low and high energy requirements.

5.6. Effect of Localization Modules

This section presents the effects of the chosen algorithms for the information combiner, the network prediction and the local estimator modules of the target tracking framework, presented in Chapter 3 and modified in Chapter 4.

First, we have investigated the information combiner module. In this module, we have used the weighted centroid information combiner for the CCL and the weighted addition combiner for the ICCL. The idea behind these implementations is to give higher priority to the sensor nodes' information closer to the target. We will compare them with the ones which are non-weighted in order to see how this idea effect the localization error.

Second, we have analyzed the network prediction module. As explained in Section 4.2, this module is decoupled from the network estimator in order to give a flexibility to the algorithm. We investigate how the presence of network prediction effects the mean error.

After the network prediction, we show the difference between the Kalman filter and the extended Kalman filter. The EKF requires more computation complexity than the KF. We have presented the mean localization error difference between these two local

estimators for the CCL.

At the end, we analyzed the effect of alpha (α) parameter of the cooperative alpha-beta filter. We have selected the alpha parameter for other simulations based on the results of this test case.

All simulations in this section is performed using 300 sensor nodes in a sensing area of 200m $\times$ 200m. Target is moving in a sinusoidal trajectory with noise as explained in the previous simulations.

5.6.1. Effect of Weighted Centroid Combination to Localization Error

Figure 5.16 shows the difference between the weighted centroid combination and the centroid combination for the CCL algorithm.

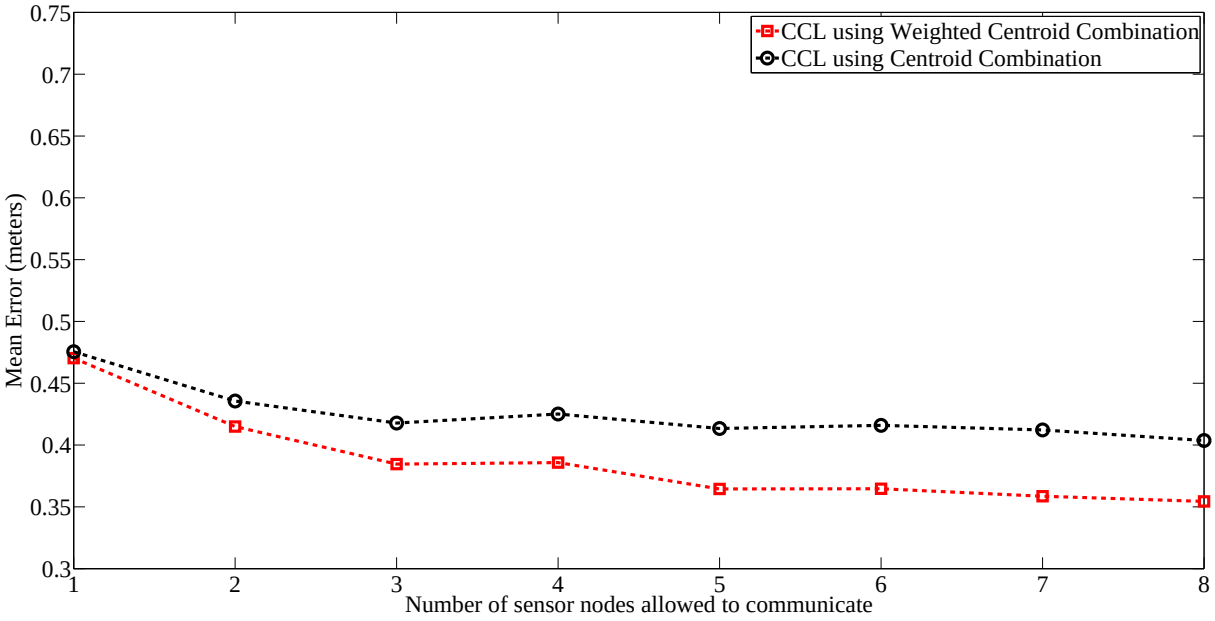

Figure 5.16. Effect of weighted combination to CCL algorithm

If only one sensor node contributes to the collaboration, there is almost no difference in mean errors for CCL algorithm's weighted and non-weighted centroid combination methods. The reason for that is the closest sensor node's weight does not make a big impact on the mean error by itself. As the contributing sensor node count increases, the gap between these two combination methods is increasing because non-weighted combination thread all contributing sensor node's information equally. By definition, the Euclidian distance based sensor selection algorithm assumes that the farther sensor node's information is worse than a closer one. So, the weighted combination method takes the worse information less but good information more into collaboration, by the help of weights. This behavior results in having better localization accuracy compared to that of the non-weighted centroid combination method.

5.6.2. Effect of Weighted Addition Combination to Localization Error

In Figure 5.17, it is shown that weighted combination makes the difference between ICCL and CIF-MMI localization methods.

