Cooperative Localization and Mapping in Sparsely-Communicating Robot Networks

by

Keith Yu Kit Leung

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Aerospace Science and Engineering
University of Toronto

Abstract

Cooperative Localization and Mapping in Sparsely-Communicating Robot Networks

Keith Yu Kit Leung

Doctor of Philosophy

Graduate Department of Aerospace Science and Engineering

University of Toronto

2012

This thesis examines the use of multiple robots in cooperative simultaneous localization and mapping (SLAM), where each robot must estimate the poses of all robots in the team, along with the positions of all known landmarks. The robot team must operate under the condition that the communication network between robots is never guaranteed to be fully connected. Under this condition, a novel algorithm is derived that allows each robot to obtain the centralized-equivalent estimate in a decentralized manner, whenever possible. The algorithm is then extended to a decentralized and distributed approach where robots share the computational burden in considering different data association hypotheses in generating the centralized-equivalent consensus estimate.

Dedication

This thesis is dedicated to my parents, Li Li Chen, Leung Yat Chue, and my beloved Selina Leung.

Acknowledgements

First and foremost, I need to thank my supervisors, Tim Barfoot, and Hugh Liu, for their guidance in the past four years at the University of Toronto Institute for Aerospace Studies (UTIAS). My time at UTIAS was most enjoyable and motivating. The research presented in this thesis initiated and evolved from Tim and Hugh's vision of an air-ground cooperative multi-robot system. They gave me the freedom of investigating different problems associated with cooperative systems, and I decided to focus my study on the state estimation aspect. As part of my study, Tim and Hugh provided me with the resources to construct a fleet of robots, which was essential in conducting my research. They also assisted greatly with my career development by funding my attendance to numerous international conferences, and allowing me to take on side projects, such as my involvement with the lunar analogue field deployments. I hope that that I have generated enough academic currency (i.e., papers) to repay the both of you for your continuing support. I also hope that I have contributed in the development of both the Autonomous Space Robotics Lab, and the Flight Systems Control Lab.

I would like to extend my gratitude to Professor Chris Damaren, the chair of my doctoral examination committee, for his support during my studies. I would also like to thank Professor Bruce Francis from the University of Toronto, and Professor Hong Zhang from the University of Alberta for reviewing this thesis.

The experiment dataset presented in this thesis was prepared with the help of the members in the Autonomous Space Robotics Lab. Thank you Paul, Braden, Chi, Colin, Andrew, Hang, Rehman, for their precision robot driving skills, and for not playing 'tag' and 'capture the flag' with my robots until my experiments were completed. A special mention goes to Yoni Halpern, the summer student who helped with a large part of experiment preparations and also assisted in conducting the experiments.

Lastly, I want to thank the National Science and Engineering Research Council (NSERC) of Canada for their financial support through the PGS-D award. I also need to thank the funding agency for the Ontario Graduate Scholarship (OGS).

Contents

List of Figures

List of Algorithms

Notation

$k_1 : k_2$ time interval from k_1 to k_2 inclusive.

$\mathbf{h_m}(\cdot)$ landmark measurement function.

ψ_k landmark measurement noise at timestep k.

N set of all robots.

M set of all landmarks.

$N_{i,k}$ set of robots known to robot i at timestep k.

$M_{i,k}$ set of landmarks known to robot i at timestep k.

M_k set of landmarks that have been observed by at least one robot up to timestep k.

$X_{i,k}$ set of states (robots and landmarks) known to exist by robot i at timestep k.

$|Q|$ the cardinality (number of members) of set Q.

$\bigcup_j$ set union operator on all sets indexed by j.

$\mathsf{bel}(\cdot)$ belief, a probability density function representing the likelihood of a set of states given an initial belief, as well as odometry and measurement information.

$v(i, k)$ a node representing robot i at timestep k in an information flow graph.

$v_1 \xrightarrow{\textbf{path}} v_2$ a path that exists between nodes v_1 and v_2 on an information flow graph.

$S_{i,k}$ set of all estimates, odometry data, and measurement data known to robot i at timestep k after communication with other robots within communication range.

s^i metric associated with the distributed contribution from robot i.

k timestep.

$\mathbf{x}_{i,k}$ state of robot or landmark i at timestep k.

$\mathbf{u}_{i,k}$ odometry information of robot i at timestep k.

$\mathbf{g}(\cdot)$ process (state transition) function.

ϵ_k process noise at timestep k.

$\mathbf{y}_{i,k}^{j,i}$ relative measurement of robot or landmark j with respect to robot i, expressed in the local reference frame of robot i at timestep k.

$\mathbf{h}(\cdot)$ robot measurement function.

δ_k robot measurement noise at timestep k.

$d_k^{j,i}$ distance between robot or landmark i and robot or landmark j at timestep k.

r_{obs} measurement range limit.

r_{comm} communication range limit.

X_k set of states (robots and landmarks) known to exist by all robots at timestep k.

$X_{Q,k}$ set of states (robots and landmarks) known to exist by robots in set Q at timestep k.

U_k set of odometry information from all robots at timestep k.

$U_{Q,k}$ set of odometry information from robots in set Q at timestep k.

Y_k set of all relative measurements made by all robots at timestep k.

$Y_{i,k}$ set of all relative measurements made by robot i at timestep k.

$Y_{Q,k}$ set of all relative measurements made by robots in set Q at timestep k.

$R_{i,k}$ set of robots that is able to exchange information with robot i at time k..

$p(\cdot)$ probability density function (pdf).

$\mathcal{G}_{k_1:k_2}$ information flow graph from timestep k_1 to k_2.

$S_{i,k}^{-}$ set of all estimates, odometry data, and measurement data known to robot i at timestep k before communication with other robots within communication range.

$C(k_c, k_e)$ Checkpoint occurring at timestep k_c and existing at timestep k_e.

$C_p(k_{c,i}, k_{e,i})$ Partial checkpoint occurring at timestep $k_{c,i}$ and existing at timestep $k_{e,i}$.

$\mathsf{bel}\left(X_{k_1:k_2}\right)^i$ distributed contribution, an estimate calculated based on a particular data association hypothesis.

$\mathsf{bel}\left(X_{k_1:k_2}\right)^*$ consensus estimate.

$f^a(\cdot)$ arbitration function, which produces a consensus estimate from distributed contributions).

$C^d(k_s, k_c, k_d)$ distributed checkpoint that exists at k_d, when $\mathsf{bel}\left(X_{k_s:k_c}\right)^*$ is obtainable by all robots.

$C^d_p(k_{s,i}, k_{c,i}, k_{d,i})$ distributed partial checkpoint that exists at $k_{d,i}$, when $\mathsf{bel}\left(X_{k_{s,i}:k_{c,i}}\right)^*$ is obtainable by a robot.

$O(\cdot)$ big O notation for complexity analysis.

Acronyms

AUV	Autonomous underwater vehicle.
AVATAR	Autonomous Vehicle Aerial Tracking and Reconnaissance.
DARCES	Data-Aligned Rigidity-Constrained Exhaustive Search.
DARPA	Defence Advanced Research Project Agency.
EKF	Extended Kalman Filter.
LAGR	Learning Applied to Ground Vehicle.
LIDAR	Light detection and ranging.
MAP	Maximum a posteriori.
MARS	Mobile Autonomous Robot Software.
NEES	Normalized-estimator-error-squared.
NTP	Network Time Protocol.
OOSM	Out-of-sequence measurements.
PDF	Probability density function.
PerceptOR	Perception of Offroad Robotics.
PF	Particle filter.
rms	Root-mean-squared.
SEIF	Sparse Extended Information Filter.
SLAM	Simultaneous localization and mapping.
sonar	Sound navigation and ranging.
UAV	Unmanned aerial vehicle.
UKF	Unscented Kalman Filter.

Chapter 1

Introduction

He that does good to another does good also to himself.

Lucius Annaeus Seneca (4 BC-65)
Roman philosopher and playwright.

Before the 1990s, mobile robotic research focused on achieving autonomous navigation with a single robot. With the advance and maturity of techniques for single-robot localization, mapping, and path planning, researchers began to examine the possibility of cooperative multi-robot systems.

It is often claimed that the benefits of deploying a multi-robot system include redundancy against the failure of a robot, the implementation of complex strategies that require more than a single robot, a decrease in task completion time, and increase in efficiency through cooperation. However, before these claims can be made, we need to examine the architecture of the specific multi-robot system in terms of computation and decision making for the task that the system intends to perform (such as localization and mapping, or state estimation). In general, we can classify a multi-robot system as *centralized* or *decentralized*, and *distributed* or *non-distributed.*

Definition 1: *A* centralized multi-robot system *is a group of robots in which the computation for a task is performed only by a specific robot in the team or by an external computer.*

Definition 2: *A* decentralized multi-robot system *is a group of robots in which the computation for a task can be performed by any one or more robots in the team.*

Definition 3: *A* distributed multi-robot system *is a group of robots in which the computation for a task is divided amongst the robots in the team.*

Note that whether a system is centralized or decentralized is independent of whether this system is distributed or non-distributed. For instance, we can have a distributed system where each robot performs a part of the computation for a certain task, but a centralized processor is required to combine the computations from each robot to determine the final result.

The benefit of redundancy (against the failure of a robot) exists when a decentralized system architecture is adopted, and allows all other robots to proceed with their computations even if one robot fails. In terms of improving efficiency, a distributed system can be used to make use of the computational resources available from multiple robots. Note that it is possible for a system to be both decentralized and distributed. This implies that the computation for a task does not depend on a specific robot, and can be divided amongst a number of robots. However, communication is an important aspect of cooperative systems that have a large impact on performance. This is the main theme of this thesis.

1.1 Applications of Multi-Robot Systems

Multi-robot systems can be used as a network of mobile sensors to increase situational or environmental awareness. For instance, robots with different sensing capabilities can use relative measurements to help each other localize [1]. Sensing is one of the reasons why defence agencies such as *Defence Advanced Research Project Agency* (DARPA) are interested in the use of multi-robot systems. One of the earliest studies on deploying multi-robot systems for military surveillance was the *Autonomous Vehicle Aerial Tracking and Reconnaissance* (AVATAR) testbed supported by the DARPA Tactical Mobile Robot program, which showed the potential of using multi-robot systems (involving aerial and ground vehicles) in cooperative localization and surveillance [2, 3]. The DARPA *Perception of Offroad Robotics* (PerceptOR) is a similar program but focused on the autonomous navigation of a ground vehicle in unknown terrain with the help of an autonomous “flying eye” helicopter [4, 5, 6]. Another DARPA initiated program, designated *Mobile Autonomous Robot Software* (MARS), again aimed at the development of robust multi-robot systems for reconnaissance, surveillance, as well as search and rescue. Part of the work in this program looked at maintaining communication connectivity between robots during mapping [7, 8, 9] when robots are required to operate in urban canyons. Cooperative target localization and surveillance was another component of the MARS program [10, 11, 12, 13].

Although the use of multi-robot systems to date is heavily linked with surveillance applications, there are a number of notable humanitarian applications as well. These include: (i) the use of autonomous aerial and ground vehicles for the detection and removal of landmines [14, 15, 16], (ii) the use of *unmanned aerial vehicles* (UAVs) for detecting forest fires [17, 18, 19], (iii) search and rescue, and (iv) disaster management [20, 21, 22, 23, 24, 25]. The use of an entomopter robot for assisting a planetary rover in exploration has also been proposed in the past [26].

1.2 Communication Limitations

Regardless of the type of system architecture or application, communication is an important aspect of any cooperative multi-robot system. This ability to mutually share information leads to performance gains in many tasks. Some of the earlier research on the sharing of information between sensors (i.e., sensor fusion) for state estimation purposes were conducted by Berg and Durrant-Whyte [27, 28]. They used the information filter to centrally incorporate measurements from many sensors. Grime and Durrant-Whyte [29] extended this work and considered a decentralized approach and allowed measurement information to be relayed between sensors. As part of this work, the *channel filter* was introduced to ensure that only new information were used to update state estimates. However, this was shown to work only in an acyclic network (with no loops). The approach was later extended by Utete and Durrant-Whyte [30] to work for arbitrary network topologies that are fully connected at all times. A fully connected network implies that it is always possible for information from one robot to reach another robot. This is an assumption that is common in past research on multi-robot systems. Abandoning this assumption, we will explore the impact of operating a multi-robot system under a dynamic communication network that is never guaranteed to be fully connected. We will specifically examine the impact of this in the context of robot localization and mapping (by recursive filtering). This is a problem that a multi-robot system may need to deal with while operating over a large area, or where occlusions obstruct and limit the communication range. One solution to this communication problem is to constrain robots to remain in each other's communication range limit [31]. We will take a different approach that does not impose any spatial constraints on the robots.

One of the problems with performing state estimation in a dynamic network is that the sequence in which information is communicated to a robot is often out of order, and this leads to the *out-of-sequence measurements* (OOSM) problem. Bar-Shalom [32][33] examined some possible remedies for treating OOSM in state estimation using a Kalman filter (also known as the negative timestep problem), and applied this to target tracking.

It was shown that for a missed measurement, the only way to properly incorporate the information into an estimate is to sequentially reprocess all following measurements. This is an important concept to remember. In situations where past measurements are no longer available, it is possible to proceed with measurements that are available but this leads to sub-optimal estimates (in the sense that we can obtain a better estimate if we have more measurements) [34].

Another issue with performing state estimation in a dynamic network is the *cyclic update* problem, where robots repeatedly use a measurement. For instance, this occurs when one robot provides information to another robot to update its state estimate, which in turn is provided back to the first robot for updating its state estimate. This causes the resulting state estimate to be inconsistent (over-confident). Previously, we mentioned how the channel filter was applied in sensor networks for this reason, but it was only shown to work for fully connected networks. Howard et al. [35, 36] introduced the *dependency tree* as a remedy to this problem for a multi-robot system but it is not guaranteed to work in all cases.

In this thesis, we will develop an approach to state estimation in a dynamic network that is never guaranteed to be fully connected, and which avoids both the OOSM and cyclic update problems. We call this the *centralized-equivalent approach*, because it is able to allow robots to recover an estimate equivalent to that produced by a centralized estimator whenever possible.

1.3 The Centralized-Equivalent Approach

The problem that we are trying to solve in this thesis is *cooperative simultaneous localization and mapping* (SLAM) with a multi-robot system under the assumption that the communication network for information exchange is never guaranteed to be fully connected (i.e., in the worst-case scenario, the network is never fully connected). In this problem, we expect each robot in the team to estimate the poses (positions and orientations) of all robots, as well as the positions of all known landmarks. We will present in this thesis a novel decentralized solution to this problem that we call the centralized-equivalent approach, because it enables all robots to recover the centralized-equivalent estimates when possible. That is, the estimates are exactly the same as ones produced by a centralized estimator, even when the network is never fully connected.

We began this chapter with the epigraph “He that does good to another does good to himself” from Lucius Annaeus Seneca, a Roman philosopher and playwright. This philosophy is often used as the basis and justification for cooperation and teamwork.

Ironically, with our centralized-equivalent approach to decentralized cooperative SLAM, we show that “He that does good to himself does good to another”. In state estimation, recursive filtering methods are used to limit computational requirement and memory usage. These methods are based on the assumption that the Markov property is valid, which allows a robot to discard measurements that have already been used in generating a state estimate. While operating in a dynamic network, the brute-force approach to obtaining centralized-equivalent estimates would be to keep all measurements and communicate everything with all the robots within communication range until memory usage is at the maximum capacity [37]. Alternatively, the Markov property can be applied but a robot needs to ensure that other robots no longer need the information that it wishes to discard. This thesis will prove that if a robot invokes the Markov property whenever possible (i.e., without consideration of what other robots need) to reduce its own memory use, it does not affect other robots' ability to also obtain the centralized-equivalent estimate. This (in contrast to the work in [38] for instance) implies that a robot does not need to keep track of what other robots know.

We will begin to explore the decentralized cooperative SLAM problem in Chapter 2 by first ignoring the need to estimate the position of landmarks. This simplification allows us to focus our attention on the effect of operating in a dynamic network in the context of state estimation, and essentially reduces the decentralized cooperative SLAM problem into the decentralized cooperative localization problem. Once we have developed several important theorems for our centralized-equivalent approach based on the decentralized cooperative localization scenario, we will apply them in Chapter 3 to the decentralized cooperative SLAM problem, and detail extensive hardware experiments that validate our approach. Finally, in Chapter 4, we look at how estimation with multiple data association hypotheses can be distributed amongst robots. We show a distributed and decentralized solution to SLAM that still provides the centralized-equivalent estimate to all robots whenever possible, while exploiting the distributed computing platform inherent in a multi-robot system.

Chapter 2

Decentralized Cooperative Localization

If you want to be incrementally better: Be competitive. If you want to be exponentially better: Be cooperative.

Unknown author

In this chapter, we begin our study of the decentralized simultaneous localization and mapping problem by temporarily ignoring the need to estimate a map of the environment[1]. This simplification enables us to focus on handling the network connectivity constraint. That is, it allows us to more easily examine how centralized-equivalent estimates can be obtained when the communication network is never guaranteed to be fully connected at any time. This simplification essentially transforms our cooperative SLAM problem into a *cooperative localization*[2] problem, where robots only use relative measurements to estimate the pose of each other. In Chapter 3, we will reintroduce the map back into our problem. We will first provide an overview of the recent advances in cooperative localization. This is followed by the mathematical formulation of our problem. We will introduce the novel concepts of a *checkpoint* and a *partial checkpoint*, which are vital in satisfying the requirement that our centralized-equivalent approach has to work even when the network is never fully connected. Our approach is validated and evaluated through extensive simulations.

[1]The work in this chapter was presented at the IEEE International Conference on Robotics and Automation 2009 [39], and also published in the IEEE Transaction on Robotics in 2010 [40]. Since then, it has been extended by Nerurkar and Roumeliotis [41] who imposed bandwidth constraints on the network.

[2]In earlier literature, cooperative localization is often referred to as collaborative localization.

2.1 An Overview Cooperative Localization

The work by Kurazume et al. [42] is one of the earliest in using multiple robots for cooperative localization. They are the first to introduce the idea of using robots as features in the environment, which can be used by other robots for localization. The system introduced in [42] is the first version of their cooperative positioning system (CPS-I). In their experiments with CPS-I, three robots were used. While one robot moved, the other two remained stationary, and essentially served as static landmarks as the moving robot tried to localize. This created a 'leap-frogging' motion pattern, which was later adopted by a number of other researchers such as Rekleitis et al. [43]. In their work with two robots, one robot remained stationary and tracked the other moving robot with a *light detection and ranging* (LIDAR) sensor while it performed mapping at the same time. The stationary robot also guided the moving robot safely through the mapped region. This work was extended by using *sound navigation and ranging* (sonar) sensors on the robots [44].