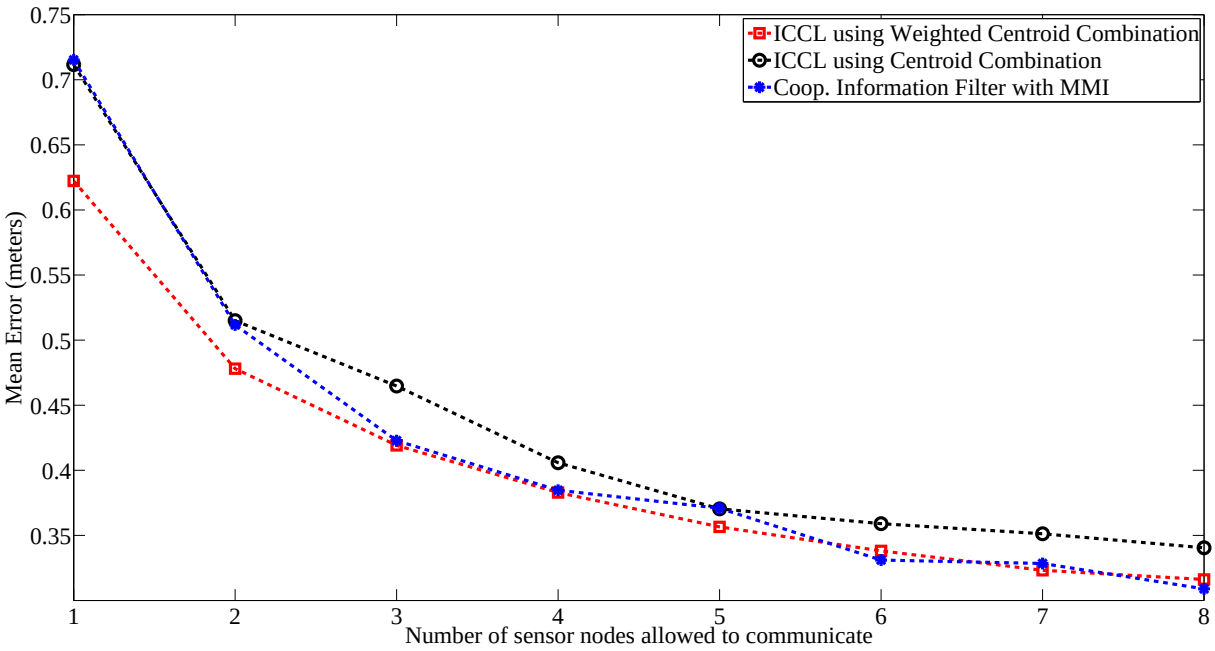

Figure 5.17. Effect of weighted combination to ICCL algorithm

The difference between weighted combination and non-weighted combination for ICCL algorithm is almost constant. For low sensor contributions, ICCL with non-weighted centroid combination method perform the same as the CIF-MMI.

When we have compared CCL and ICCL, we will see that weighted combination have a bigger effect for ICCL, especially if the number of sensor nodes contribute to the collaboration is low. The reason is that the CCL is using a non-linear estimator. The non-linear estimator gives better results in prediction and local filter results and so decrease the effect of weighted combination.

5.6.3. Effect of Network Prediction to Localization Error

In this section, we analyzed how the mean errors would be if we do not use the network prediction sub-module. Figure 5.18 shows the difference between the mean errors of the CCL algorithm with the network prediction and without the network prediction as the number of sensor nodes allowed to communicate increases.

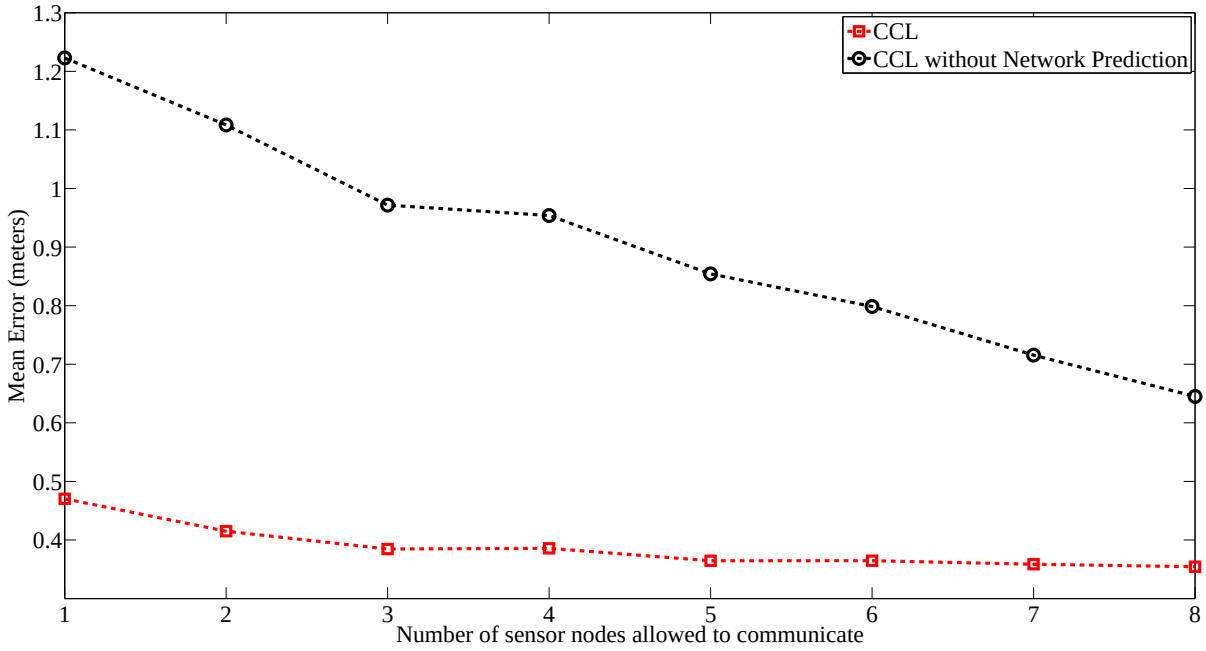

Figure 5.18. Effect of network prediction to CCL algorithm

When the contribution is in the minimum level, the prediction at the network filter decreases the mean error by 47.31 per cent. It is clear that network prediction has a huge decreasing effect on the mean error with the same number of sensor nodes communicating. The network prediction is an essential part of the filtering process which helps us to estimate the target's position more accurately. As the prediction is a key part of filtering, we will not be able to discard it in the localization process.