Positioning is deterministic in CPS-I as it does not account of measurement noise in the the measurement model. That is not until the second version of the cooperative positioning system (CPS-II), which includes a number of other improvements over the first system [45, 46]. In the final version (CPS-III), Kurazume and Hirose [47] looked for optimal motion strategies for improving localization performance in large open areas, as well as areas with a large number of occlusions. On the note of optimal motion strategies, Trawny and Barfoot [48] also examined this problem with three robots. They found that there are more optimal trajectories for reducing estimate uncertainty as compared to when the robots remained in an equilateral triangle formation. More recently, Tully et al. [49] used three robots from the DARPA *Learning Applied to Ground Vehicle* (LAGR) program and examined optimal 'leap-frogging' motion strategies to maximize information gain.

All variants of the cooperative positioning system are essentially centralized systems, where the processing of the state estimate is performed by one computer. With the availability of multiple computers on multiple robots, researchers began to look at how some of the computation for pose estimation can be distributed amongst the team of robots. Sanderson [50] had some success in distributing estimate calculations but had to make simplifications by ignoring orientation uncertainty in state propagation and in updating the estimate with relative position measurements. Madhavan et al. [51] enabled the distribution of computations by only requiring each robot to run an *Extended Kalman Filter* (EKF) for estimating its own pose. When robots measured each other, they exchanged their self estimates to calculate the predicted measurement in the update step of the EKF. Each robot then returned to estimating their own pose. The problem with this method is that it ignores the correlation between the pose estimates of two robots when a relative measurement is made between them. In other words, the estimates produced by each robot are sub-optimal with respect to a centralized estimator. Fox et al. [52] took a similar approach to [51] in that they only required each robot to maintain an estimate of their own pose. The difference is that a *particle filter* (PF) [53] is used as the filtering method instead of an EKF. When measurements are made between two robots, a method known as the density tree approach is used to temporarily combine the *probability density functions* (PDFs) (i.e., the particles) from the two robots to perform importance sampling.

One of the most notable advances in cooperative localization was achieved by Roumeliotis and Bekey [54, 55], who performed distributed multi-robot localization by decomposing the EKF into a number of filters that can perform the prediction step of the EKF locally on each robot. Their work shows how the propagation of the covariance matrix can be factored using singular value decomposition such that the factored terms can be computed by each robot individually using their own odometry data. Furthermore, the factored terms only need to be combined before a measurement update, which then requires full network connectivity (i.e., all robots need to communicate with each other for the EKF correction step). This approach is able to produce the same result as a centralized estimator. In [55], relative range and bearing measurements are used for estimate updates. Martinelli et al. [56] made an extension to this by showing how relative bearing-only, relative-distance-only, and relative-orientation-only measurements can be used in the updates. Recently, Bailey et al. [57] also proposed a method which uses the information form of the EKF for cooperative localization. In their approach, robots process odometric data locally, while relative measurements between robots are processed by a central processor. Also, Nerurkar et al. [58] performed cooperative localization using a distributed *maximum a posteriori* (MAP) estimator. These approaches are able to replicate the result of a centralized estimator, but require a fully connected network.

Researchers continued to look for fully distributed approaches to cooperative localization, but many ran into the problem of producing inconsistent and over-confident estimates because correlations between robot pose estimates were not properly taken into accounted. For instance, Karam et al. [59] had each robot in a team maintain an estimate for the entire team and exchange estimates with each other when they are within communication range. A robot would use estimates from other robots as a measurement in performing EKF updates. As estimates passed back and forth between robots, cyclic updates occurred and this led to inconsistent estimates. Panzieri et al. [60] used an approach known as the interlaced EKF, which consists of a bank of EKFs. Each individual filter only processed a subset of the robot team state. This again is sub-optimal and led to inconsistent estimates due to the decoupling of the team state estimate. Their solution to the inconsistency was to artificially increase measurement noise. Martinelli [61] tried an hierarchical approach by dividing robots into subgroups. A leader in each subgroup is responsible for producing the estimate of the subgroup. Furthermore, leaders themselves also form a subgroup. However, since some of the correlations between subgroups are still ignored, the estimates produced for the team are still sub-optimal and inconsistent.

Another aspect of cooperative localization that researchers have looked at is performance limitation, such as the bound on the uncertainty of an estimate. Roumeliotis and Rekleitis [62] analytically quantified the benefit of cooperative localization, and showed how increasing the number of robots in a team can improve localization performance in terms of reducing uncertainty. However, it was discovered that there are diminishing returns in terms of uncertainty reduction as the number of robots increases. Mourikis and Roumeliotis [63] examined the upper bound in position estimate uncertainty given a sensor model and a graph of relative measurements between robots. They concluded that aside from the number of robots in the team, a robot's own proprioceptive sensors and orientation estimates (and not the number of relative measurements between robots) are the most important factors in determining the increase in position uncertainty.

The method of inter-robot measurements has also received the attention from some researchers. Rekleitis et al. [64] examined how varying sensing paradigm and the number of robots affect localization performance. The sensing paradigms included range-only, bearing-only, range-and-bearing, and full-pose sensing (range, bearing and relative orientation). The results indicated that full-pose measurement provides slightly better results than range-and-bearing measurement, as well as range-only measurement. Increasing the number of robots also showed better localization results. However, bearing-only measurement performed poorly regardless of the number of robots used. Nevertheless, bearing-only measurements continued to be used by other researchers in cooperative localization because the measurements can be obtained using an inexpensive monocular camera. The work by Spletzer et al. [65] is one such example, but they used the dimensions of known targets on each robot to determine the scale factor in their estimates. Another example is the work by Montesano et al. [66], who compared the performance of the EKF and the PF in cooperative localization with bearing-only measurements using two robots.

In terms of application, Bahr et al. [67, 68] recently performed cooperative localization with *autonomous underwater vehicles* (AUVs). Some of the vehicles are surface vehicles while some are submersibles. Their experiments showed how a small number of vehicles with good sensors can aid less capable vehicles with acoustic sensors in the heterogeneous team. To distribute computation, each AUV maintained a bank of filters. Depending on the measurement, only some of the filters are updated. This is done so that when estimates are exchanged with other robots, only the estimate that is uncorrelated with the other robot is communicated to avoid cyclic updates and to maintain consistency.

Other works related to cooperative localization that are worth mentioning include the works by Spletzer and Taylor [69], and Taylor and Spletzer [70]. They examined how relative pose between robots can be determined at a given timestep assuming a measurement model with bounded error. Their solution involved performing optimization over all robot poses, using the relative range and bearing measurements as constraints. Dieudonné et al. [71, 72] showed that deterministic cooperative localization with relative measurements is a NP-hard problem. Note that besides the robotics community, the signals and communication community also have an interest in cooperative localization. However, their focus is mainly on the modelling and use of communication signal strength as a measurement for position estimation [73, 74]. Furthermore, they are often interested in position estimation of static nodes instead of moving robots.

A recent area of interest in cooperative localization is communication and network connectivity. Trawny et al. [75] showed how the communication cost of cooperative localization can be significantly reduced by using quantized measurements. By quantizing measurements to four bits, their work shows that the estimates produced are practically the same compared with using real-valued measurements in a MAP estimator.

In this chapter, we will present another advancement in cooperative localization, which allows the centralized-equivalent estimate to be calculated by all robots whenever possible in a network that is never guaranteed to be fully connected. We will first present the mathematical formulation of the cooperative localization problem, under the sparse network assumption. We will then examine how the distribution of information from one robot to another affects each robot's ability to produce the centralized-equivalent state estimate of the robot team (i.e., the poses of all the robots). From this, we will develop several key theorems that will be applied within an algorithm that allows all robots to obtain the centralized-equivalent estimate whenever possible. Through simulations, we will examine how estimation performance of the algorithm is affected by the number of robots in the team, the size of the workspace, as well as communication limitations.

2.2 Mathematical Formulation

2.2.1 System Model

Before we provide a mathematical formulation for the decentralized cooperative localization problem, we first need to define a system model. In a multi-robot system, let N represent the set that contains the unique identification indices of all robots. The total number of robots corresponds to $|N|$, the cardinality of the set, and we assume that the identification indices of the robots range from 1 to $|N|$. Furthermore, we define $N_{i,k}$ as the set of robots known to robot i at a specific timestep, k.

We assume a general system model for the team of robots:

$$\begin{aligned} \mathbf{x}_{i,k} &= \mathbf{g}\left(\mathbf{x}_{i,k-1}, \mathbf{u}_{i,k}, \epsilon_k\right), \ \forall i \in N & (2.1) \\ \mathbf{y}_{i,k}^{j,i} &= \mathbf{h}\left(\mathbf{x}_{i,k}, \mathbf{x}_{j,k}, \delta_k\right), \ \forall i \in N, \ \forall j \in N, \ i \neq j, \ d_k^{j,i} \leq r_{\mathsf{obs}} & (2.2) \end{aligned}$$

where in this discrete-time system:

k represents the timestep
$\mathbf{x}_{i,k}$ represents the state (pose) of robot i
$\mathbf{u}_{i,k}$ represents the odometry information of robot i
$\mathbf{g}(\cdot)$ is the state transition function
ϵ_k represents the process noise (we do not know the value of this quantity)
$\mathbf{y}_{i,k}^{j,i}$ represents the measurement (e.g., range/bearing) of robot j with respect to robot i in robot i's local reference frame
$\mathbf{h}(\cdot)$ is the measurement function
δ_k is the measurement noise (we do not know the value of this quantity)
$d_k^{j,i}$ is the distance between robot i and j
r_{obs} is the measurement range limit

Robots within the communication range limit, r_{comm}, of each other are able to exchange and relay information, which includes state estimates, odometry data, and measurement data. To facilitate the mathematics presented later in this chapter, we will define some additional notation that represent sets of information. Let

$$X_k = \{\mathbf{x}_{i,k} | i \in N\}$$

represent the set of all robot poses at timestep k. For robots belonging to a subset

$Q \subseteq N$, let

$$X_{Q,k} = \{\mathbf{x}_{i,k} | i \in Q\}$$

represent the set of robot poses at timestep k, for the robots in the subset. Similarly, let

$$U_k = \{\mathbf{u}_{i,k} | i \in N\}$$

represent the set of odometry information from all robots at timestep k, and let

$$U_{Q,k} = \{\mathbf{u}_{i,k} | i \in Q\}$$

represent the set of odometry data at timestep k for all robots in subset Q. For measurement information, let

$$Y_k = \{\mathbf{y}_{i,k}^{j,i} | i \in N, j \in N, d_k^{j,i} \leq r_{\mathsf{obs}}\}$$

represent the set of all relative measurements (made by all robots) at timestep k,

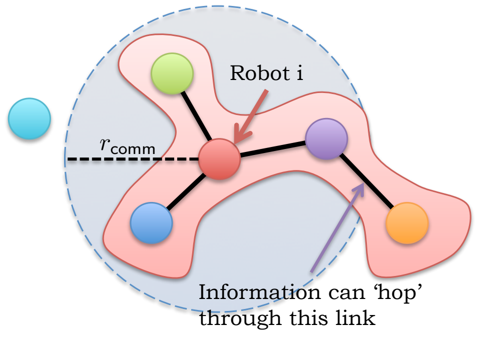
$$Y_{i,k} = \{\mathbf{y}_{i,k}^{j,i} | j \in N, d_k^{j,i} \leq r_{\mathsf{obs}}\}$$

represent the set of all relative measurements made by robot i at timestep k, and

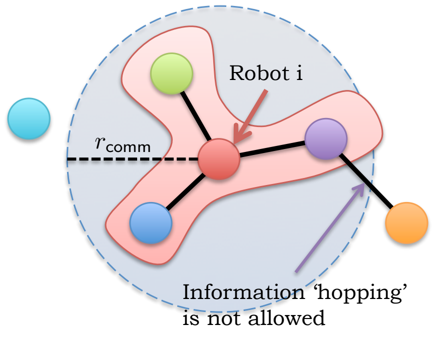
$$Y_{Q,k} = \{\mathbf{y}_{i,k}^{j,i} | i \in Q, j \in N\, d_k^{j,i} \leq r_{\mathsf{obs}}\}, Q \subseteq N$$

represent the set of relative measurements made by the robots in set Q at timestep k.

Finally, let $R_{i,k}$ represent the set of robots that can exchange information with robot i at time k. As illustrated in Fig. 2.1, we can define $R_{i,k}$ with respect to a robot in one of two ways. Let us consider the red robot in the figure as robot i. In the first case, as shown in Fig. 2.1a, we assume that $R_{i,k}$ is the set of robots with which robot i is connected. In other words, robot i can communicate with robots within communication range, r_{comm}, and with other robots outside of its communication range by the instantaneous relay of information through another robot. In the second case, as shown in Fig. 2.1b, we assume that robot i can only exchange information strictly with other robots within its communication range. Information relay is still possible, but additional timesteps are required for the information to 'hop' between robots. For the decentralized cooperative localization problem in this chapter, we will assume that only the first mode of information exchange (i.e., Fig. 2.1a) applies. That is, a robot can communicate with any robot with which it is connected. For subsequent chapters, when we examine the decentralized cooperative SLAM problem, we can assume either mode of communication. The reason why we impose the first mode of communication for decentralized cooperative localization is because this allows robots to not have to initially know the total number of robots operating in the team. We will explain this in greater detail in Section 2.4.1.



(a) Communication mode 1. $R_{i,k}$ is defined as the set of robots (in the red patch) connected to robot i.



(b) Communication mode 2. $R_{i,k}$ defined as the set of robots (in the red patch) within communication range, r_{comm}, of robot i.

Figure 2.1: Illustrated examples of the two ways in which $R_{i,k}$ can be defined. A black line between two robots indicate that they are within communication range, r_{comm}.

2.2.2 The Probabilistic Framework

In the presence of uncertainty (i.e., noise) in both state transition and measurements ((2.1) and (2.2)), the true state of the system cannot be found exactly, but can only be estimated using odometry and measurement data. In general, the centralized *belief* is represented by a probability density function, $p(\cdot)$, over all robot states, X_k:

$$\mathsf{bel}(X_k) := p\left(X_k | \mathsf{bel}(X_0), U_{1:k}, Y_{1:k}\right),$$

which is conditioned on the initial belief, $\mathsf{bel}(X_0)$, past odometry data, $U_{1:k}$, and past measurements $Y_{1:k}$. The notation $k_1 : k_2$ implies the time interval k_1 to k_2, inclusive. It is important to note that this centralized belief accounts for all the information from the initial timestep to the time of the state estimate (i.e., 0 to k). In our decentralized approach, we will attempt to have every robot produce an equivalent estimate as the one produced by the centralized estimator.

From a practical and computation point of view, it is helpful to apply the *Markov* *property*[3] [76, 77]

$$p\left(X_k|\mathsf{bel}(X_0), U_{1:k}, Y_{1:k}\right) = p\left(X_k|\mathsf{bel}(X_{k-1}), U_k, Y_k\right)$$

when performing state estimation. This property is typically assumed to apply in dealing with stochastic systems such as the one we have defined in (2.1) and (2.2). The assumption is that the belief of the current state of a system is independent of all past states, and that future estimates are only dependent on the current state estimate. Making use of this assumption reduces computational requirements by allowing state estimation to be performed recursively. For a centralized state estimator, this can be accomplished by using the *Bayes filter* [76]:

$$\begin{aligned}\mathsf{bel}(X_k) &= p\left(X_k|\mathsf{bel}(X_0), U_{1:k}, Y_{1:k}\right)\\ &= \eta\, p\left(Y_k|X_k\right)\int p\left(X_k|X_{k-1}, U_k\right) p\left(X_{k-1}|\mathsf{bel}(X_0), U_{1:k-1}, Y_{1:k-1}\right) dX_{k-1}\end{aligned}$$

where η is a normalizing constant to ensure that the resulting posterior probability density function, $\mathsf{bel}(X_k)$, preserves the axiom of total probability (i.e., integrates to one). Applying the Markov property when we have $\mathsf{bel}(X_k)$ allows us to replace previous estimates, $\mathsf{bel}(X_{0:k-1})$, odometry data, $U_{1:k-1}$, and measurements, $Y_{1:k-1}$, by the estimate $\mathsf{bel}(X_k)$. Since these information are no longer needed, they can be safely discarded and this helps to limit memory usage.

In a decentralized framework wherein robots are not always in contact with each other, the Markov property can only be applied once a robot obtains sufficient information regarding other robots through communication (i.e., a robot will rarely have the most up-to-date information from all other robots due to the communication constraint). Furthermore, each robot must ensure that other robots will no longer require any of the past information it possesses (because the information will be discarded after applying the Markov property). Hence, given that robots are only occasionally exchanging information with each other, our key problem here is to determine the necessary and sufficient conditions under which the Markov property can be applied in order to obtain an estimate that is equivalent to that obtained by a centralized state estimator. Accordingly, our objective is for each robot, $i \in N$, to estimate the state of all known robots (i.e., find $\mathsf{bel}(X_k)$) in a decentralized manner. We call this the *centralized-equivalent approach.*

[3]also referred to as the generalized causality principle

2.3 Information Flow in a Dynamic Network

In the case of sporadic communication and observations, it is essential to track the information available to each robot for making state estimates to ensure that:

1. a robot does not apply the Markov property without receiving all information required to calculate the centralized state estimate, and
2. a robot does not re-use information and cause *cyclic updates* to occur.
3. *out-of-sequence-measurements* are accounted for in the correct temporal order.

A cyclic update is an event where the same piece of information is used multiple times (double-counted) in calculating a state estimate. For instance, suppose a robot communicates its state estimate to another robot, which uses the information to update its own estimate, and communicates this updated estimate back to the first robot. If the first robot now uses the updated estimate to perform another update of its own estimate, then a cyclic update has occurred. A consequence of this is that the estimate will no longer be equivalent to the centralized estimate. Furthermore, there is a high chance that the estimate is now inconsistent[4] (i.e., the estimate is over-confident) [36].

In a dynamic network, the sequence in which information is communicated to a robot is often out of order, and this leads to the out-of-sequence measurements (OOSM) problem. Bar-Shalom [32], Bar-Shalom et al. [33] examined some possible remedies for treating OOSM in state estimation using a Kalman filter (also known as the negative timestep problem). It was shown that for a missed measurement, the only way to incorporate it to produce a centralized state estimate is to sequentially reprocess all following measurements. This is an important concept to note in this chapter.

A graph is a convenient tool for representing network topology. It not only allows us to represent the communication network between robots at a given time instance, but also over a time interval. This enables us to capture the dynamics of the network and keep track of the flow of information between robots. We will first examine the robot network from the perspective of an outside observer who has the ability to see all the interactions between robots. This benefit of the global perspective (although unrealistic) allows us to analyze how information flows through the network to all robots over time. We will then look at the network locally (i.e., from the perspective of a particular robot) and analyze when a robot is able to obtain the centralized-equivalent estimate (and when it becomes appropriate for a robot to apply the Markov property). Without loss of generality, we will assume for now that $r_{\mathsf{comm}} = r_{\mathsf{obs}}$, but this is only for explanation purposes. We

[4]Refer to [78] to learn more on estimator consistency.

will revisit this assumption in a later section to show that the two do not need to be equivalent.