5.6.4. Comparison of the Kalman and the Extended Kalman Filters

Figure 5.19 shows the mean error belonging to the CCL algorithms that employ the KF and the EKF as local estimators. The CCL with the local estimator KF is not able to compete with other algorithms for a non-linear target on high number of contributing sensor nodes. On the other hand, if the system has low level communication requirements, CCL using the KF can be used as an option. Compared to CIF-MMI, it has 10.43 per cent and 5.96 per cent better localization accuracy for one and two sensor nodes allowed to communicate respectively.

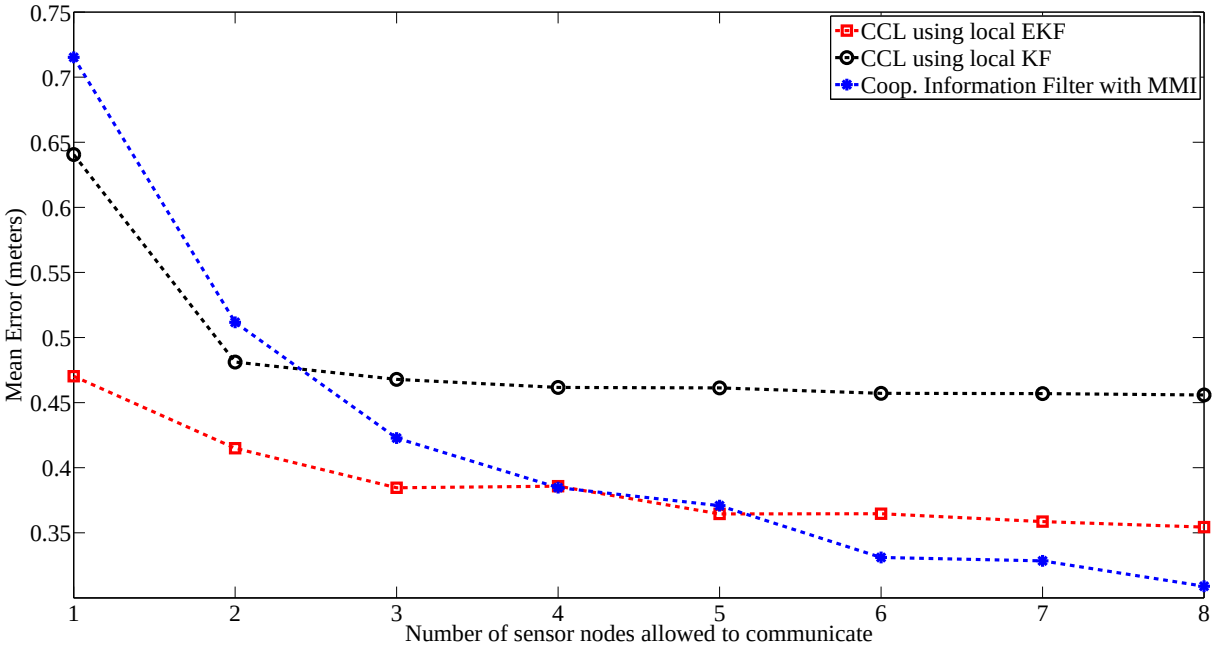

Figure 5.19. Comparison of local KF and local EKF effect on mean error

5.6.5. Effect of Alpha (α) to Localization Error

Figure 5.20 shows the mean error according to the change on alpha parameter of the cooperative alpha-beta filter. As shown in Equation 4.24, this parameter is used to decide how to combine the network prediction results and the state information received from the network. If alpha is zero, it means that the prediction is not taken into account. If alpha is one, then only the network predicted location is used as the result of cooperative alpha-beta filter. In our tests, three sensor nodes allowed to share their information with the network.