2.3.1 The Global Perspective

Following our plan, we will first examine network dynamics from the perspective of an outside observer, who has the benefit of seeing the communication linkages between all robots at any timestep. Let $\mathcal{G}_{k_1:k_2}$ be a directed graph that shows the communication links established between robots from timestep k_1 to k_2. This graph can be used to show the flow and distribution of information and we will refer to $\mathcal{G}_{k_1:k_2}$ as the *global information flow graph*. The number of nodes in $\mathcal{G}_{k_1:k_2}$ is the product of the number of robots and the number of timesteps represented by the graph. These nodes, v, can be systematically numbered using the time index, k, and the robot index, i, such that

$$v(i,k) = (k-k_1)|N| + i, \quad (k_1 \leq k \leq k_2)\,(1 \leq i \leq |N|)\,.$$

An example of an information flow graph is depicted in Fig. 2.2. In relation to the system model, an arc connecting two robots at a timestep represents a measurement between the two robots, and also a communication window that allows the exchange of information between both robots (remember that we have temporarily assumed that $r_{\mathsf{comm}} = r_{\mathsf{obs}}$). A horizontal arc represents information carried forward through time by a robot. During state transitions, new odometry data become available, and can be passed to other robots according to the graph. In general, the presence of an arc connecting two nodes implies that all information at the originating node is also available at the destination node. Furthermore, odometry and measurement information that are labelled on the arcs making up the path in between are also available at the destination node.

As a robot traverses a workspace and occasionally observes and communicates with other robots, it accumulates information regarding the entire team. The specific data in its possession will depend on the evolving topology of the information flow graph. To keep track of what a robot knows, let the *knowledge set*, $S_{i,k}$, consist of all odometry and measurement data, as well as the previous state estimates known to robot i at time k. We will assume at the initial time that $S_{i,0} = \{\mathsf{bel}(\mathbf{x}_{i,0})\}$. It must be noted here that all the initial beliefs are expressed in the same reference frame (or can be initially determined). As such, we do not deal with the initial correspondence problem of determining the relative transformations between the initial local reference frames on each robot. The practical implication of this is that the robots either have to start in close proximity, or the initial pose needs to be predetermined before deployment.

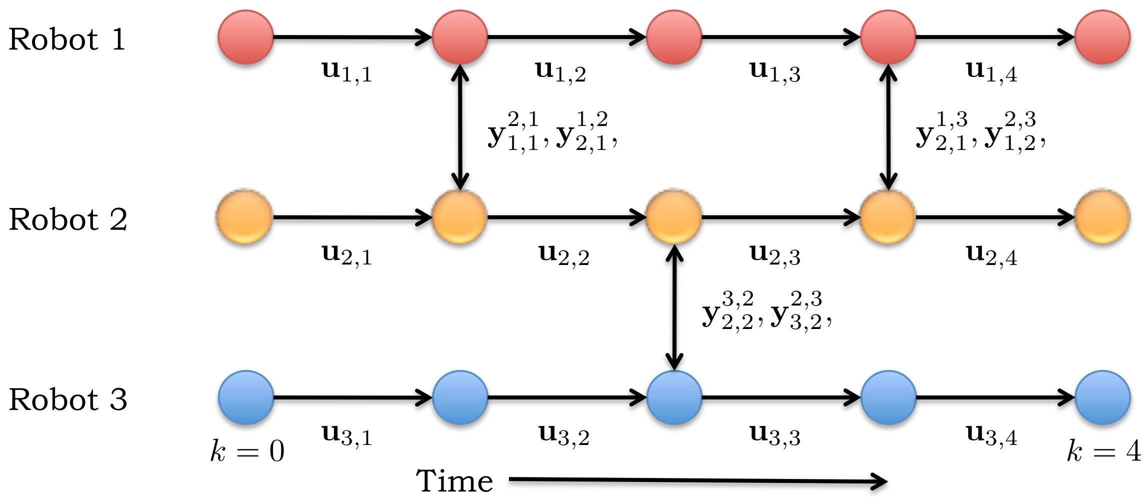


Figure 2.2: An example global information flow graph, $\mathcal{G}_{k_1:k_2}$, where the edges indicate how information from one robot reaches another robot between timesteps k_1 and k_2. The presence of an arc connecting two nodes implies that all information at the originating node is also available at the destination node. Odometry and measurement information that are labeled on the arcs making up the path in between are also available at the destination node.

At each timestep, the knowledge set expands with the addition of new odometry data as well as measurement data if another robot is observed. Recall that $R_{i,k}$ represents the set of robots connected to robot i at time k. Let $S_{i,k}^{-}$ represent the knowledge set after state transition and observations, but before communication is established with any other robot:

$$S_{i,k}^{-} = S_{i,k-1} \cup \{\mathbf{u}_{i,k}, Y_{i,k}\} \tag{2.3}$$

When communication occurs between robots i and j, they will make their knowledge sets available to each other, and the knowledge set of both robots will become identical. When a robot has communicated with all robots within its communication range, its knowledge set will become the union of its own knowledge set, and the knowledge sets of all other robots within communication range as follows, where the $\underset{j \in R_{i,k}}{\cup}$ operator represents the union with all the sets indexed by j:

$$S_{i,k} = S_{i,k}^{-} \underset{j \in R_{i,k}}{\cup} S_{j,k}^{-} \tag{2.4}$$

The above equations model information flow within the robot network at every timestep. With the progression of time, the knowledge set for each robot will continue to expand, causing the information storage requirement to increase as well. If the Markov property is not exploited in an estimator, the amount of data in each knowledge set will increase over time without bound. In most centralized recursive state estimators, we make use of the Markov property to reduce memory storage requirements. In our decentralized state estimation problem, this must be done with extreme care to ensure that all robots can also make the same centralized-equivalent estimate. For this purpose, a *checkpoint* is defined as follows:

Definition 4: *A* checkpoint, $C(k_c, k_e)$, *is an event that occurs at the checkpoint time, k_c, that first comes into existence at k_e, in which the set of knowledge for each robot i contains for all j:*

1. *the previous state estimate of robot j at some timestep, $k_{s,j} \leq k_c$,*
2. *all the odometry and measurement data of robot j from timestep $k_{s,j}$ to k_c.*

Equivalently, a checkpoint occurs at timestep k_c when $S_{i,k_e} \supseteq \{S_{j,k_c} | j \in N\}, (\forall i \in N)$.

Using Fig. 2.2 as an example, and assuming that each robot begins with $S_{i,0} = \{\mathsf{bel}(\mathbf{x}_{i,0})\}$, one of the checkpoints that can be found in this figure is $C(1, 3)$. It can be verified that at $k = 3$ (i.e., k_e), each robot has the previous state estimate of all robots (at $k = 0$). Also, each robot has the odometry and measurement data of all robots up to $k = 1$ (i.e., k_c).

By its definition, the existence of a checkpoint is a necessary and sufficient condition for obtaining the centralized-equivalent estimate. Note that there exist other sufficient conditions for obtaining the centralized-equivalent estimate. For instance, having only two robots in the team that follow (2.3) and (2.4) for knowledge set exchange is a sufficient condition. A fully connected network also allows robots to obtain the centralized-equivalent estimate at all times. Furthermore, having a fixed communication pattern between robots may also be a sufficient condition (although this is not investigated explicitly). Our notion of a checkpoint encompasses all the scenarios above. That is, a checkpoint exists for all the conditions listed above. More importantly, the existence of a checkpoint is a necessary and sufficient condition for obtaining the centralized-equivalent estimate for any arbitrary number of robots, when the network is never fully connected, and when communication does not occur in any predictable pattern.

The existence of a checkpoint allows for the application of the Markov property (without concern of another robot not having adequate information to perform the same task). In applying the Markov property, old state estimates, odometry, and measurement data up to k_c, will be replaced by a new centralized-equivalent estimate at k_c.

To make use of a checkpoint, it is first necessary to detect its existence. According to the definition of a checkpoint, we need to show that the knowledge set of each robot at timestep k_e contains the state estimate for all robots at a timestep $k_{s,j}$, as well as the odometry and measurement data for all robots from timestep $k_{s,j}$ to k_c. Note that from (2.3) and (2.4), a property of knowledge sets is that

$$S_{i,k_1} \subseteq S_{i,k_2} \quad (k_1 \leq k_2),$$

provided that the Markov property is not applied between k_1 and k_2. This property guarantees that all information is retained as the knowledge set of a robot accumulates information over time. By this argument, the knowledge set of a robot must always contain its previous state estimate, all its previous odometry data, as well as measurements. That is, if the Markov property is not applied,

$$\begin{aligned} \mathsf{bel}(\mathbf{x}_{i,k_s}) \in S_{i,k_s} &\quad \Rightarrow \quad \mathsf{bel}(\mathbf{x}_{i,k_s}) \in S_{i,k}, (k \geq k_s), \\ \mathbf{u}_{i,k} \in S_{i,k} &\quad \Rightarrow \quad \mathbf{u}_{i,k} \in S_{i,k_c}, (k_c \geq k), \\ \mathbf{y}_{i,k} \in S_{i,k} &\quad \Rightarrow \quad \mathbf{y}_{i,k} \in S_{i,k_c}, (k_c \geq k). \end{aligned}$$

The following is a theorem for showing checkpoint existence given an information flow graph.

Theorem 1.1: *$C(k_c, k_e)$ exists if and only if a path exists from $v(i, k_c)$ to $v(j, k_e)$ on $\mathcal{G}_{k_c,k_e}$, $(\forall i, j)$.*

Proof. We will first assume that $C(k_c, k_e)$ exists. This implies that all odometry, measurements and state estimates from all robots at k_c are in the knowledge sets of all robots at k_e. This is only possible if there exists an information flow path from $v(i, k_c)$ to $v(j, k_e)$, $(\forall i, j)$.

Now we will assume that a path exists from $v(i, k_c)$ to $v(j, k_e)$, $(\forall i, j)$. By the knowledge set update rules, all odometry, measurements and state estimates from all robots at k_c will become incorporated into the knowledge sets of all robots at k_e. Therefore, $C(k_c, k_e)$ exists.

□

The necessary and sufficient conditions of Theorem 1.1 provide a method for verifying checkpoint existence (an indication of when the Markov property is applicable) given an information flow graph. However, there needs to be a more practical verification method for checkpoint existence for implementation purposes (that does not first require the construction of a graph). Before showing this, we need to introduce a lemma. Referring to Fig. 2.3, the lemma states that in the interval between the occurrence and existence time of one checkpoint (i.e., k_{c_1} and k_{e_1}), another checkpoint cannot come into existence. Hence, information in knowledge sets within this interval cannot be replaced by state estimates since we know (for now) that the Markov property can only be applied at a checkpoint.

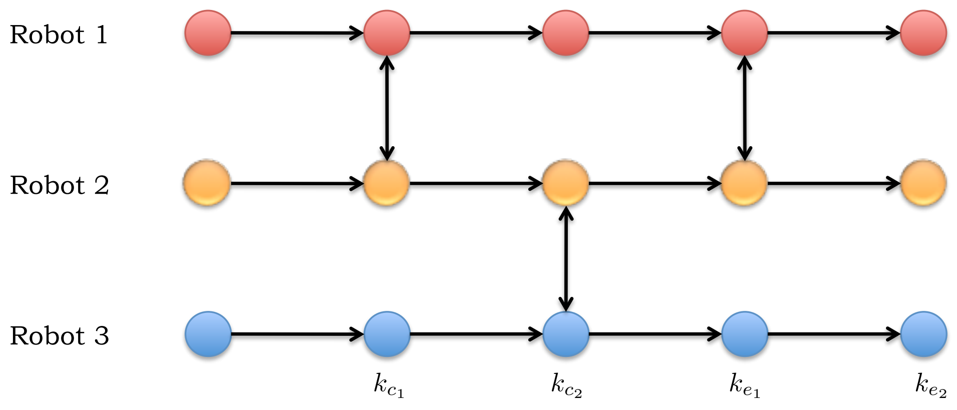


Figure 2.3: The existence times for sequential checkpoints; checkpoint $C(k_{c_2}, k_{e_2})$ cannot come into existence before an earlier occurring checkpoint $C(k_{c_1}, k_{e_1})$ comes into existence. Hence, k_{e_1} must be less than k_{e_2} since k_{c_1} is less than k_{c_2}.

Lemma 1.1: *Suppose $C(k_{c_1}, k_{e_1})$ and $C(k_{c_2}, k_{e_2})$ exist, and $k_{e_1} \neq k_{e_2}$. Then $k_{c_1} < k_{c_2}$ if and only if $k_{e_1} < k_{e_2}$.*

Proof. We will use the information flow graph as an aid in this proof. Furthermore, the notation $v_1 \xrightarrow{\text{path}} v_2$ will be used to denote that a path exists from node v_1 to node v_2. We will show that there is a violation in the existence of a checkpoint if the necessary and sufficient conditions are not followed. More formally, first let us assume that $k_{c_1} < k_{c_2}$.

$$\begin{aligned} & k_{c_1} < k_{c_2} \\ \Rightarrow\ & v(i, k_{c_1}) \xrightarrow{\text{path}} v(i, k_{c_2}) \xrightarrow{\text{path}} v(j, k_{e_2}), (\forall i, j) \\ \Rightarrow\ & v(i, k_{c_1}) \xrightarrow{\text{path}} v(j, k), (\forall i, j)(k_{e_1} \leq k < k_{e_2}) \\ \Rightarrow\ & k_{e_1} < k_{e_2} \end{aligned}$$

To explain in greater detail, we know that a robot always carries its knowledge forward in time (from k_{c_1} to k_{c_2}), and that $C(k_{c_2}, k_{e_2})$ exists. Therefore, there must exist a path for information flow between all robots from k_{c_1} to k_{e_2}. Next, we also know that $C(k_{c_1}, k_{e_1})$

exists, and so there must exist a path for information flow between all robots from k_{c_1} to k_{e_1}. Since, by definition of a checkpoint, k_{e_1} is the earliest existence time, this implies that k_{e_1} must be less than k_{e_2}. This completes the first half of the proof. Now let us assume that $k_{e_1} < k_{e_2}$.

$$\begin{aligned}
& k_{e_1} < k_{e_2} \\
\Rightarrow\ & v(i,k_{c_1}) \xrightarrow{\text{path}} v(j,k_{e_1}) \xrightarrow{\text{path}} v(j,k_{e_2}), (\forall i,j) \\
\Rightarrow\ & v(i,k) \xrightarrow{\text{path}} v(j,k_{e_2}), (\forall i,j)(k_{c_1} < k \leq k_{c_2}) \\
\Rightarrow\ & k_{c_1} < k_{c_2}
\end{aligned}$$

Given that $C(k_{c_1}, k_{e_1})$ exists, and with the assumption that $k_{e_1} < k_{e_2}$, there must exist a path for information flow between all robots from k_{c_1} to k_{e_2}. Knowing that $C(k_{c_2}, k_{e_2})$ exists, there must exist a path for information flow between all robots from k to k_{e_2}, where k can be less than or equal to k_{c_2}. However, k must be greater than k_{c_1} or else the earliest existence time of the checkpoint will no longer be k_{e_2}. Therefore $k_{c_1} < k_{c_2}$.

□

Note that in the above lemma statement, we specified different existence times (k_{e_1} and k_{e_2}) for the two checkpoints. If these times were the same, it would imply that all robots can simultaneously obtain the centralized-equivalent for two occurrence times. In this situation, the checkpoint that occurs earlier becomes irrelevant because being able to calculate the centralized-equivalent estimate at a later timestep guarantees that we can calculate, or have already calculated the centralized-equivalent estimate for the earlier occurrence time.

Using the above lemma, we now present a theorem that gives a more practical method for verifying the existence of a checkpoint for implementation purposes. We will show that it is possible to know when a checkpoint exists by looking for a subset of odometry information in the knowledge set of each robot.

Theorem 1.2: *$C(k_c, k_e)$ exists if and only if the knowledge set of each robot at k_e contains $\{(\mathbf{u}_{j,k_c}$ or $\mathsf{bel}(X_{j,k_c}))\,|j \in N\}, (\forall i \in N)$.*

Proof. Expressed mathematically the theorem statement is:

$$S_{i,k_e} \supseteq \{S_{j,k_c}|j \in N\}, (\forall i \in N) \Leftrightarrow S_{i,k_e} \supseteq \{(\mathbf{u}_{j,k_c} \text{ or } \mathsf{bel}(X_{j,k_c}))\,|j \in N\}, (\forall i \in N)$$

We will approach this proof by first assuming that $C(k_c, k_e)$ exists:

$$\begin{aligned}
S_{i,k_e} &\supseteq \{S_{j,k_c} | j \in N\}, (\forall i \in N) \\
\Rightarrow S_{i,k_e} &\supseteq \begin{cases} \left\{\mathsf{bel}(X_{j,k_{s,j}}), Y_{j,k_{s,j}+1:k_c}, \mathbf{u}_{j,k_{s,j}+1:k_c} | j \in N\right\}, (\forall i \in N) \text{ if } (k_{s,j} < k_c) \\ \left\{\mathsf{bel}(X_{j,k_{s,j}}) | j \in N\right\}, (\forall i \in N) \text{ if } (k_{s,j} = k_c) \end{cases} \\
\Rightarrow S_{i,k_e} &\supseteq \{(\mathbf{u}_{j,k_c} \text{ or } \mathsf{bel}(X_{j,k_c})) | j \in N\}, (\forall i \in N)
\end{aligned}$$

From $S_{i,k_e} \supseteq S_{j,k_c}$, we expand $S_{j,k_c}(\forall j)$ to show the information that can be found in the knowledge sets depending on whether $k_{s,j}$ (the time of the latest belief for robot j) is less than k_c or equal to k_c. Then we show that either $\mathbf{u}_{j,k_c}$ or $\mathsf{bel}(X_{j,k_c})$ for all robots j will always be available.

Now assuming that the knowledge set of each robot at k_e contains $\mathbf{u}_{j,k_c}$ or $\mathsf{bel}(\mathbf{x}_{j,k_c})$:

$$\begin{aligned}
S_{i,k_e} &\supseteq \{(\mathbf{u}_{j,k_c} \text{ or } \mathsf{bel}(X_{j,k_c})) | j \in N\}, (\forall i \in N) \\
\Rightarrow S_{i,k_e} &\supseteq \{S_{j,k} | j \in N\}, (\forall i \in N)(k_c \leq k \leq k_e), \\
\Rightarrow S_{i,k_e} &\supseteq \{S_{j,k_c} | j \in N\}, (\forall i \in N).
\end{aligned}$$

According to the knowledge set update rules, all the information from a robot is communicated to another robot that is within communication range. This implies that having one piece of information at a particular timestep from a robot is a sufficient condition for having all the information at the same timestep from the same robot. It follows then in line 2 above that since odometry data and the belief at k_c from all robots j are available, then $S_{j,k}$ must also be available for all robots j, where $k_c \leq k \leq k_e$. Furthermore, we know that the knowledge set of a robot will always contain its past knowledge set, provided the Markov property has not been applied. We are certain of this because Lemma 1.1 indicates that another checkpoint cannot come into existence between k_c and k_e. Therefore the Markov property can never be applied within this timeframe.

□

We have just shown how every robot in a robot team can obtain the centralized-equivalent estimate and apply the Markov property while operating in a network that is never fully connected.