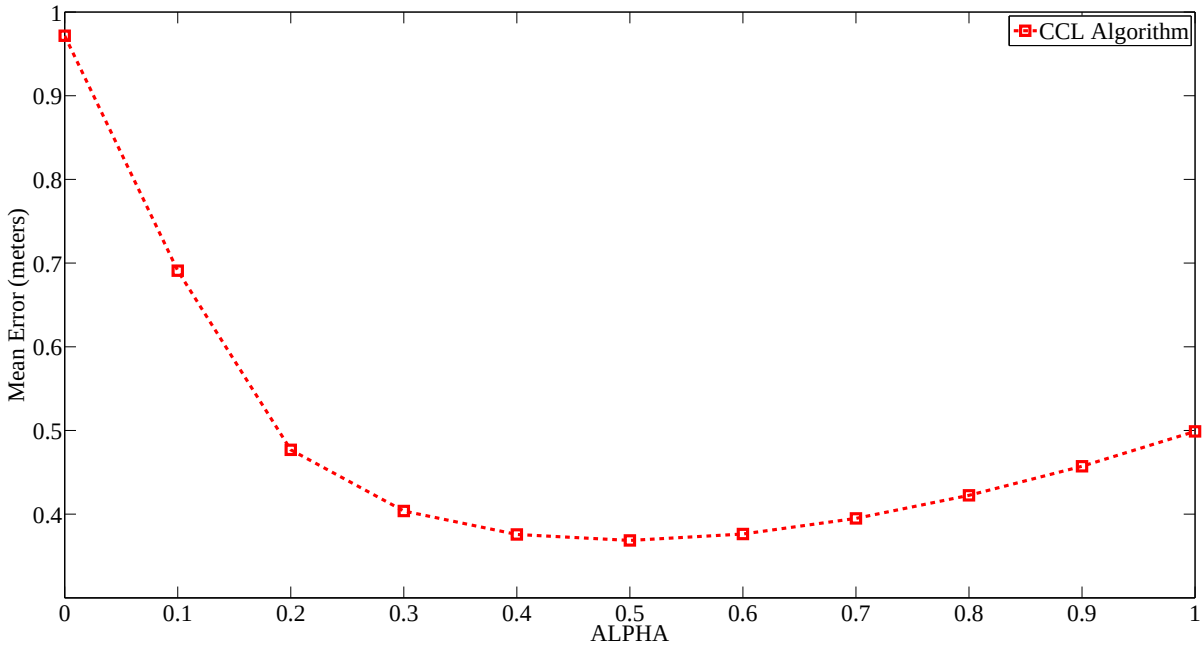

Figure 5.20. Effect of Alpha(α) to mean error

The minimum localization error is for the case that alpha is 0.5, which means that the network predicted state and the state information received from network is combined equally. As the alpha parameter increases or decreases, the localization accuracy is decreasing. Figure 5.20 also shows that the mean error only with the network prediction is lower than the mean error only with information from the network. This is an expected condition for this scenario because the information quality from the network is not as good

as the prediction as a result of the sparse network and only three sensor nodes allowed to share their data. Another reason is the non-linear local estimator, whose prediction step is also used for network prediction results in better predictions.

On the other hand, in Section 5.2, we analyzed that for the sparse networks when the maximum number of sensor nodes allowed to contribute to the collaboration is increasing, the mean error for the CCL algorithm is almost constant. This could be because of the alpha parameter, used as 0.5. Although the quality of the information received from the network is increasing, we still do not change the ratio of the network information in the cooperative filter.

6. CONCLUSIONS

In this thesis, we investigated the problem of the target localization problem of target tracking applications in WSNs. We proposed two new localization algorithms and made a performance comparison for different scenarios.

Our work is based on the multiple target tracking framework, proposed in [2]. We have updated this framework and implemented new methods for the modules in the framework. We have designated formal steps of a localization algorithm and then implemented two new target algorithms using different methods in the framework modules.

We have tested the proposed localization algorithms in eight different scenarios. We compared new algorithms with the existing localization methods in the framework. First, we tested algorithms for tracking single targets in sparse and dense networks. These two scenarios showed that our proposed algorithms are working better in terms of localization accuracy for the low level of energy consumption. Then, we evaluated the performance of algorithms in the scenarios with multiple targets. We also tested our algorithms with a linear moving target and showed that our algorithms can work better with this target trajectory for low contributions to collaboration. At the end, we have tested and compared the modules and parameters of our proposed algorithms including the weighted combination, the local estimator, the network prediction and the alpha parameter.

To conclude, we recommend our new algorithms if there is a low level energy consumption constraint. We also recommend that these algorithms can be used for tracking multiple targets under low energy consumption requirements.

6.1. Future Work: Adaptive and Dynamic Localization Algorithms

A target can move with different trajectories in a sensing area. It can be a sinusoidal or linear model, which are not changing from the beginning to the end of the sensing area in the current framework. However, this is not the condition for real life applications. A target is moving in an unpredictable dynamic model. The lack of all localization algorithms is to work statically without any change based on the target dynamics.

A dynamic localization decider, which chooses the best localization algorithm, can be implemented on top of these localization algorithms. There can be two approaches for the implementation of this decider.

- Decide at the beginning: Decider will work for the first t time steps and decide the best algorithm for the rest of tracking.
- Decide at any time: Decider will work at all time steps and decide to change or not the localization algorithm.

The first approach is easier to realize but it will not solve the problem of targets changing their trajectory by time. The second approach causes an overhead in computational processing in a sensor node. The key criteria in a decider implementation is that it should work in a distributed fashion. All sensor nodes should give the same decision of changing or not changing the localization algorithm locally based on the information received from the network.

Not only the target trajectory but also the predicted localization error, the number of sensor nodes that can participate in collaboration and the density should be used as input to the decision process. For instance, we know that the CCL works better than the CIF-MMI for low participation to collaboration. A decider can decide to use CCL algorithm if it is likely that the number of sensor nodes detecting and participating to the collaboration is low.

6.1.1. Using Perfect Decider

In this section, we aim to show the maximum gain we can achieve if we are able to implement a perfect decider. Perfect decider always selects the localization algorithm that will give the minimum mean error in time step t. In other words, this section will illustrate the lower bound for the decider.

Figure 6.1 illustrates a sinusoidal moving target in a medium dense network, consists of 550 sensor nodes and 200m $\times$ 200m sensing area. In Figure 6.2, it is shown that perfect decider performs much better than all localization algorithms. Using a perfect decider gives 23.5 per cent better localization accuracy in average of all energy levels than the best performing localization algorithm.