Up to this point, we have defined how information is exchanged between robots when they are within communication range. We have also defined the condition that allows us to determine when all robots have adequate information in their knowledge sets to calculate the centralized-equivalent estimate. This can be done graphically using the information flow graph, or by searching the knowledge set for a subset of odometry information. However, recall that we have been examining the network from the perspective of an outside observer who is able to see all the communication linkages established between all robots, at every timestep. In terms of implementation, having such an overseer would defeat the purpose of utilizing a decentralized approach because it would be analogous to having a centralized agent commanding each robot what to do. In the next part, we will break away from the luxury of having an overseer, and look at network topology from the perspective of a single robot.

2.3.2 The Local Perspective

Previously, we have shown a global information flow graph in Fig. 2.2, which represents the interactions of all robots as viewed by an outside observer. In relation to this, the graph topology from the point of view of an individual robot will differ as illustrated in Fig. 2.4.

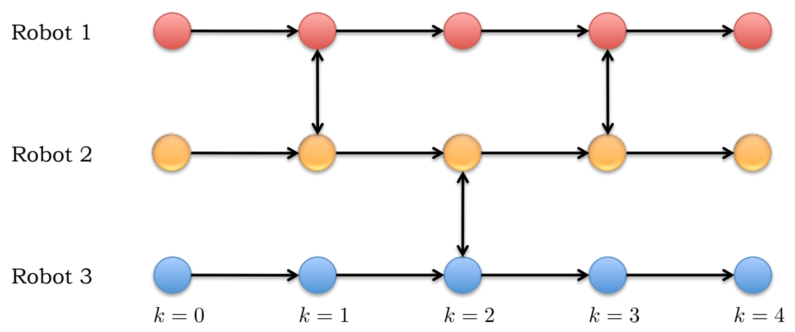
From a robot's point of view, it may not be aware of communication events between other robots until a much later time. This leads to the question of whether or not it is necessary for a robot to know the complete knowledge set of all other robots (i.e., determine that a checkpoint exists) before it can calculate the centralized-equivalent estimate and safely apply the Markov property without affecting other robots' abilities to do the same. It turns out that we do not. To show this, we present the definition of a *partial checkpoint*, and a theorem for its existence.

Definition 5: *A* partial checkpoint, *$C_p(k_{c,i}, k_{e,i})$, is an event that occurs for robot i at time $k_{c,i}$, that first comes into existence at $k_{e,i}$, in which the set of knowledge for robot i contains for all j:*

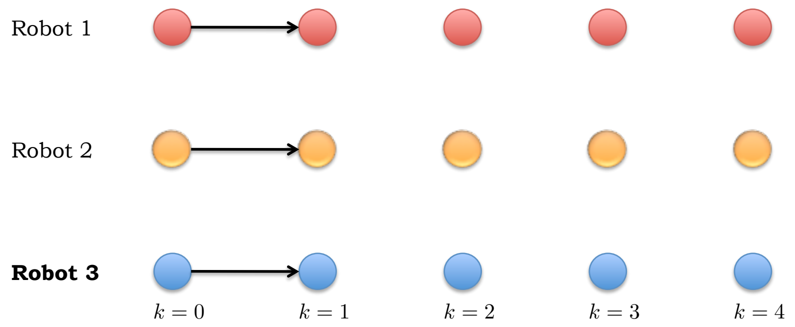
1. *the previous state estimate of robot j at some timestep, $k_{s,j} \leq k_{c,i}$,*
2. *all the odometry and measurement data of robot j from timestep $k_{s,j}$ to $k_{c,i}$.*

Equivalently written in symbols, a partial checkpoint for robot i occurs at timestep $k_{c,i}$ when $S_{i,k_e} \supseteq \{S_{j,k_c} | j \in N\}$.

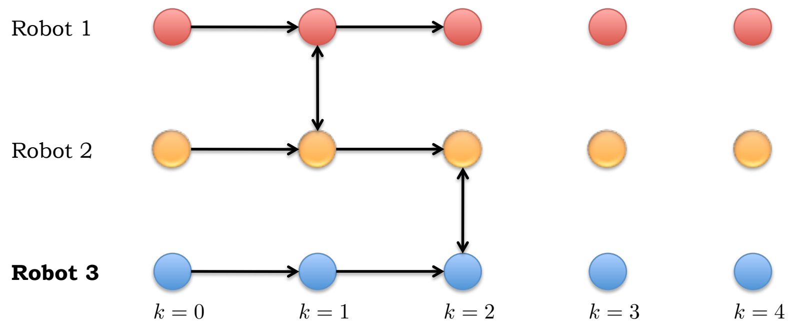
Notice that the difference between a partial checkpoint and a checkpoint is that we are now looking at the knowledge set of only one robot. As with checkpoints, we have a number of theoretical statements for partial checkpoints' existence. These statements (and proofs) are largely similar to Theorem 1.1, Lemma 1.1 and Theorem 1.2.



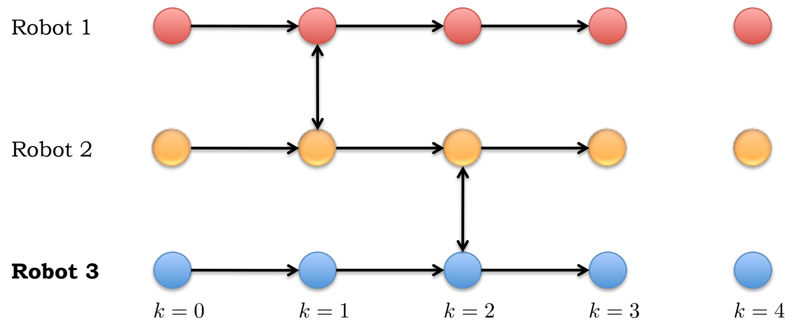
(a) The global information flow graph



(b) At k=1, robot 3 is unaware of the interaction between robot 1 and robot 2.



(c) At k=2, robot 3 is now aware of the previous communication between robot 1 and robot 2 from timestep 1.



(d) At k=3, robot 3 is unaware of the interaction between robot 1 and robot 2.

Figure 2.4: Information flow in the perspective of a robot (Robot 3).

Theorem 2.1: *For robot i, $C_p(k_{c,i}, k_{e,i})$ exists if and only if a path exists from $v(j, k_{c,i})$ to $v(i, k_{e,i})$ on $\mathcal{G}_{k_c,k_e}$, $(\forall j)$.*

Proof. We will first assume that $C_p(k_{c,i}, k_{e,i})$ exists. This implies that all odometry, measurements and state estimates from all robots at $k_{c,i}$ are in the knowledge sets of robots i at $k_{e,i}$. Clearly, this is only possible if there exists an information flow path between $v(i, k_c)$ to $v(j, k_e)$, $(\forall j)$.

Next we assume that a path exists between $v(i, k_c)$ to $v(j, k_e)$, $(\forall j)$. By the knowledge set update rules, all odometry, measurements and state estimates from all robots at $k_{c,i}$ will become incorporated into the knowledge sets of robot i at $k_{e,i}$. Therefore, $C_p(k_{c,i}, k_{e,i})$ exists.

□

Lemma 2.1: *Suppose $C_p(k_{c_1,i}, k_{e_1,i})$ and $C_p(k_{c_2,i}, k_{e_2,i})$ exist, and $k_{e_1,i} \neq k_{e_2,i}$. Then $k_{c_1,i} < k_{c_2,i}$ if and only if $k_{e_1,i} < k_{e_2,i}$.*

Proof. First let us assume that $k_{c_1,i} < k_{c_2,i}$.

$$\begin{aligned}
& k_{c_1,i} < k_{c_2,i} \\
\Rightarrow\ & v(i, k_{c_1,i}) \xrightarrow{\text{path}} v(i, k_{c_2,i}) \xrightarrow{\text{path}} v(j, k_{e_2,i}), (\forall j) \\
\Rightarrow\ & v(i, k_{c_1,i}) \xrightarrow{\text{path}} v(j, k), (\forall j)(k_{e_1,i} \leq k < k_{e_2,i}) \\
\Rightarrow\ & k_{e_1,i} < k_{e_2,i}
\end{aligned}$$

Now let us assume that $k_{e_1,i} < k_{e_2,i}$.

$$\begin{aligned}
& k_{e_1,i} < k_{e_2,i} \\
\Rightarrow\ & v(i, k_{c_1,i}) \xrightarrow{\text{path}} v(j, k_{e_1,i}) \xrightarrow{\text{path}} v(j, k_{e_2,i}), (\forall j) \\
\Rightarrow\ & v(i, k) \xrightarrow{\text{path}} v(j, k_{e_2,i}), (\forall j)(k_{c_1} < k \leq k_{c_2,i}) \\
\Rightarrow\ & k_{c_1,i} < k_{c_2,i}
\end{aligned}$$

Therefore, $k_{c_1,i} < k_{c_2,i}$ if and only if $k_{e_1,i} < k_{e_2,i}$. For further justification of this proof, one can refer back to the proof of Lemma 1.1, which provides a similar argument structure.

□

Theorem 2.2: *$C_p(k_{c,i}, k_{e,i})$ exists if and only if the knowledge set of robot i at k_e contains $\left\{ \left(\mathbf{u}_{j,k_{c,i}} \text{ or } \mathsf{bel}(X_{j,k_{c,i}}) \right) | j \in N \right\}$.*

Proof. Expressed mathematically, the theorem states that:

$$S_{i,k_{e,i}} \supseteq \left\{S_{j,k_{c,i}} | j \in N\right\} \Leftrightarrow S_{i,k_{e,i}} \supseteq \left\{\left(\mathbf{u}_{j,k_{c,i}} \text{ or } \mathsf{bel}(X_{j,k_{c,i}})\right) | j \in N\right\}$$

Assume that $C_p(k_{c,i}, k_{e,i})$ exists:

$$\begin{aligned}
S_{i,k_{e,i}} &\supseteq S_{j,k_{c,i}}, (\forall j) \\
\Rightarrow S_{i,k_{e,i}} &\supseteq \begin{cases} \left\{\mathsf{bel}(X_{j,k_{s,j}}), Y_{j,k_{s,j}+1:k_{c,i}} \mathbf{u}_{j,k_{s,j}+1:k_{c,i}} | j \in N\right\}, \text{ if } (k_{s,j} < k_{c,i}) \\ \left\{\mathsf{bel}(X_{j,k_{s,j}}) | j \in N\right\}, \text{ if } (k_{s,j} = k_{c,i}) \end{cases} \\
\Rightarrow S_{i,k_{e,i}} &\supseteq \left\{\left(\mathbf{u}_{j,k_{c,i}} \text{ or } \mathsf{bel}(X_{j,k_{c,i}})\right) | j \in N\right\}
\end{aligned}$$

Now assuming that the knowledge set of each robot at $k_{e,i}$ contains $\mathbf{u}_{j,k_{c,i}}$ or $\mathsf{bel}(X_{j,k_{c,i}})$:

$$\begin{aligned}
S_{i,k_{e,i}} &\supseteq \left\{\left(\mathbf{u}_{j,k_{c,i}} \text{ or } \mathsf{bel}(X_{j,k_{c,i}})\right) | j \in N\right\} \\
\Rightarrow S_{i,k_{e,i}} &\supseteq \{S_{j,k} | j \in N\}), (k_{c,i} \leq k \leq k_{e,i}) \\
\Rightarrow S_{i,k_{e,i}} &\supseteq \left\{S_{j,k_{c,i}} | j \in N\right\}
\end{aligned}$$

Therefore, $S_{i,k_{e,i}} \supseteq \left\{S_{j,k_{c,i}} | j \in N\right\} \Leftrightarrow S_{i,k_{e,i}} \supseteq \left\{\left(\mathbf{u}_{j,k_{c,i}} \text{ or } \mathsf{bel}(X_{j,k_{c,i}})\right) | j \in N\right\}$ □

Similarly to Theorem 1.2, Theorem 2.2 is important as it provides a practical method for detecting partial checkpoint existence. Note that a partial checkpoint can come into existence at different times for different robots depending on the evolving topology of the robot network. The question that remains to be answered is whether a robot's decision to obtain a centralized-equivalent estimate and apply the Markov property (based on local information) affects other robots' abilities to do the same. To answer this, we now present a lemma, and two important theorems that relate partial checkpoints to checkpoints, to show how the occurrences of these events are affected when the Markov property is applied:

Lemma 3.1: *$C(k_c, k_e)$ exists if and only if $C_p(k_{c,i}, k_{e,i})$ exists, $(\forall i \in N)$, with $(k_{c,i} = k_c)$ and $(k_{e,i} \leq k_e)$.*

Proof. The underlying message in this lemma is that when all robots individually detect a partial checkpoint that occurs at the same timestep, then a checkpoint exists. The approach to this proof is to use the definitions of a checkpoint and a partial checkpoint. We will show that when the knowledge set of each robot satisfies the condition for partial checkpoint existence for timestep k_c, then we also satisfy the condition for checkpoint existence for timestep k_c.

First, assume $C(k_c, k_e)$ exists:

$$\begin{aligned} & C(k_c, k_e) \text{ exists} \\ \Rightarrow\ & S_{i,k_e} \supseteq \{S_{j,k_c} | j \in N\}, (\forall i \in N) \\ \Rightarrow\ & (\exists k_{e,i} \leq k_e) \text{ such that}, S_{i,k_{e,i}} \supseteq \{S_{j,k_c} | j\ inN\}, (\forall i \in N) \\ \Rightarrow\ & (\exists k_{e,i} \leq k_e) \text{ such that}, S_{i,k_{e,i}} \supseteq \left\{S_{j,k_{c,i}} | j \in N\right\}, (\forall i \in N)(k_{c,i} = k_c) \\ \Rightarrow\ & C_p(k_{c,i}, k_{e,i}) \text{ exists}, (\forall i)(k_{c,i} = k_c)(k_{e,i} \leq k_e) \end{aligned}$$

We rewrite the expression of a checkpoint existence using the knowledge set notation. Note that $S_{j,k_c}, (\forall j)$ is available to all robots at earliest timestep k_e, but it is possible for individual robots to obtain this at an earlier time $k_{e,i}$. We then substitute the timestep variable k_c by $k_{c,i}$. Finally, we use the definition of a partial checkpoint to conclude that a partial checkpoint (that occurs at the same time) exists for all robots. Now we will assume $C_p(k_{c,i}, k_{e,i})$ exists, $(\forall i)$, with $(k_{c,i} = k_c)$ and $(k_{e,i} \leq k_e)$:

$$\begin{aligned} & C_p(k_{c,i}, k_{e,i}) \text{ exists}, (\forall i)(k_{c,i} = k_c)(k_{e,i} \leq k_e) \\ \Rightarrow\ & S_{i,k_{e,i}} \supseteq \left\{S_{j,k_{c,i}} | j \in N\right\}, (\forall i \in N)(k_{c,i} = k_c)(k_{e,i} \leq k_e) \\ \Rightarrow\ & S_{i,k_e} \supseteq \{S_{j,k_c} | j \in N\}, (\forall i \in N) \\ \Rightarrow\ & C(k_c, k_e) \text{ exists} \end{aligned}$$

We first use the definition of a partial checkpoint to express its existence using knowledge sets. Next, we note that S_{i,k_e} is a superset of $S_{i,k_{e,i}}$ since $k_{e,i} \leq k_e$. We then substitute the timestep variable $k_{c,i}$ with k_c, and use the definition of a checkpoint to conclude its existence.

□

Using this Lemma 3.1, we can be sure that when partial checkpoints (that occur at the same time $k_{c,i}$ for all robots) exist, then a checkpoint also exists (with $k_c = k_{c,i}, (\forall i)$). This will be used in the next theorem, which is the most important theorem of this chapter. It tells us that a robot's decision to apply the Markov property (based only on local information) does not affect other robots' abilities to do the same.

Theorem 3.1: *Suppose the checkpoint $C(k_c, k_e)$ exists, and robot m applies the Markov property when $C_p(k_c, k_{e,m})$ exists (i.e., at $k_{e,m}$). Then $C_p(k_c, k_{e,i})$ continues to exist,* $(\forall i)$.

Proof. We approach this proof by examining the knowledge set of each robot and the changes caused by applying the Markov property. We then verify that partial checkpoints continue to exist for all robots.

When a checkpoint exists, and before the Markov property is applied by robot m, the knowledge sets of all robots i contain the belief at some previous time, $k_{s,j}$, for all robots j, as well as odometry and measurements up to k_c:

$$\begin{aligned}
& C(k_c, k_e) \text{ exists} \\
\Rightarrow\ & C_p(k_c, k_{e,i}) \text{ exists}, (\forall i) \\
\Rightarrow\ & S_{i,k_{e,i}} \supseteq \{S_{j,k_c} | j \in N\}, (\forall i \in N) \\
\Rightarrow\ & S_{i,k_{e,i}} \supseteq \begin{cases} \left\{\mathsf{bel}(\mathbf{x}_{j,k_{s,j}}), \mathbf{u}_{j,k_{s,j}+1:k_c}, Y_{j,k_{s,j}+1:k_c} | j \in N\right\}, (\forall i \in N) \text{ if } (k_{s,j} < k_c) \\ \left\{\mathsf{bel}(\mathbf{x}_{j,k_{s,j}}) | j \in N\right\}, (\forall i \in N) \text{ if } (k_{s,j} = k_c) \end{cases}
\end{aligned}$$

It is of interest to know how the knowledge sets of robots that are receiving information from robot m will change once robot m applies the Markov property. Again using $\xrightarrow{\text{path}}$ to denote the existence of a path on the information flow graph, let

$$Q = \left\{ \text{all robots } i \middle| v(m, k_{e,m}) \xrightarrow{\text{path}} v(i, k_{e,i}) \right\}.$$

and $\overline{Q} = N - Q$. Now if we suppose that robot m has applied the Markov property at $k_{e,m}$, then $S_{m,k_{e,m}} \supseteq \{\mathsf{bel}(\mathbf{x}_j, k_c) | j \in N\}$. Furthermore, all robots in Q will also obtain this belief in their knowledge set:

$$\begin{aligned}
& S_{i,k_{e,i}} \supseteq \begin{cases} \{\mathsf{bel}(\mathbf{x}_{j,k_c}) | j \in Q\} \\ \left\{\mathsf{bel}(\mathbf{x}_{j,k_{s,j}}), \mathbf{u}_{j,k_{s,j}+1:k_c}, Y_{j,k_{s,j}+1:k_c} | j \in \overline{Q}\right\}, \text{ if } (k_{s,j} < k_c) \\ \left\{\mathsf{bel}(\mathbf{x}_{j,k_{s,j}}) | j \in \overline{Q}\right\}, \text{ if } (k_{s,j} = k_c) \end{cases} \qquad \forall i \in N \\
\Rightarrow\ & S_{i,k_{e,i}} \supseteq \{S_{j,k_c} | j \in N\}, (\forall i \in N) \\
\Rightarrow\ & C_p(k_c, k_{e,i}) \text{ exists}, (\forall i \in N)
\end{aligned}$$

For robots in $\overline{Q}$, their knowledge sets will remain unaffected and contain the same information as before the Markov property was applied by robot m. Regardless, each of the three cases for $S_{i,k_{e,i}}$ shown above allows us to conclude that by definition, a partial checkpoint exists for all robots i. Fig.2.5 is an illustration of this theorem. □

The implication of Theorem 3.1 is significant because we are now certain that a robot's decision to invoke the Markov property as soon as its partial checkpoint exists (and based on its own knowledge) will have no effect on the other robots' abilities to obtain a partial checkpoint (that occurs for the same timestep, k_c). This implies that all robots will only

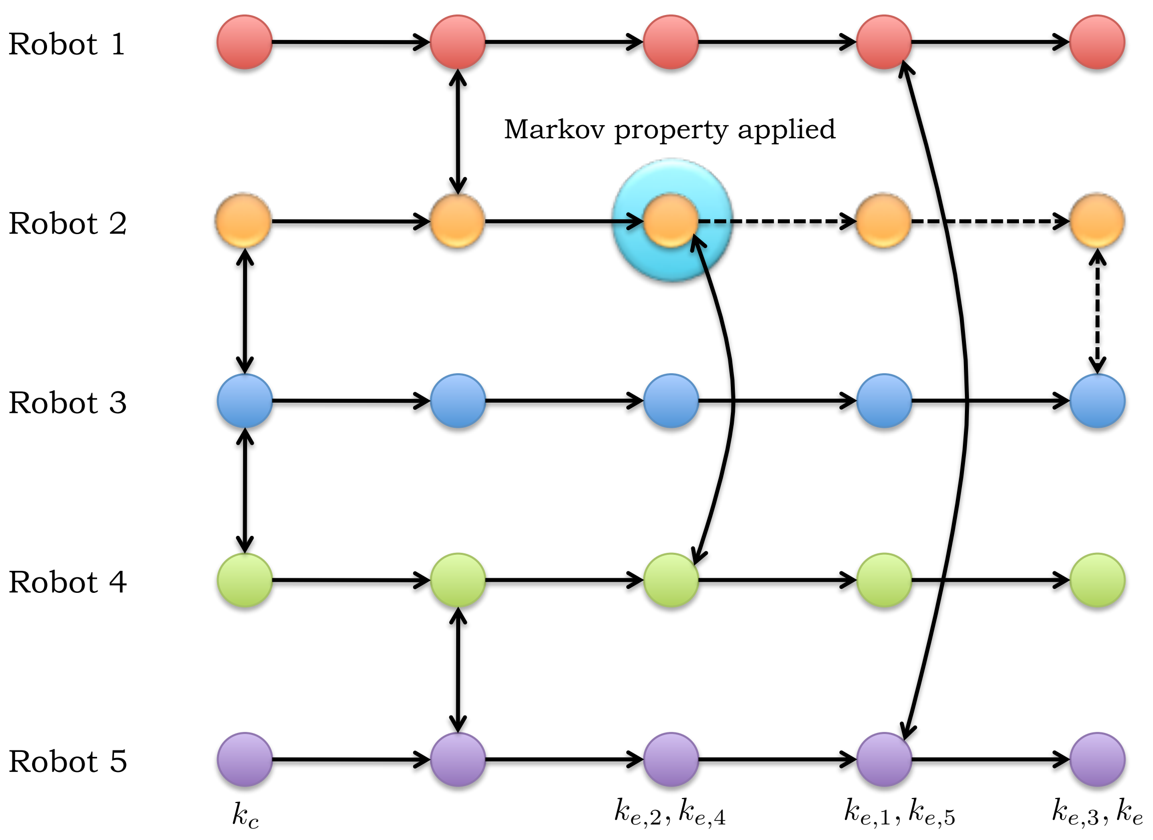


Figure 2.5: An illustration of Theorem 3.1. $C(k_c, k_e)$ exists, and robot 2 applies the Markov property at k_{e_2} when $C_p(k_c, k_{e_2})$ exists. The set of robots in Q are robots 2 and 3 (i.e., the nodes connected by the dashed edges) and they will obtain $\mathsf{bel}(X_{k_c})$, which is calculated by robot 2 at $k_{e,2}$. Note that robot 4 is not in Q because robot 2 obtains the centralized-equivalent estimate and applies the Markov property after communication with robot 4.

need to consider their local knowledge when applying the Markov property.

A point worth noting is that some robots do not actually have to perform any calculations in obtaining a centralized-equivalent estimate. In Theorem 3.1, we grouped these robots into the set Q. Except for robot 2, which invoked the Markov property, other robots within this set do not need to perform any calculations because the centralized-equivalent estimate has already been calculated by another robot (m), and is simply communicated to them. Therefore, there is no point in performing a redundant calculation for the same centralized-equivalent estimate. This suggests that with a decentralized approach, the amount of computation required is less than the amount required to run a centralized estimator on each robot. However, the difference will depend on how the network topology evolves over time. We will now conclude this section with the following theorem, which tells us that if all robots apply the Markov property whenever possible based on their local knowledge, all robots are still guaranteed to be able to obtain centralized-equivalent estimates.

Theorem 3.2: *Suppose that* $(\forall i)$, *robot* i *applies the Markov property when* $C_p(k_{c,i}, k_{e,i})$ *exists. Then* $(\forall C(k_c, k_e)), S_{i,k_{e,i}} \supseteq \{\mathsf{bel}(X_{k_c})\}, (\forall i)$ *where* $k_{e,i} \leq k_e$ *and* $\mathsf{bel}(X_{k_c})$ *is the centralized state estimate.*

Proof. When a checkpoint exists, we know according to Lemma 3.1 that partial checkpoints occurring at the same time also exist for all robots. We also know from Theorem 3.2 that regardless of when the Markov property is applied by each robot, partial checkpoints will always exists for all robots. Therefore, all robots are able to apply the Markov property in any order. Hence, the following mathematical statements are true before and after the Markov property is applied by all robots:

$$\begin{aligned} & (\forall C(k_c, k_e)), C_p(k_c, k_{e,i}) \text{ exists}, (\forall i)(k_{e,i} \leq k_e) \\ \Rightarrow \quad & (\forall C(k_c, k_e))\,, S_{i,k_{e,i}} \supseteq \{\mathsf{bel}(X_{k_c}), (\forall i)(k_{e,i} \leq k_e)\} \end{aligned}$$

When robot i applies the Markov property at $k_{e,i}$, the centralized state estimate will replace the equivalent information in its knowledge set (up to time k_c). When all robots have applied the Markov property, all robots will have the centralized state estimate. Furthermore, this will occur at $k_e = \max_i k_{e,i}$. □

With the above theorem, we are certain that robots can apply the Markov property based on local knowledge without affecting the ability for others to do so. It also confirms that all robots are able to obtain the centralized-equivalent state estimate at their partial checkpoint existence times even if each robot applies the Markov property whenever possible. Furthermore, we have shown that in having each robot obtain the centralized-equivalent estimate in a decentralized fashion, an outside observer (the overseer) is an unnecessary entity.

At this point, it may be useful to review how we arrived at Theorem 3.2. Referring to Fig. 2.6, we first looked at our problem of cooperative localization in a dynamic network from the perspective of an outside observer. We introduced the concept of a checkpoint in Definition 4, followed by Theorem 1.1, Lemma 1.1, and Theorem 1.2 for detecting the existence of a checkpoint. Next, we looked at the problem from the local perspective of a robot. We introduced the concept of a partial checkpoint in Definition 5, followed by Theorem 2.1, Lemma 2.1, and Theorem 2.2 for detecting the existence of a partial checkpoint. We then related partial checkpoints to checkpoints in Lemma 3.1, and used this to show in Theorem 3.1 that a robot's decision to invoke the Markov property does not affect other robots' abilities to do the same. Finally, this leads to Theorem 3.2, which shows that all robots can obtain a centralized-equivalent estimate (at certain times with latency dependent on the communication network topology) if they apply the Markov property whenever possible. Note that in Fig. 2.6, there is one additional theorem that we have not yet introduced. This theorem deals with the initial knowledge of robots necessary to guarantee that the centralized-equivalent estimate is obtainable by all robots, and will be presented in Section 2.4.

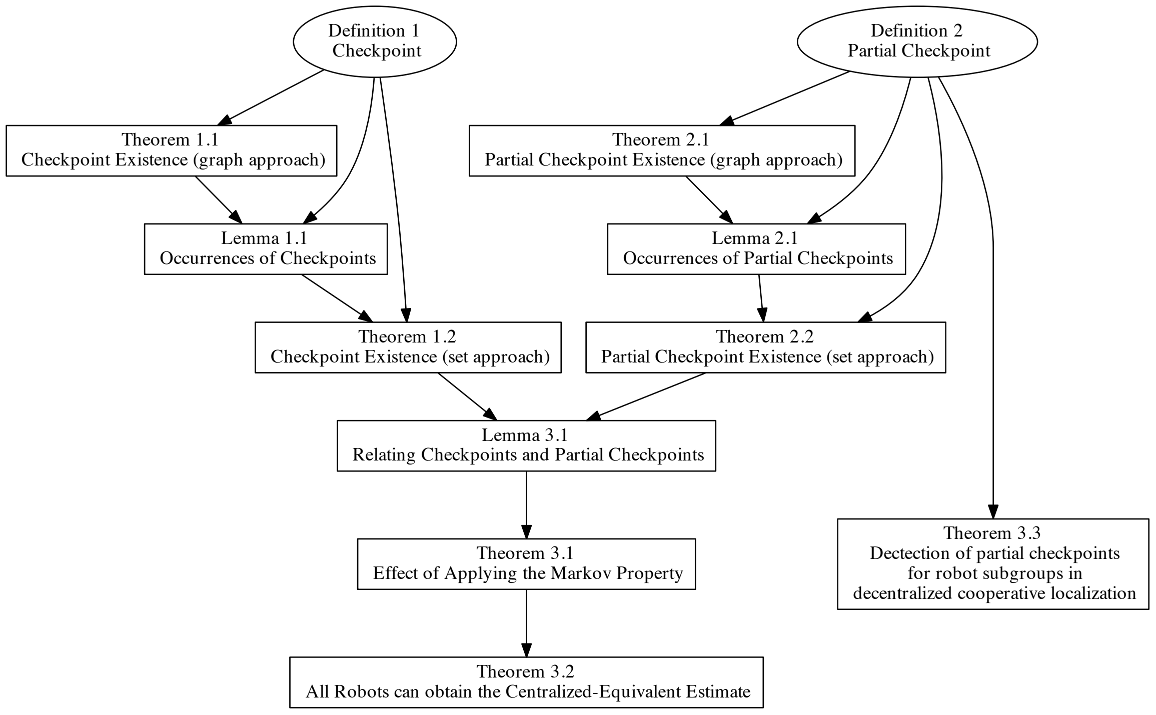

Definition 1
Checkpoint
Definition 2
Partial Checkpoint
Theorem 1.1
Checkpoint Existence (graph approach)
Theorem 2.1
Partial Checkpoint Existence (graph approach)
Lemma 1.1
Occurrences of Checkpoints
Lemma 2.1
Occurrences of Partial Checkpoints
Theorem 1.2
Checkpoint Existence (set approach)
Theorem 2.2
Partial Checkpoint Existence (set approach)
Lemma 3.1
Relating Checkpoints and Partial Checkpoints
Theorem 3.3
Dectection of partial checkpoints
for robot subgroups in
decentralized cooperative localization
Theorem 3.1
Effect of Applying the Markov Property
Theorem 3.2
All Robots can obtain the Centralized-Equivalent Estimate

Figure 2.6: A graphical summary of the relationship between theorems in this chapter.

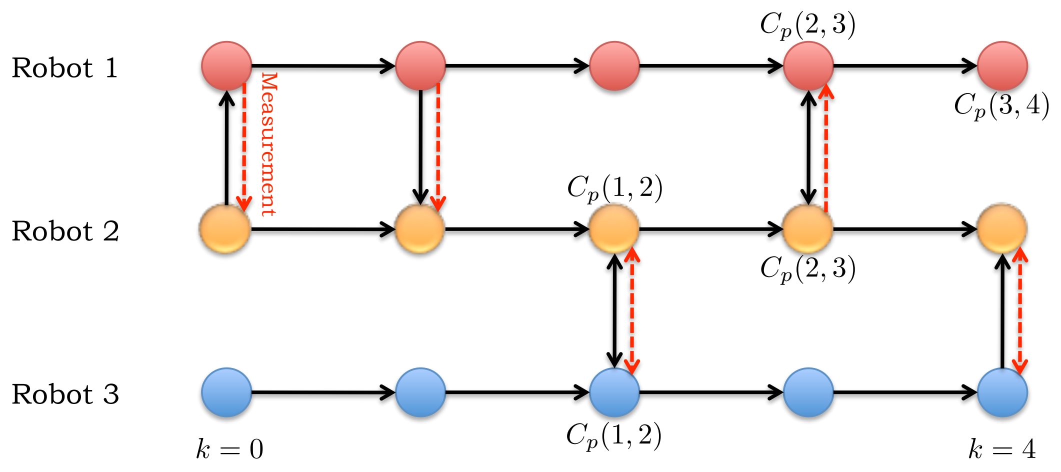
Previously, we have indicated that performing state estimation in a dynamic network requires careful consideration in applying the Markov property, to avoid cyclic updates, and proper processing of OOSMs. We have already shown that each robot should apply the Markov property whenever possible. In terms of cyclic updates, we can be sure that this will never occur in the centralized-equivalent approach because: (a) the knowledge set update rules ensure that there are no repeating data in a knowledge set, and (b) each robot will know whether an estimate is equivalent to the centralized estimate (which is never updated again). In terms of dealing with OOSMs, we only need to ensure that all information in a knowledge set is processed in the order in which it is produced to avoid this problem. These important results will be used to develop our decentralized state estimation algorithm.

Communication and Measurement Ranges

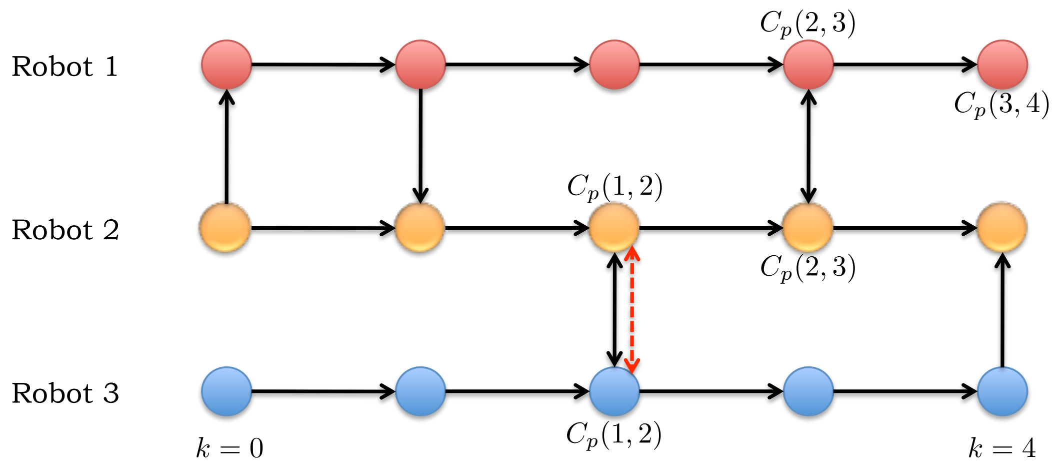
Up to this point, we have assumed that $r_{\mathsf{comm}} = r_{\mathsf{obs}}$, and implicitly assumed that communication and measurements are bi-directional. The applicability of the theorems presented remains the same regardless of whether communication range is different from measurement range, and whether they are uni-directional or bi-directional. This is because using Theorem 2.1 or 2.2, we ensure that each robot will apply the Markov property if and only if it has all the information necessary to calculate the centralized-equivalent estimate at the partial checkpoint occurrence time. Fig. 2.7 is an example illustrating this fact. In case 1, $r_{\mathsf{comm}} = r_{\mathsf{obs}}$ as we have previously assumed. Using the theorems that we have developed, robots 2 and 3 will both find $C_p(1, 2)$, then robots 1 and 2 will find $C_p(2, 3)$. Lastly, robot 2 will find $C_p(3, 4)$. In all partial checkpoint instances, it can be verified that all robots have gathered the required information to calculate the centralized estimate regardless of whether communication and measurements occur uni-directionally or bi-directionally. In case 2 ($r_{\mathsf{comm}} > r_{\mathsf{obs}}$), most of the measurements seen from the previous case do not occur. Still, all partial checkpoint instances are identical to case 1, and it can be again verified that the centralized estimates can be calculated at partial checkpoint occurrence times. In case 3 ($r_{\mathsf{comm}} < r_{\mathsf{obs}}$), most of the communication instances seen from case 1 do not occur, and it is not possible for robots 1 and 3 to calculate the centralized estimate at the timesteps shown. $C_p(1, 4)$ is the only partial checkpoint that occurs for robot 2, which is the same conclusion that Theorems 2.1 or 2.2 will provide when they are applied.

2.4 Applying the Theory

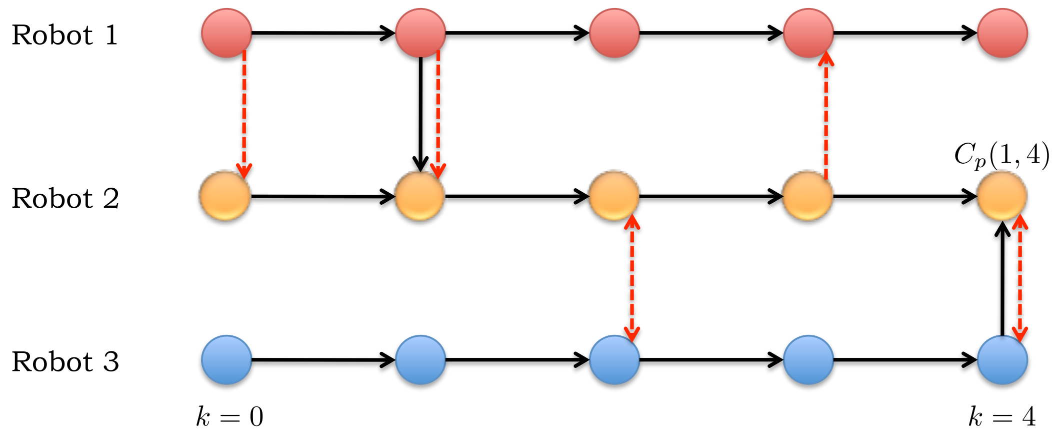
The theoretical development in the previous section provides the basis for developing our decentralized state estimation algorithm. We will first discuss the topic of initial knowledge, and show one of the scalable aspects of our method. This is followed by the detailed explanation of our decentralized state estimation algorithm, and examine computational complexity.



(a) $r_{\text{comm}} = r_{\text{obs}}$.



(b) $r_{\text{comm}} > r_{\text{obs}}$.



(c) $r_{\text{comm}} < r_{\text{obs}}$.