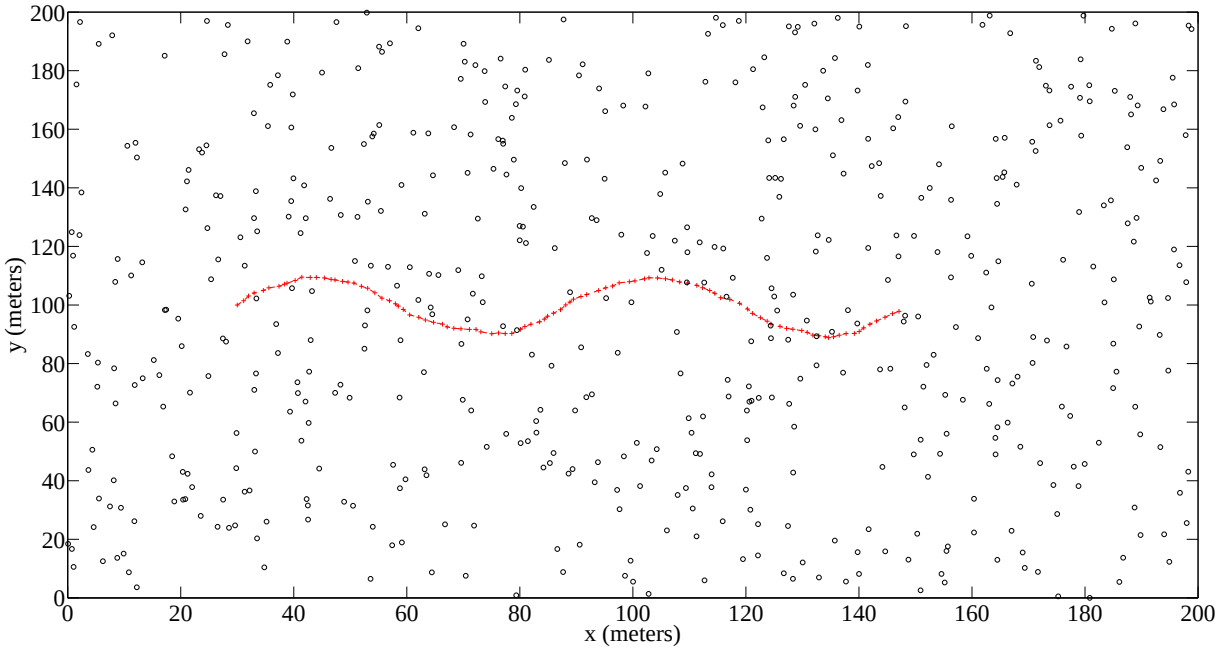

Figure 6.1. Illustration of the medium dense network scenario with sinusoidal target

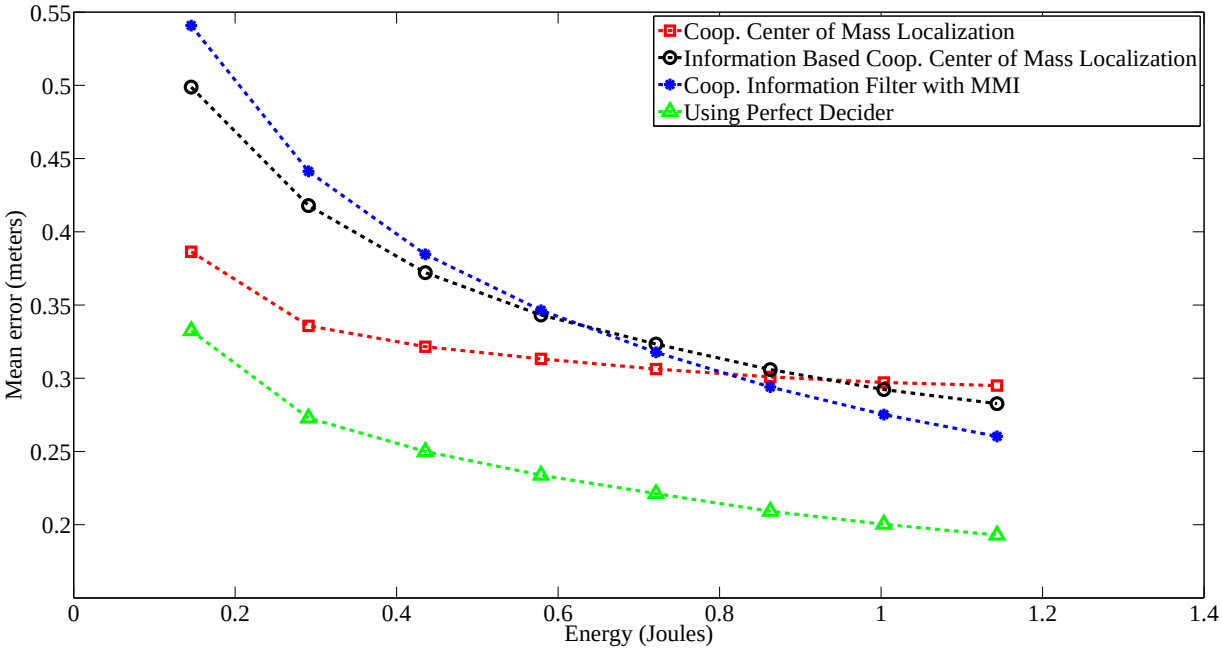

Figure 6.2. Effect of the perfect decider for medium dense networks with sinusoidal target

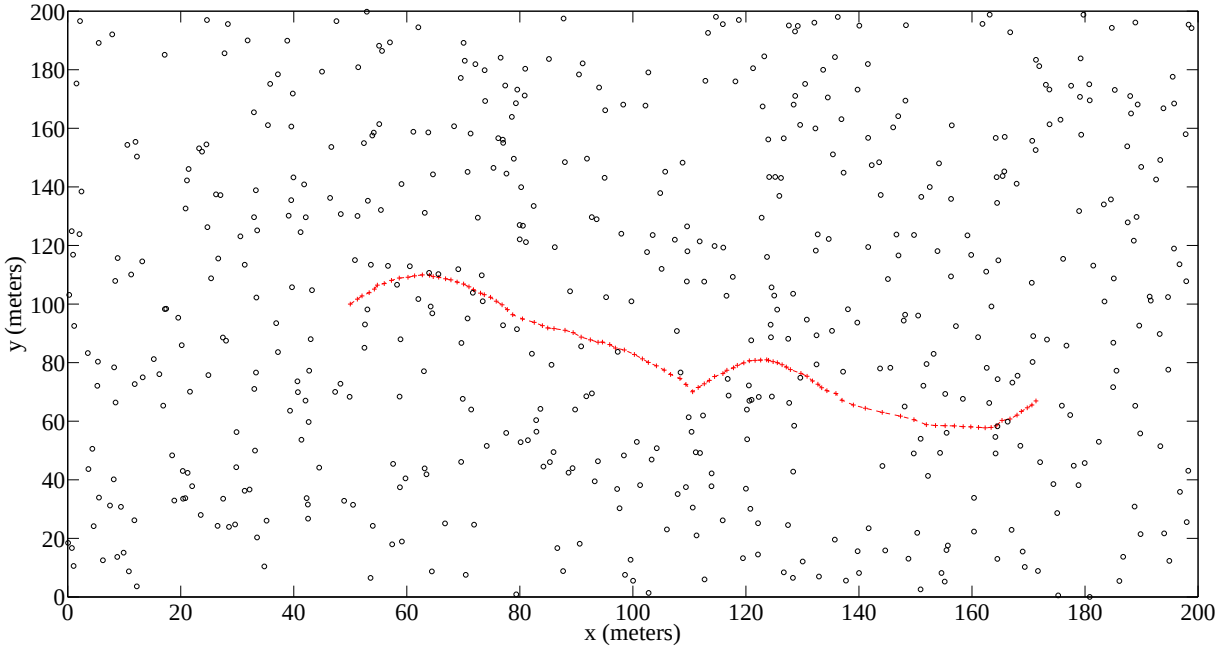

Figure 6.3. Illustration of a maneuvering target in a medium dense network

Figure 6.3 illustrates a maneuvering target in a medium dense network, consists of 550 sensor nodes and 200m × 200m sensing area. In Figure 6.4, it is shown that perfect decider performs much better than all localization algorithms. Using a perfect decider gives 23.5 per cent better localization accuracy in average of all energy levels than the best performing localization algorithm.

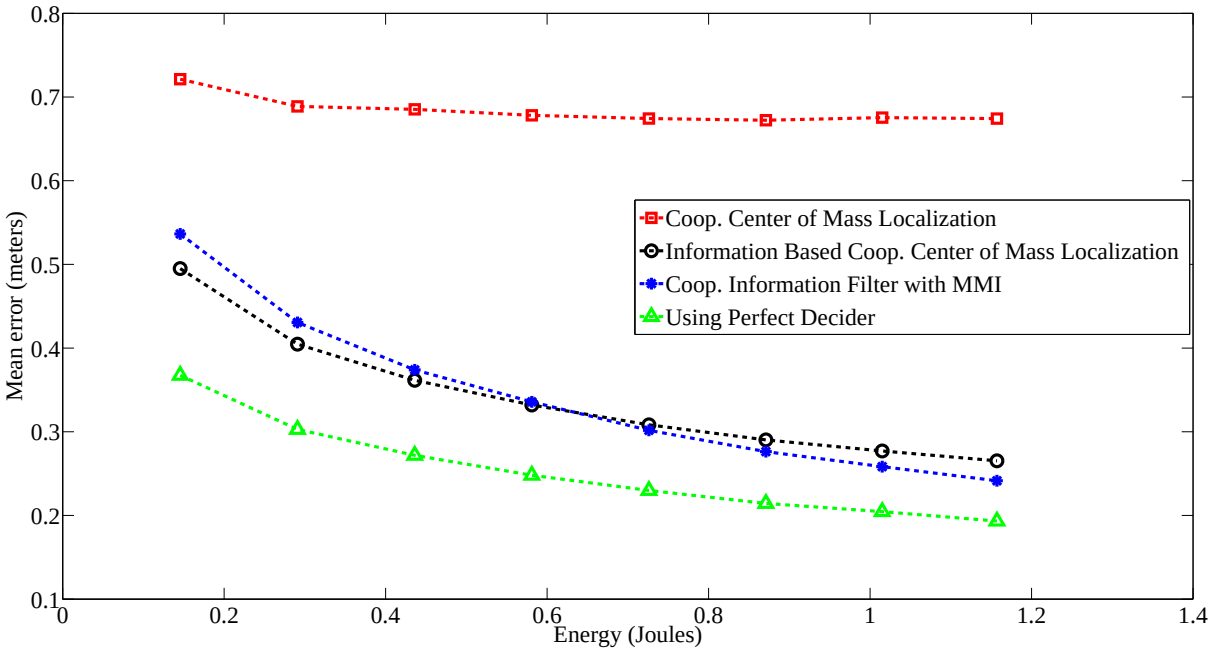

Figure 6.4. Effect of the perfect decider for maneuvering target

REFERENCES

1. Onel, T., C. Ersoy, and H. Delic, "Information Content-Based Sensor Selection for Collaborative Target Tracking", *EUSIPCO, Florance, Italy*, September 2006.

2. Onel, T., C. Ersoy, and H. Delic, "On Collaboration in a Distributed Multi-Target Tracking Framework", *IEEE International Conference on Communications*, pp. 3265 – 3270, June 2007.