Figure 2.7: The red dashed edges indicate measurements between robots in the above figures. Even when communication and measurement ranges are different, or uni-directional, Theorems 2.1 or 2.2 (detection of a partial checkpoint) allows us to correctly determine when each robot can apply the Markov property to obtain the centralized state estimate.

2.4.1 Initial Knowledge of Robots

For a system of robots, we have assumed that each robot initially only has a belief for its own state. Hence, each robot is only aware of its own existence, and a robot will only know of the existence of another robot at first contact (i.e., when communication or a measurement is made between the robots). Correlation between beliefs of two robots is generated through relative measurements (recall we are assuming that there are no landmarks for the moment). When the beliefs over all robots are correlated, any odometry and measurement data for a single robot will not only influence the belief over the individual robot's state, but also the belief over all robot states. It is precisely this reason why there is the need for the notion of a checkpoint, so that all odometry and measurement data are accounted for to produce a centralized-equivalent estimate in a decentralized manner.

Due to the independence of beliefs before an encounter, individual (or a group of) robots can be treated as independent subsystems. In other words, the system of all robots can be decoupled into smaller subsystems, and into $|N|$ subsystems (of individual robots) at the initial timestep.

Suppose Q_1 and Q_2 are two sets of robots that have never encountered each other before, and let $\mathsf{bel}(X_{Q_1,k})$ and $\mathsf{bel}(X_{Q_2,k})$ represent the beliefs over the states of the two groups, respectively. Before accounting for any measurements between the two groups, statistical independence allows the state estimate for the combined system to be written as [79]:

$$\mathsf{bel}(X_{Q_1,k}, X_{Q_2,k}) = \mathsf{bel}(X_{Q_1,k})\mathsf{bel}(X_{Q_2,k}) \tag{2.5}$$

To implement this, we will always combine state estimates made at the same timestep if they are found within a knowledge set using (2.5). Measurements can then be processed to couple and update the states in the combined state estimate. This seemingly simple process contributes to the scalability of our decentralized state estimation algorithm, in the sense that it is unnecessary for each robot to initially know how many robots there are. However, this is only possible when communication is bi-directional and when the communication range limit is greater than or equal to the measurement range limit (i.e., the coupling of state estimates is detectable). Otherwise, each robot will initially need to know the total number of robots in the team. To formalize this, we have the following theorem. We will assume that a robot knows of another robot's existence if its knowledge set contains any information (odometry or measurement) related to that robot.

Theorem 3.3: *Assume that $r_{\mathsf{comm}} \geq r_{\mathsf{obs}}$ and that robots do not initially know the total number of robots in the team. In decentralized cooperative localization, if robot i detects $C_p(k_{c,i}, k_{e,i})$ based on robots known to it, $N_{i,k_{e,i}}$, and applies the Markov Property when the partial checkpoint exists, then robot i is guaranteed to obtain the centralized-equivalent estimate for all robots when it knows of the existence of all robots.*

Proof. For this proof, we want to show that robot i's estimate of robots known to itself is always independent of the estimate of robots not known to it. This allows us to combine these estimates at any time as shown in (2.5). Assume that $j \in N_{i,k_m}$, meaning that robot j is known to robot i at some timestep k_m (e.g., the time of a measurement). We need to consider two scenarios: a) what happens when robot j obtains a measurement of any other robot, n, and b) what happens when robot j is measured by another robot, n. Dealing with the former case first, assume that $\mathbf{y}^{n,j}_{j,k_m}$ exists.

$$\begin{aligned} & \mathbf{y}^{n,j}_{j,k_m} \text{ exists} \\ \Rightarrow\ & \mathbf{y}^{n,j}_{j,k_m} \in S^{-}_{j,k_m} \\ \Rightarrow\ & \mathbf{y}^{n,j}_{j,k_m} \in S_{j,k_m} \\ \Rightarrow\ & n \in N_{j,k}, k \geq k_m \end{aligned}$$

The result above basically shows that if robot j measures another robot, it will know of its existence, which is always expected. Now if a partial checkpoint that occurs at k_m exists for robot i (i.e., $C_p(k_m, k_{e,i})$ exists), then it follows that:

$$\begin{aligned} \Rightarrow\ & \mathbf{y}^{n,j}_{j,k_m} \in S_{j,k_m} \\ \Rightarrow\ & \mathbf{y}^{n,j}_{j,k_m} \in S_{i,k_{e,i}} \\ \Rightarrow\ & n \in N_{i,k_{e,i}} \end{aligned}$$

Therefore, for any robot n that is measured by a robot j, which is known to exist to robot i, robot n's existence will also be known to robot i when it produces a centralized-equivalent estimate for the timestep, k_m, at which robot n was first observed.

Next we want to consider the latter case when a robot known to robot i (i.e., $j \in N_i$) is measured by another robot, n, at timestep k_m. Assume that $\mathbf{y}^{j,n}_{n,k_m}$ exists.

$$\begin{aligned} & \mathbf{y}^{j,n}_{n,k_m} \text{ exists} \\ \Rightarrow\ & \mathbf{y}^{j,n}_{n,k_m} \in S^{-}_{n,k_m} \\ \Rightarrow\ & \mathbf{y}^{j,n}_{n,k_m} \in S_{j,k_m} \\ \Rightarrow\ & n \in N_{j,k}, k \geq k_m \end{aligned}$$

Robot j knows about any measurements made of it because we have assumed that $r_{\mathsf{comm}} \geq r_{\mathsf{obs}}$. So if robot n measures j, robot n will communicate the measurement to robot j. Now if we consider the existence of a partial checkpoint that occurs at k_m for robot i (i.e., $C_p(k_m, k_{e,i})$ exists), then it follows that:

$$\begin{aligned} &\Rightarrow \quad \mathbf{y}^{j,n}_{n,k_m} \in S_{j,k_m} \\ &\Rightarrow \quad \mathbf{y}^{j,n}_{n,k_m} \in S_{i,k_{e,i}} \\ &\Rightarrow \quad n \in N_{i,k_{e,i}} \end{aligned}$$

Therefore, for any robot n that makes a measurement of robot j, which is known to exist to robot i, robot n's existence will also be known to robot i when it produces a centralized-equivalent estimate for the timestep, k_m, at which robot n measured robot j.

Finally, since all measurements made by robots in N_i and made to robots in N_i are known to robot i, it can produce a centralized-equivalent estimate for the robots in N_i that is statistically independent from the estimate of robots not in N_i (i.e., all the robots in $\overline{N_i}$). Mathematically, when a partial checkpoint, $C_p(k_{c,i}, k_{e,i})$, exists.

$$\begin{aligned} &\left(\forall \mathbf{y}^{n,j}_{j,k_m} \in S_{i,k_{e,i}}\right) \left(\forall \mathbf{y}^{j,n}_{n,k_m} \in S_{j,k_{e,i}}\right), \{n, j\} \in N_i \\ \Rightarrow \quad &\mathsf{bel}\left(X_{N_i,k_{c,i}}, X_{\overline{N_i},k_{c,i}}\right) = \mathsf{bel}\left(X_{N_i,k_{c,i}}\right) \mathsf{bel}\left(X_{\overline{N_i},k_{c,i}}\right) \end{aligned}$$

In conclusion, the centralized-equivalent estimate for all robots can always be recovered by robot i (if it is given the estimate for the robots in $\overline{N_i}$), using (2.5). □

It is of interest to note that in the case where $r_{\mathsf{comm}} < r_{\mathsf{obs}}$, we can not guarantee that measurements made on robots known to robot i (i.e., robot $j \in N_i$) will be known by robot i. In this case, we cannot guarantee that the centralized-equivalent estimate is recoverable by robot i. In other words, we cannot guarantee that $\mathsf{bel}\left(X_{N_i,k_{c,i}}\right)$ and $\mathsf{bel}\left(X_{\overline{N_i},k_{c,i}}\right)$ are statistically independent. In this case, it would be necessary for robot i to initially know the total number of robots in the team, so that partial checkpoints are detected based on the knowledge of all robots.

2.4.2 The Decentralized Cooperative Localization Algorithm

Algorithm 1 is designed to perform decentralized state estimation in a scalable manner and is guaranteed to work in a dynamic mobile robot network wherein connectivity between all robots is not guaranteed. The same algorithm is implemented on each robot and iterates every timestep. The required inputs are:

k, the current timestep k

$\mathbf{u}_{i,k}$, odometry data

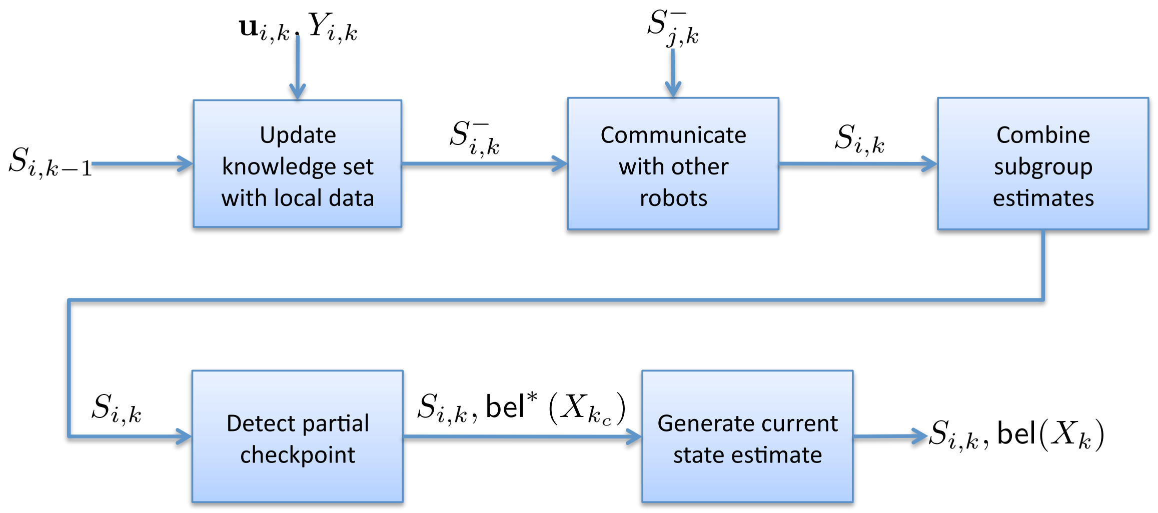
$Y_{i,k}$, measurements

$S_{i,k-1}$, the latest knowledge set

$S_{j,k}$ ($\forall j \in R_{i,k}$), the knowledge sets of all robots connected to robot i

Fig. 2.8 is a graphical overview of the algorithm. Each robot first updates its knowledge set with the current odometry and measurements. If other robots are within communication range, new information from other robots are appended to the knowledge set. Within this updated knowledge set, we apply Theorem 2.2 to detect partial checkpoints and apply the Markov property at the partial checkpoint time if possible. Lastly, the current state estimate is generated. Essentially, each robot is running its own centralized state estimator on the available information from the partial checkpoint occurrence timestep to the current timestep. This estimate is temporary until a partial checkpoint that exists at the current time exists.

Note that in most cases, it is not possible to reuse the current state estimate at a later time to generate the centralized state estimate for the same timestep. This is due to the OOSM problem.



$\mathbf{u}_{i,k}, Y_{i,k}$
$S^{-}_{j,k}$
$S_{i,k-1}$
Update knowledge set with local data
$S^{-}_{i,k}$
Communicate with other robots
$S_{i,k}$
Combine subgroup estimates
$S_{i,k}$
Detect partial checkpoint
$S_{i,k}$, bel$^{*}(X_{k_c})$
Generate current state estimate
$S_{i,k}$, bel(X_k)

Figure 2.8: An iteration of the decentralized state estimation algorithm, showing how a knowledge set is updated at timestep k, as well as the calculation of the current belief. Line numbers correspond to those in Algorithm 1.

At the very first iteration of our algorithm, we assume that each robot initially only has a state estimate of itself in its knowledge set. On line 1, we update the knowledge set of robot i by implementing (2.3) and (2.4). Line 2 determines the set of all robots known

Algorithm 1: Decentralized cooperative localization(k, $\mathbf{u}_{i,k}$, $Y_{i,k}$, $S_{i,k-1}$, $S_{j,k}$ ($\forall j \in R_{i,k}$))

```
1  $S_{i,k} \leftarrow S_{i,k-1} \cup \{\mathbf{u}_{i,k}\} \cup \{Y_{i,k}\} \bigcup_{j \in R_{i,k}} S^{-}_{j,k}$
2  $N_{i,k} \leftarrow \{Q | (\forall \mathsf{bel}(\mathbf{X}_{Q,k_s}) \in S_{i,k}), (k_s \leq k)\}$
3  repeat
4      $\mathsf{flag}_{\mathsf{repeat}} = \mathsf{false}$
5      $\{k_{s_1}, Q_1\} \leftarrow$ find smallest $k_{s_1}$ such that $\mathsf{bel}^*(X_{Q_1}, k_{s_1}) \in S_{i,k}$
6      $k_{s_2} \leftarrow k$
7      if $Q_1 \neq N_{i,k}$ then
8          $\{k_{s_2}, Q_2\} \leftarrow$ find smallest $k_{s_2}$ such that $\mathsf{bel}^*(X_{Q_2}, k_{s_2}) \in S_{i,k}$ $(Q_1 \neq Q_2)$
9      end
10     for $k_c \leftarrow k_{s_2} : k_{s_1}$ do
11         if $U_{Q_1,k_c} \in S_{i,k}$ or $k_c = k_{s_1}$ then
12             $\tilde{S}_{i,k_c} \leftarrow S_{i,k} - \{U_{Q_1,k_r}, Y_{Q_1,k_r}\}$ $(\forall k_r > k_c)$
13             $\mathsf{bel}^*(X_{Q_1,k_c}) \leftarrow p\left(X_{Q_1,k_c} | \tilde{S}_{i,k_c}\right)$
14             $S_{i,k} \leftarrow S_{i,k} \cup \mathsf{bel}^*(X_{Q_1,k_c})$
15             $S_{i,k} \leftarrow S_{i,k} - \{U_{Q_1,k_r}, Y_{Q_1,k_r}, \mathsf{bel}^*(X_{Q_1,k_r})\}$ $(\forall k_r \leq k_c)$
16             break
17         end
18     end
19     if $\{\mathsf{bel}^*(X_{Q_1,k_c}), \mathsf{bel}^*(X_{Q_2,k_c})\} \in S_{i,k}$ then
20         $\mathsf{bel}^*(X_{Q_3,k_c}) \leftarrow \mathsf{bel}^*(X_{Q_1,k_c})\, \mathsf{bel}^*(X_{Q_2,k_c})$
21         $S_{i,k} \leftarrow S_{i,k} \cup \mathsf{bel}^*(X_{Q_3,k_c}) -$
               $\{\mathsf{bel}^*(X_{Q_1,k_c}), \mathsf{bel}^*(X_{Q_2,k_c})\}$
22         $\mathsf{flag}_{\mathsf{repeat}} = \mathsf{true}$
23     end
24 until $\mathsf{flag}_{\mathsf{repeat}} = \mathsf{false}$
25 $\{k_{s_1}, Q_1\} \leftarrow$ find smallest $k_{s_1}$ such that $\mathsf{bel}^*(X_{Q_1}, k_{s_1}) \in S_{i,k}$
26 while $Q_1 \neq N_{i,k}$ do
27     $\{k_{s_2}, Q_2\} \leftarrow$ find smallest $k_{s_2}$ such that
28     $\mathsf{bel}(X_{Q_2}, k_{s_2}) \in S_{i,k}$ $(Q_1 \neq Q_2)(k_{s_1} \leq k_{s_2})$
29     $\tilde{S}_{i,k_{s_2}} \leftarrow S_{i,k} - \{U_{Q_1,k_u}, Y_{Q_1,k_r}\}$ $(\forall k_u > k_{s_2})$
30     $\mathsf{bel}\left(X_{Q_1,k_{s_2}}\right) \leftarrow p\left(X_{Q_1,k_{s_2}} | \tilde{S}_{i,k_{s_2}}\right)$
31     $\mathsf{bel}\left(X_{Q_1,k_{s_2}}, X_{Q_2,k_{s_2}}\right) \leftarrow \mathsf{bel}\left(X_{Q_1,k_{s_2}}\right) \mathsf{bel}\left(X_{Q_2,k_{s_2}}\right)$
32     $Q_1 \leftarrow Q_1 \cup Q_2$
33     $k_{s_1} \leftarrow k_{s_2}$
34 end
35 $\mathsf{bel}(X_{N_i,k}) \leftarrow p(X_{N_i,k} | S_{i,k})$
36 return $\{\mathsf{bel}(X_{N_i,k}), S_{i,k}, N_{i,k}\}$
```

to i by looking for part beliefs $\mathsf{bel}^*(X_{Q,k_s})$ in $S_{i,k}$, where Q represents a set of robots. Note that bel^* indicates a belief that is equivalent to the state estimate obtainable using a centralized state estimator. The loop beginning on line 3 repeats according to the flag variable set on line 4. The purpose of this loop is to systematically combine multiple beliefs for independent subgroups of robots in $S_{i,k}$. To do this, on line 5 we search for the earliest state estimate $\mathsf{bel}^*\left(X_{Q_1}, k_{s_1}\right)$ in the knowledge set. If $Q_1 = N$, then we have the belief over all known robots already. Otherwise, we search for the next earliest estimate $\mathsf{bel}^*\left(X_{Q_2}, k_{s_2}\right)$ on line 8. The intention here is to calculate $\mathsf{bel}^*\left(X_{Q_1}, k_{s_2}\right)$ so that the beliefs over the two groups can be combined.

The search for a partial checkpoint (for system Q_1) begins with the 'for' loop on line 10. If $Q_1 = N$, we will attempt to look for a partial checkpoint that is closest to the current timestep, and this is why k_{s_2} is initially set equal to k on line 6. Otherwise, we will attempt to find a partial checkpoint at k_{s_2}. If k_{s_1} and k_{s_2} are the same, the partial checkpoint search is skipped and we proceed directly to line 19 and combine the estimates found on lines 5 and 8. Line 11 uses Theorem 2.2 to detect the existence of a partial checkpoint. If one is found, we use the knowledge up to the partial checkpoint time determined on line 12 to obtain the state estimate on line 13. The new estimate is entered into the knowledge set on line 14 and we proceed to discard information replaceable by bel^* on line 15. On line 16, we break out of the partial checkpoint search since one has been found.

Line 19 checks if there are two estimates for the same timestep in the knowledge set. If a pair is found, we proceed to combine them according to (2.5) on line 20, and update the knowledge set on line 21. The repeat flag defined on line 4 is made true here because there may be other subgroup beliefs in the knowledge set that can be combined (i.e., in the next iteration of the line 3–24 loop, the newly combined belief on line 20 becomes the belief we will find on line 5, and we will attempt to find the next earliest belief for another subgroup on line 8).