3. Akyildiz, I., W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks", *Communications Magazine, IEEE*, Vol. 40, No. 8, pp. 102 – 114, August 2002.

4. Akyildiz, I. F., W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey", *Computer Networks*, Vol. 38, No. 4, pp. 393–422, March 2002.

5. Liu, J., M. Chu, and J. E. Reich, "Multitarget Tracking in Distributed Sensor Networks", *Signal Processing Magazine, IEEE*, Vol. 24, No. 3, pp. 36–46, May 2007.

6. Zhao, F., J. Shin, and J. Reich, "Information-driven dynamic sensor collaboration", *Signal Processing Magazine, IEEE*, Vol. 19, No. 2, pp. 61–72, March 2002.

7. Scheunert, U., H. Cramer, B. Fardi, and G. Wanielik, "Multi Sensor based Tracking of Pedestrians: A Survey of Suitable Movement Models.", *Intelligent Vehicles Symposium, IEEE*, pp. 774 – 778, June 2004.

8. Brooks, R. R., P. Ramanathan, and A. M. Sayeed, "Distributed target classification and tracking in sensor networks", *Proceedings of the IEEE*, Vol. 91, No. 8, pp. 1163–1171, January 2003.

9. Chong, C.-Y., F. Zhao, S. Mori, and S. Kumar, "Distributed tracking in wireless ad

hoc sensor networks", *Proceedings of the Sixth International Conference of Information Fusion*, Vol. 1, pp. 431–438, July 2003.

10. Cetin, M., L. Chen, J. W. Fisher, A. T. Ihler, R. L. Moses, M. J. Wainwright, and A. S. Willsky, "Distributed fusion in sensor networks", *Signal Processing Magazine, IEEE*, Vol. 23, No. 4, pp. 42–55, July 2006.

11. Hall, D. L. and J. Llinas, "An introduction to multisensor data fusion", *Proceedings of the IEEE*, Vol. 85, No. 1, pp. 6–23, January 1997.

12. Smith, D. and S. Singh, "Approaches to Multisensor Data Fusion in Target Tracking: A Survey", *Knowledge and Data Engineering, IEEE Transactions on*, Vol. 18, No. 12, pp. 1696–1710, 2006.

13. Suganya, S., "A Cluster-Based Approach for Collaborative Target Tracking in Wireless Sensor Networks", *First International Conference on Emerging Trends in Engineering and Technology, ICETET*, pp. 276–281, July 2008.

14. Wang, Z., H. Li, X. Shen, X. Sun, and Z. Wang, "Tracking and Predicting Moving Targets in Hierarchical Sensor Networks", *IEEE International Conference on Networking, Sensing and Control, ICNSC*, pp. 1169–1173, April 2008.

15. Balasubramanian, S., I. Elangovan, S. Jayaweera, and K. Namuduri, "Distributed and collaborative tracking for energy-constrained ad-hoc wireless sensor networks", *IEEE Wireless Communications and Networking Conference*, Vol. 3, pp. 1732–1737, March 2004.

16. Balasubramanian, S., S. Jayaweera, and K. Namuduri, "Energy-aware, collaborative tracking with ad-hoc wireless sensor networks", *IEEE Wireless Communications and Networking Conference*, Vol. 3, pp. 1878–1883, March 2005.

17. Li, D., K. D. Wong, Y. H. Hu, and A. M. Sayeed, "Detection, Classification and Track-

ing of Targets in Distributed Sensor Networks", *IEEE Signal Processing Magazine*, pp. 17–29, 2002.

18. Zoghi, M. and M. Kahaei, "Sensor selection for target tracking in WSN Using Modified INS algorithm", *3rd International Conference on Information and Communication Technologies: From Theory to Applications*, pp. 1–6, April 2008.

19. Vuran, M. and I. Akyildiz, "Spatial correlation-based collaborative medium access control in wireless sensor networks", *Transactions on Networking, IEEE/ACM*, Vol. 14, No. 2, pp. 316–329, April 2006.

20. Zoghi, M. and M. Kahaei, "Target tracking in collaborative sensor networks by using a decentralized leader-based scheme", *9th International Symposium on Signal Processing and Its Applications, ISSPA*, pp. 1–4, February 2007.

21. Pahalawatta, P., T. Pappas, and A. Katsaggelos, "Optimal sensor selection for video-based target tracking in a wireless sensor network", *International Conference on Image Processing, ICIP*, Vol. 5, pp. 3073–3076, October 2004.

22. Arroyo-Valles, R., S. Pino-Povedano, J. Cid-Sueiro, and F.-J. Serrano, "Distributed target tracking with selective transmitters in energy-constrained sensor networks", *International Conference on Intelligent Sensors, Sensor Networks and Information Processing, ISSNIP*, pp. 67–72, December 2008.

23. Liu, X., C. Zhang, and J. Hu, "Adaptive Weights Weighted Centroid Localization Algorithm for Wireless Sensor Networks", *4th International Conference on Wireless Communications, Networking and Mobile Computing, WiCOM*, pp. 1–4, October 2008.

24. Wang, X. and S. Wang, "Peer-to-Peer Collaborative Signal Processing for Target Tracking in Wireless Sensor Networks", *Grid and Cooperative Computing, GCC*, pp. 160–163, Oct. 2006.

25. Wang, X., S. Wang, D.-W. Bi, and J.-J. Ma, "Distributed Peer-to-Peer Target Tracking in Wireless Sensor Networks", *Sensors*, Vol. 7, No. 6, pp. 1001–1027, June 2007.

26. Wang, X. and S. Wang, "Collaborative signal processing for target tracking in distributed wireless sensor networks", *J. Parallel Distrib. Comput.*, Vol. 67, No. 5, pp. 501–515, February 2007.

27. Wang, X., S. Wang, and B. Daowei, "Distributed Visual-Target-Surveillance System in Wireless Sensor Networks", *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics,*, March 2009.

28. Li, Q., L. Yu, X. Xu, and Z. Zhao, "An Algorithm for Target Localization in Sensor Networks Based on Overlap Area Boundary of Sensor Detection", *The 9th International Conference for Young Computer Scientists, ICYCS*, pp. 469–474, November 2008.

29. Zhao, Z., T. Rong Li, and V. Jilkov, "Best linear unbiased filtering with nonlinear measurements for target tracking", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 40, No. 4, pp. 1324–1336, Oct. 2004.

30. Sheng, X., Y.-H. Hu, and P. Ramanathan, "Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor network", *Fourth International Symposium on Information Processing in Sensor Networks, IPSN*, pp. 181–188, April 2005.

31. Yick, J., B. Mukherjee, and D. Ghosal, "Analysis of a prediction-based mobility adaptive tracking algorithm", *2nd International Conference on Broadband Networks, BroadNets*, Vol. 1, pp. 753–760, Oct. 2005.

32. Kam, C. and W. Hodgkiss, "Distributed Target Tracking in a Wireless Sensor Network", *Fortieth Asilomar Conference on Signals, Systems and Computers, ACSSC*, pp. 1999–2003, 29 2006-Nov. 1 2006.

33. Zhang, Z., "Implementation of Distributed Kalman Filter Based on Mutual Coupled Oscillators in Sensor Networks", *4th International Conference on Wireless Communications, Networking and Mobile Computing, WiCOM*, pp. 1–4, October 2008.

34. Khan, U. A. and J. M. F. Moura, "Distributed Kalman Filters in Sensor Networks: Bipartite Fusion Graphs", *14th Workshop on Statistical Signal Processing, SSP '07*, pp. 700–704, Aug. 2007.

35. Yang, P., R. Freeman, and K. Lynch, "Distributed Cooperative Active Sensing Using Consensus Filters", *IEEE International Conference on Robotics and Automation*, pp. 405–410, April 2007.

36. Olfati-Saber, R. and J. Shamma, "Consensus Filters for Sensor Networks and Distributed Sensor Fusion", *44th IEEE Conference on Decision and Control and European Control Conference, CDC-ECC '05*, pp. 6698–6703, December 2005.

37. Olfati-Saber, R., "Distributed Kalman Filter with Embedded Consensus Filters", *44th IEEE Conference on Decision and Control and European Control Conference, CDC-ECC '05*, pp. 8179–8184, December 2005.

38. Olfati-Saber, R., "Distributed Kalman filtering for sensor networks", *46th IEEE Conference on Decision and Control*, pp. 5492–5498, Dec. 2007.

39. Olfati-Saber, R., "Distributed Tracking for Mobile Sensor Networks with Information-Driven Mobility", *American Control Conference, 2007. ACC '07*, pp. 4606–4612, July 2007.

40. Sandell, N. and R. Olfati-Saber, "Distributed data association for Multi-target tracking in sensor networks", *47th IEEE Conference on Decision and Control, CDC*, pp. 1085–1090, Dec. 2008.

41. Olfati-Saber, R. and N. Sandell, "Distributed tracking in sensor networks with limited

sensing range", *American Control Conference, 2008*, pp. 3157–3162, June 2008.

42. Oh, S., S. Sastry, and L. Schenato, "A Hierarchical Multiple-Target Tracking Algorithm for Sensor Networks", *Proceedings of the 2005 IEEE International Conference on Robotics and Automation, ICRA*, pp. 2197–2202, April 2005.

43. McErlean, D. and S. Narayanan, "Distributed detection and tracking in sensor networks", *Conference Record of the Thirty-Sixth Asilomar Conference on Signals, Systems and Computers*, Vol. 2, pp. 1174–1178, November 2002.

44. Tinati, M. and T. Rezaii, "Multi-target Tracking in Wireless Sensor Networks Using Distributed Joint Probabilistic Data Association and Average Consensus Filter", *International Conference on Advanced Computer Control, ICACC*, pp. 51–56, January 2009.

45. Onel, T., C. Ersoy, and H. Delic, "An Information-Controlled Transmission Power Adjustment Scheme for Collaborative Target Tracking", *11th IEEE Symposium on Computers and Communications, ISCC*, pp. 294–300, June 2006.

46. Onel, T., C. Ersoy, and H. Delic, "Information Content-Based Sensor Selection and Transmission Power Adjustment for Collaborative Target Tracking", 2009, to appear in IEEE Transactions on Mobile Computing.

47. Onel, T., E. Onur, C. Ersoy, and H. Delic, *Advances in Sensing with Security Applications*, Vol. 2 of *NATO Security through Science Series*, chapter Wireless Sensor Networks for Security Issues and Challenges, pp. 95–119, Springer Netherlands, 2006.

48. Maybeck, P. S., *Stochastic models, estimation, and control*, Mathematics in Science and Engineering, Academic Press, Inc., 1979.

49. Bar-Shalom, Y. and X.-R. Li, *Estimation with Applications to Tracking and Navigation*, John Wiley & Sons, Inc., New York, NY, USA, 2001.

50. Advantaca, “TWR-ISM-002-I Micro-power Impulse Radar (MIR)”, http://www.advantaca.com, 2009.