After searching for partial checkpoints, we turn our attention to determining the state estimate for the current timestep. This is necessary because the occurrence time for the partial checkpoint last discovered may not be the current timestep. In fact, occurrence time is equal to the current (existence) time if and only if a robot is able to communicate with all other robots at the current timestep. We first need to consider to presence of multiple estimates (for independent groups of robots at different timesteps) that may still exist in the knowledge set. We again begin with the earliest state estimate in the knowledge set on line 25, and aim to produce a state estimate at the time of the next earliest estimate (line 27) in the knowledge set so that the two can be combined. This process repeats in the loop between lines 26 and 33 until we have a single state estimate over the states of all known robots. The current state estimate is then determined on line 35, which is based on the estimate at the last partial checkpoint and any information since then to the current time. Hence, it uses all the information available and is the best estimate that can be produced at the current time. In calculating this current estimate, we may not have all the information required to make an estimate that is equivalent to one produced by the centralized state estimator (i.e., a partial checkpoint does not exist). In this situation, we will assume the last known velocity for robots from which we do not have odometry data, but this is only temporary.

There are some important points to highlight about Algorithm 1. First, as mentioned already, it is unnecessary for a robot to initially know how many robots there are in the system. Second, since our algorithm is based on the information flow graph, the sequence of communication between multiple robots or delays in communication can be handled naturally without any changes to our algorithm. This is one of the practical advantages with our framework. Third, any recursive filtering method can be used on lines 13, 30 and 35. To give a few examples, the EKF, the *Unscented Kalman Filter* (UKF), and the PF [53], could be used in our decentralized state estimation framework. Thus, the algorithm is very general and is widely applicable in any situation in which there is a need to perform decentralized state estimation in a dynamic network. Furthermore, a centralized-equivalent state estimate can always be calculated at the time of a partial checkpoint. Although the state estimate at the current time may be suboptimal due to missing information, the equivalent centralized state estimate is guaranteed to be obtainable later. Once again, a robot assumes that another robot maintains its last known velocity until its odometry is known.

2.4.3 Complexity Analysis

Since we are exploiting the Markov property, computation and memory usage are limited provided that all robots are able to detect partial checkpoints in the future, which is a reasonable assumption if their task is to cooperatively localize. The computational complexity of Algorithm 1, its memory usage, as well as communication bandwidth requirements will vary depending on the number of timesteps since the previous partial checkpoint, $k - k_c$, the number of states that need to be estimated, n (which is proportional to the number of robots, $|N|$), and the filtering method selected for use within Algorithm 1. The frequency at which partial checkpoints occur depends on factors such as communication range as well as the size of the workspace. Fig. 2.9 illustrates the worst-case scenario for a 4-robot team. The scenario shows that while relative measurements are made between all robots at all timesteps, information exchange does not occur with one of the robots (Robot 4). This may occur if Robot 4 fails to communicate. Hence, new partial checkpoints will not come into existence for all the robots, and information will continue to accumulate in each robot's knowledge set. This will continue until communication is re-established to Robot 4.

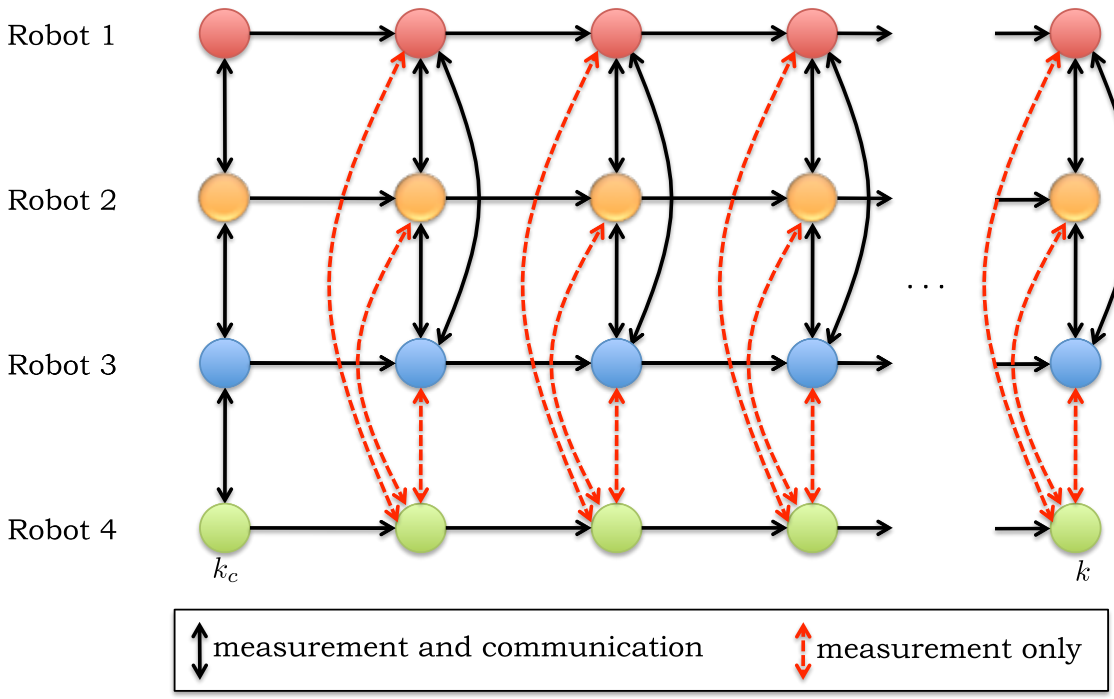


Figure 2.9: The worst-case scenario in decentralized cooperative localization in terms of memory usage, computation, and communication

Assume that the EKF is used in the above scenario. Since $n = c|N|$ where c is a constant representing the number of states per robot, we can simplify this complexity analysis by letting $c = 1$ without affecting the end result. In a centralized estimator (where all robots can exchange information at all timesteps), there are $(n^2 - n)$ measurements at each timestep since every robot makes a measurement of every other robot. Assuming that the dimension of each measurement is constant (i.e., all measurements always provide only range and bearing information), and we process measurements sequentially, the computational complexity of processing each measurement is $O(n^2)$ (from the Kalman gain calculation and the covariance update steps). With $(n^2 - n)$ measurements, the overall computational complexity is therefore $O(n^4)$ for the centralized estimator. Furthermore, storing the covariance matrix and measurements requires memory usage of $O(n^2)$.

Using the EKF in our decentralized estimator, memory usage will be of $O((k-k_c)n^2)$ for each robot in the worst-case scenario. This is because it is necessary to keep all information in the knowledge set that comes after k_c and up to the current timestep, k. For the worst-case scenario shown, at every timestep, all robots except Robot 4 communicate with each other, passing on measurement and odometry information accumulated since the last partial checkpoint to $n-2$ robots (remember that $n = |N|$). This makes the communication bandwidth requirement of $O((k-k_c)n^3)$ for each robot. The computational complexity of calculating the current state estimate is $O((k-k_c)n^4)$ for each robot, but only when a new partial checkpoint is discovered since the calculation must then be performed from the last partial checkpoint time. Otherwise, by knowing the state estimate from the previous timestep $(k-1)$, the computational complexity of calculating the current state estimate is $O(n^4)$. In general, the difference in computational complexity from the centralized approach (i.e., the $k-k_c$ factor) is the result of network connectivity (i.e., the cost of operating in a dynamic and sparse network).

In practice, performing state estimation at high frequency may cause difficulties in real-time implementations as the number of timesteps since the previous partial checkpoint $(k-k_c)$ increases at a high rate. A practical solution may be to aggregate odometry data between measurements to in effect increase the size of each timestep.

Recall from Theorem 3.1 that the ability for robots to pass on state estimates will eliminate the need for some other robots to perform the calculations for the same state estimate, hence reducing computational cost. Also, if robots keep track of the information they have already sent to other robots, redundancy in communication can be reduced in exchange for increase in memory usage. Furthermore, robots consistently in contact with each other could form a subgroup and temporarily store their current state estimate to reduce the amount of computation required for the next timestep (assuming that no past information is added to their knowledge sets).

2.5 Simulations and Performance Analysis

In the theoretical development of a checkpoint, we showed that a centralized-equivalent estimate can be obtained by all robots when a checkpoint exists (or by a particular robot when a partial checkpoint exists). It is of interest to compare the performance of the decentralized state estimation algorithm (Algorithm 1) against a centralized state estimator. For this purpose, simulations were performed for a group of uniquely-identifiable robots moving in a workspace in which each robot does not initially know the total number of robots in the team. The intention of the simulation is to have each robot estimate the states of all robots known to itself. Communication range and measurement range are set equal, and are limited so that the robots are in a dynamic network that is not always fully connected. Note, the centralized state estimator would simply not work under these assumptions. However, to generate results against which we can compare, we give robots the ability to always communicate with each other at all timesteps for the centralized estimator only.

2.5.1 Simulation Setup

The *Extended Kalman Filter (EKF)* [53] was used as the filtering method on lines 13, 20 and 35 of Algorithm 1. Note that many other recursive filtering methods could also be used. The state of a robot included its position and orientation, $(x_{i,k}, y_{i,k}, \theta_{i,k})$. Two different discrete-time unicycle motion models were used for the simulation. Model (2.6) was used when the angular velocity component of the input, ω_k, equaled zero (or very close to zero for numerical stability). This model can be derived from the continuous-time unicycle model using the forward Euler method:

$$\begin{bmatrix} x_{i,k} \\ y_{i,k} \\ \theta_{i,k} \end{bmatrix} = \begin{bmatrix} x_{i,k-1} \\ y_{i,k-1} \\ \theta_{i,k-1} \end{bmatrix} + \begin{bmatrix} \Delta T \cos\theta_{i,k-1} & 0 \\ \Delta T \sin\theta_{i,k-1} & 0 \\ 0 & \Delta T \end{bmatrix} \begin{bmatrix} v_{i,k} \\ \omega_{i,k} \end{bmatrix} \tag{2.6}$$

In all other cases, motion model (2.7) was used instead. In comparison to (2.6), model (2.7) is a better discrete-time approximation of the continuous-time unicycle model. Model (2.7) can be derived by holding the velocity inputs constant over the sampling time, ΔT [53]. However, the model can not be used when ω_k equals zero:

$$\begin{bmatrix} x_{i,k} \\ y_{i,k} \\ \theta_{i,k} \end{bmatrix} = \begin{bmatrix} x_{i,k-1} \\ y_{i,k-1} \\ \theta_{i,k-1} \end{bmatrix} + \begin{bmatrix} -\frac{v_k}{\omega_k} \sin\theta_{i,k-1} + \frac{v_k}{\omega_k} \sin\left(\theta_{i,k-1} + \omega_k \Delta T\right) \\ \frac{v_k}{\omega_k} \cos\theta_{i,k-1} - \frac{v_k}{\omega_k} \cos\left(\theta_{i,k-1} + \omega_k \Delta T\right) \\ \omega_k \Delta T \end{bmatrix} \tag{2.7}$$

The inputs to the both motion models were forward and angular velocities, (v_k, ω_k), corrupted by independent zero-mean Gaussian noise, $\boldsymbol{\epsilon}_k$:

$$\mathbf{u}_k = \begin{bmatrix} v_k \\ \omega_k \end{bmatrix} = \begin{bmatrix} \overline{v}_k \\ \overline{\omega}_k \end{bmatrix} + \boldsymbol{\epsilon}_k, \quad \boldsymbol{\epsilon}_k \sim \left(\mathbf{0}, \begin{bmatrix} \sigma_v & 0 \\ 0 & \sigma_\omega \end{bmatrix} \right)$$

When robot i observed another robot j within range r_{obs}, it measured the relative range, $r_k^{j,i}$, and bearing, $\phi_k^{j,i}$, of robot j with respect to itself. Each measurement component was assumed to contain independent zero-mean additive Gaussian noise:

$$\mathbf{y}_{i,k}^{j,i} = \begin{bmatrix} r_k^{j,i} \\ \phi_k^{j,i} \end{bmatrix} = \begin{bmatrix} \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \\ \arctan\left(\frac{y_{j,k} - y_{i,k}}{x_{j,k} - x_{i,k}}\right) - \theta_{i,k} \end{bmatrix} + \boldsymbol{\delta}_k \quad \boldsymbol{\delta}_k \sim \left(\mathbf{0}, \begin{bmatrix} \sigma_r & 0 \\ 0 & \sigma_\phi \end{bmatrix}\right) \tag{2.8}$$

Simulations started with robots at a random pose, and each had an estimate of its own pose. These estimates were all expressed within the same reference frame. Robots moved to random waypoints in the bounded workspace using a feedback control law similar to that presented in [80, 81]. All simulations ended after 10000 timesteps.

Non-dimensional Parameterization

We postulate that the number of robots, $|N|$, the size of the workspace, A, (or robot density ρ), and communication range, r_{comm}, are all the major factors that will influence the connectivity of the robot network and localization performance. Therefore, it is necessary to test if and how simulation results are affected by varying these parameters. To reduce the dimensionality of this analysis, Buckingham's Pi Theorem [82] is applied to generate the dimensionless variables:

$$\begin{aligned} \pi_1 &= |N| \\ \pi_2 &= \frac{|N|}{A} r_{\mathsf{comm}}^2 = \rho r_{\mathsf{comm}}^2 \end{aligned}$$

Since we have 3 variables ($|N|$, A, r_{comm}), and the units of these contain only 1 fundamental quantity (distance), this enables us to use $3 - 1$ dimensionless parameters to analyze the effects of each of the three variables on localization performance.

Network Connectivity

There have been many studies on network connectivity in the past for random graphs. A network is *connected* at a given timestep if there exists a path between every pair of nodes. This is true on the information flow graph if at timestep k a path exists from $v(i,k)$ to $v(j,k)$, $(\forall i, j)$. Furthermore, we define the *frequency of connectivity* as the percentage of time that a robot network is connected. This is shown in Fig. 2.10 for different numbers of (stationary) robots randomly populated in a workspace 1000 times to obtain an average at each data point. The trend observed also applies to moving robots [83], and we have also confirmed this through simulation.

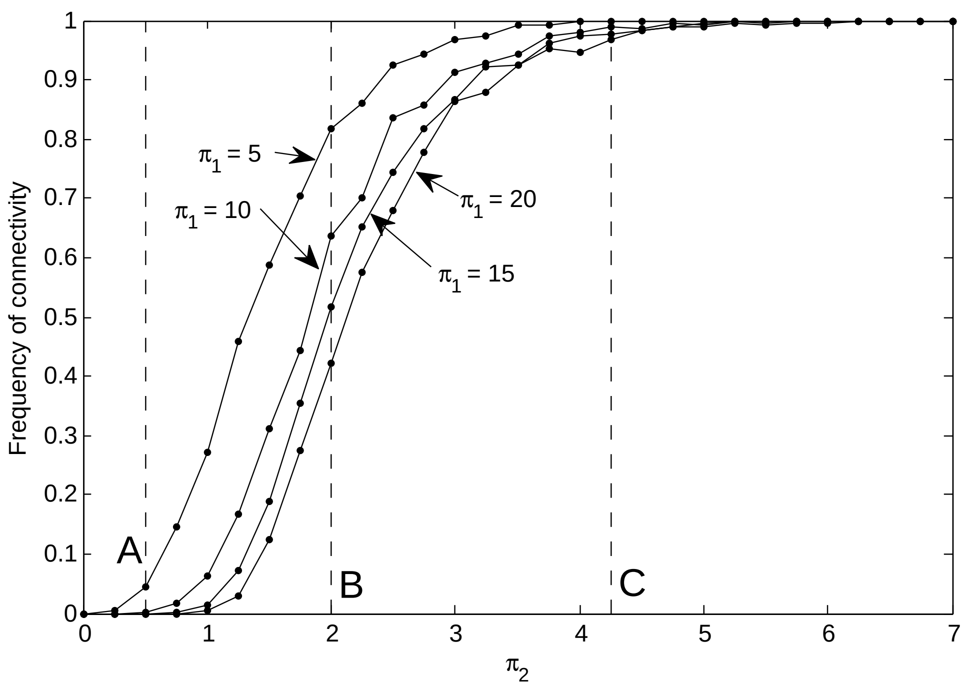


Figure 2.10: The frequency of connectivity observed for different numbers of stationary robots.

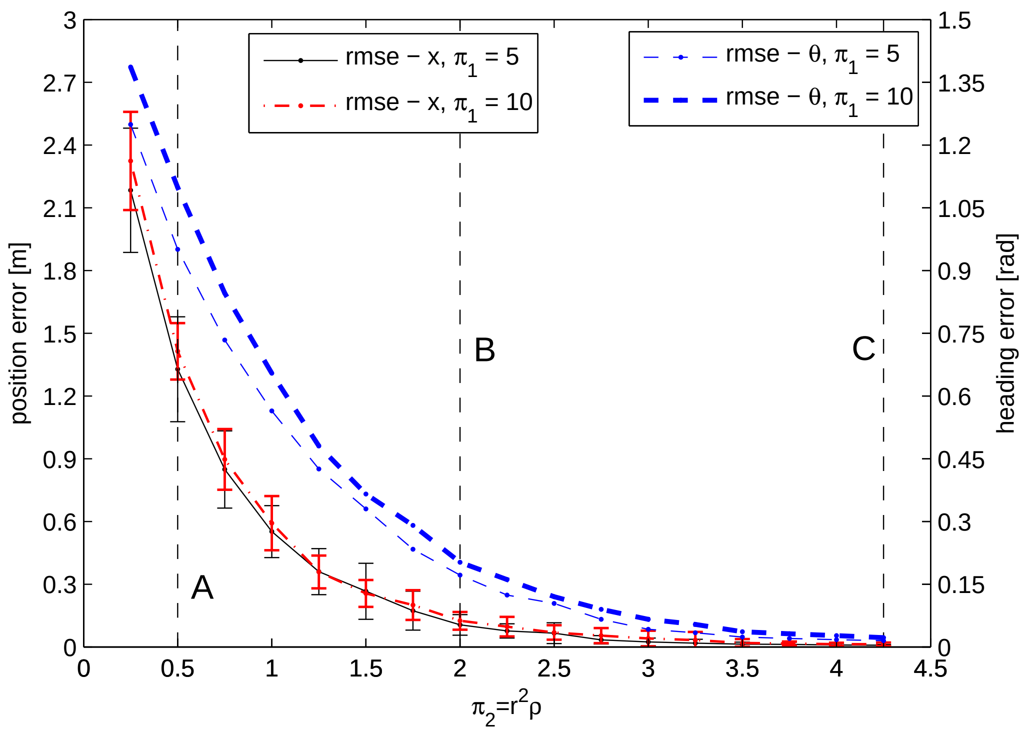


Figure 2.11: Average error between decentralized and centralized state estimates

The zero-to-one transition phenomenon observed in Fig. 2.10 is typical for *Bernoulli random graph models* wherein connections between robots are determined based on frequency (and not distance) [84], as well as the *fixed-radius-random-graph-model* used for generating our robot network [85]. Furthermore, we note that as π_1 increases, the π_2 value at which phase transition occurs, as well as the steepness of the transition also increases to approach an asymptotic curve. This corresponds to the findings in [86].

2.5.2 Estimator Performance Results

The network connectivity phase transition is important in providing us with an indication of how our decentralized state estimation algorithm performs. Low π_2 values can be interpreted as low robot density or short communication range. Under these conditions, robots are rarely in contact with each other and are infrequently establishing partial checkpoints. Conversely, high π_2 values (after the phase transition) correspond to high robot density or long communication range. With these conditions, robots are frequently in contact with each other and establishing partial checkpoints. The range of π_2 values at which the phase transition occurs is an indication of when the implementation of our decentralized state estimate algorithm becomes advantageous.

We will first present overall performance results collected over hundreds of simulation trials. Fig. 2.11 plots the root-mean-squared error between the decentralized state estimates produced by our algorithm and the centralized state estimates for x position (the results for y position are very similar), along with 2-standard-deviation error bars, as well as orientation, θ. The averaged error at each data point is obtained over 50 simulation trials.

In Fig. 2.10 and Fig. 2.11, we have identified three specific π_2 values: Point A ($\pi_2 = 0.5$) corresponds with low frequency of connectivity, point B ($\pi_2 = 2.0$) is within the connectivity phase transition region, and point C ($\pi_2 = 4.25$) corresponds with high frequency of connectivity. Over the phase transition, the difference between the decentralized and centralized estimates reduces drastically. This is where our decentralized algorithm begins to show its merits. At point C, connectivity has a high likelihood of being maintained and the difference in the decentralized and centralized estimates approaches zero. Note that although the performance between the two is almost identical, the centralized state estimator will only work if connectivity is guaranteed for all times.

Memory usage is limited since our algorithm makes use of the Markov property. Actual memory usage will depend on how many robots there are in the network (i.e., the number of states to estimate), and the frequency of partial checkpoint occurrence.

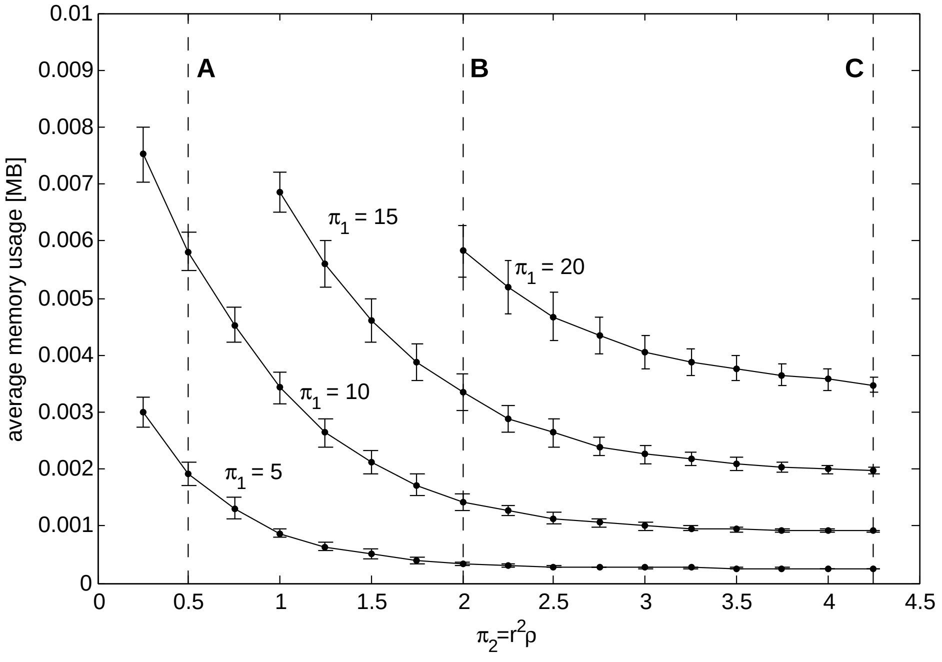
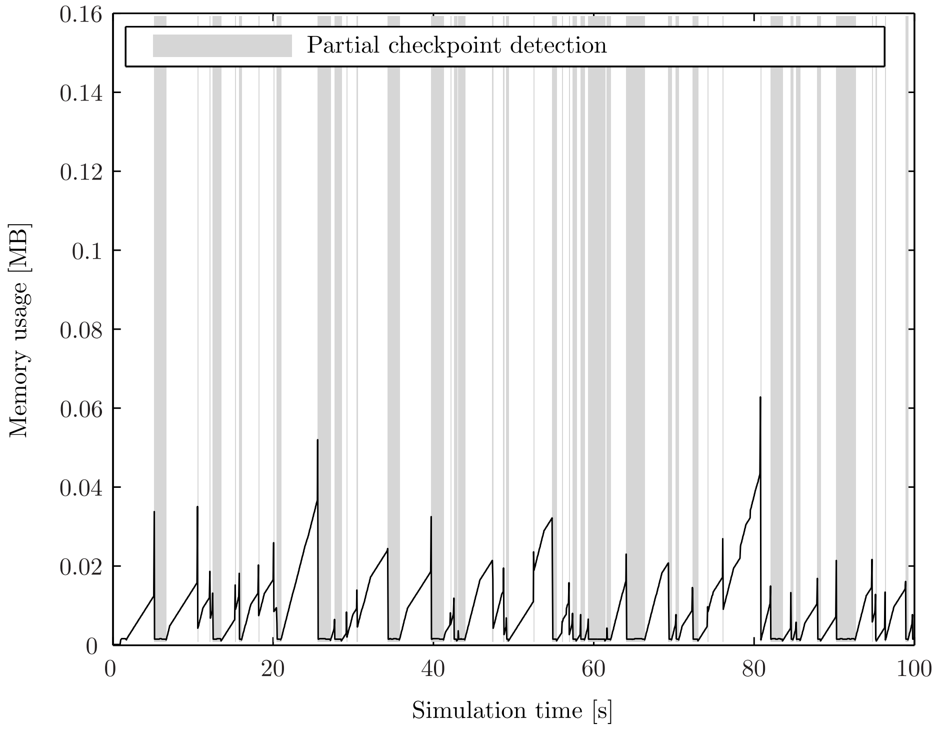


Figure 2.12: Average memory usage of a robot for decentralized state estimation

Fig. 2.12 is a plot of average memory usage with 2-standard-deviation bars. Each plotted point represents the averaged results of over 50 simulation runs. At low frequency of connectivity settings, partial checkpoints occur infrequently and each robot must store all its accumulated data in the duration between partial checkpoints, leading to high average memory usage. As π_2 increases, robot encounters occur more often and the frequency of partial checkpoint occurrence also increases, leading to reductions in memory usage. Note, as we increase the number of robots, memory usage also increases due to the increased dimensionality of the system state.

Now we will present detailed results corresponding to single simulation trials at points A, B, and C, beginning first with memory usage for a single robot (robot 1) in Fig. 2.13a–2.13c. Momentary increases in memory usage occur as a robot accumulates information between partial checkpoints and reduction occurs when the Markov property is applied. The grey areas in the these figures indicate the timesteps at which partial checkpoints were detected (i.e., when robot 1 is able to produce a centralized-equivalent estimate and apply the Markov property). The characteristics and the frequency of these fluctuations were observed to be dependent on connectivity settings. Note that at high frequency of connectivity, memory usage approaches that of a centralized estimator. Using these three graphs, we can also get a sense of computation and communication requirements. As

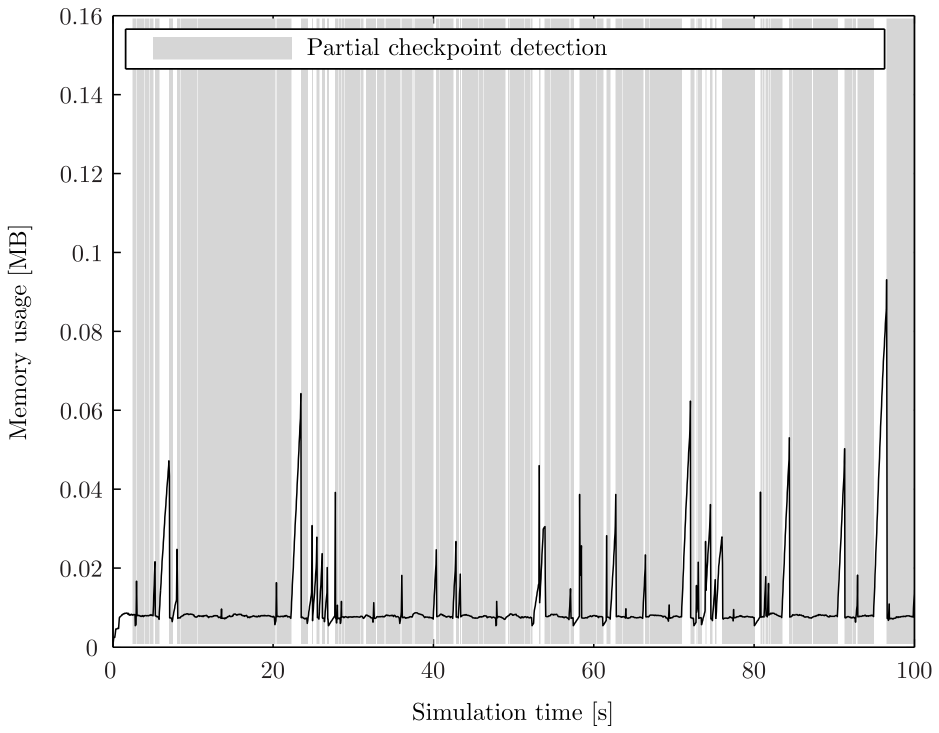


(a) $\pi_1 = 3, \pi_2 = 0.50$ (pt.A).

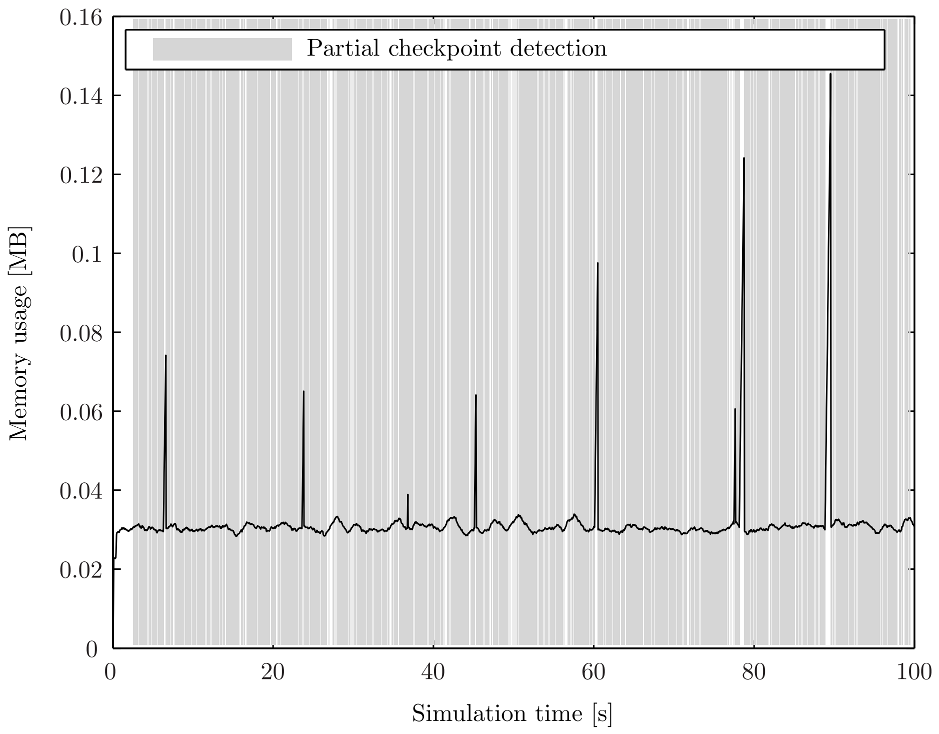
Figure 2.13: Memory usage for robot 1 at various connectivity settings.

memory usage increases, more information is stored in the knowledge set, which needs to be communicated and processed during state estimation. However, note that we rarely experience the worst-case scenario shown in Fig. 2.9, and even in the low-frequency-of-connectivity case, the timesteps between partial checkpoints are limited.

The difference between the overall uncertainty of robot 1's decentralized state estimate and that of the centralized state estimate can be observed in Fig. 2.14a–2.14c. Shown in these plots are the determinant of the estimation error covariance (which is proportional to the volume of the uncertainty ellipsoid) for the centralized state estimator (CSE), our decentralized state estimator (DSE), and decentralized dead-reckoning (DDR), which only uses odometry data. The grey areas again represent when robot 1 detects the existence of a partial checkpoint. As the frequency of connectivity increases, uncertainty for the decentralized state estimate approaches that of the centralized state estimate. Note that because there are no stationary landmarks, and since all robots are always in motion, the error covariance gradually inflates over time. This is most evident with the plots corresponding to the uncertainty growth while dead-reckoning with only odometry data. In cooperative localization, relative measurements between robots help to reduce the rate of increase for the uncertainty, but its continual increase in inevitable.

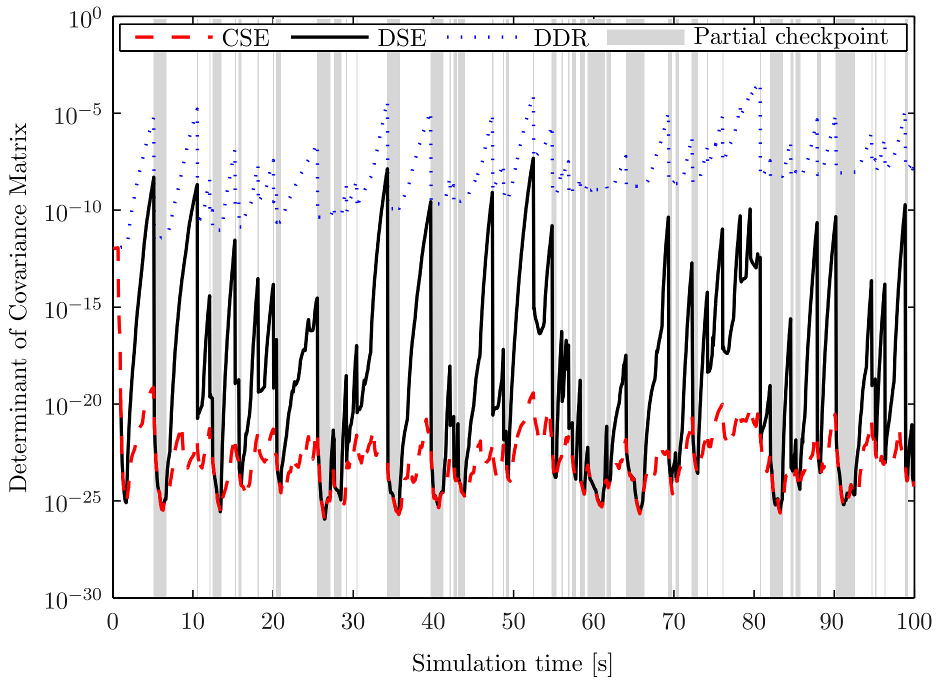


(b) $\pi_1 = 8, \pi_2 = 2.00$ (pt.B).

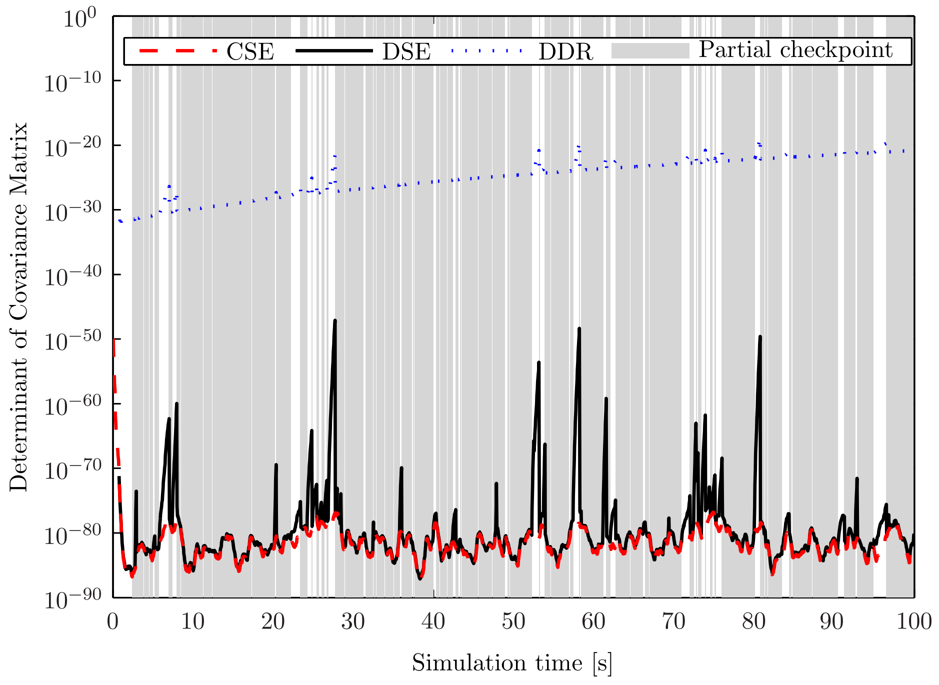


(c) $\pi_1 = 17, \pi_2 = 4.25$ (pt.C).

Figure 2.13: Memory usage for robot 1 at various connectivity settings.

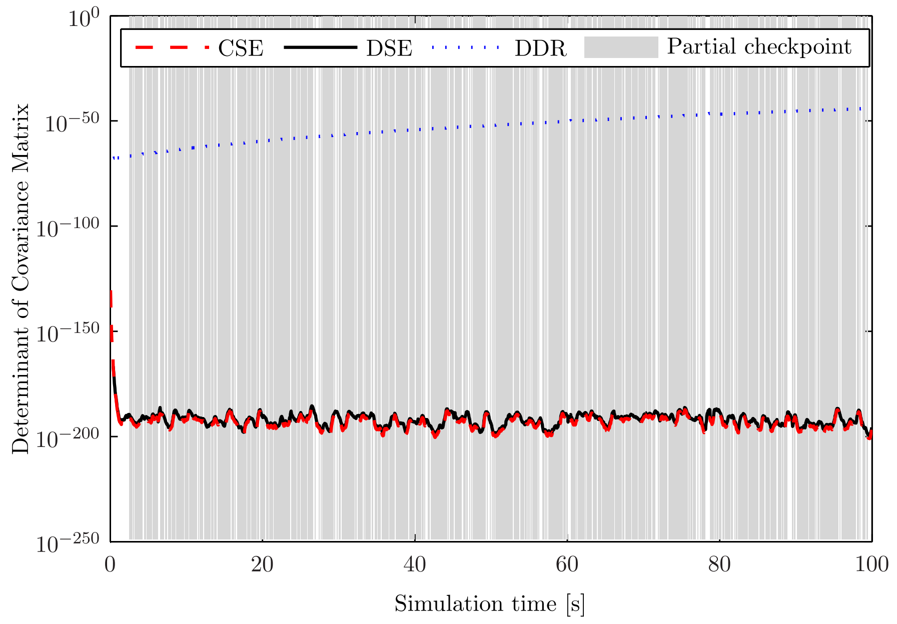


(a) $\pi_1 = 3, \pi_2 = 0.50$ (pt.A).



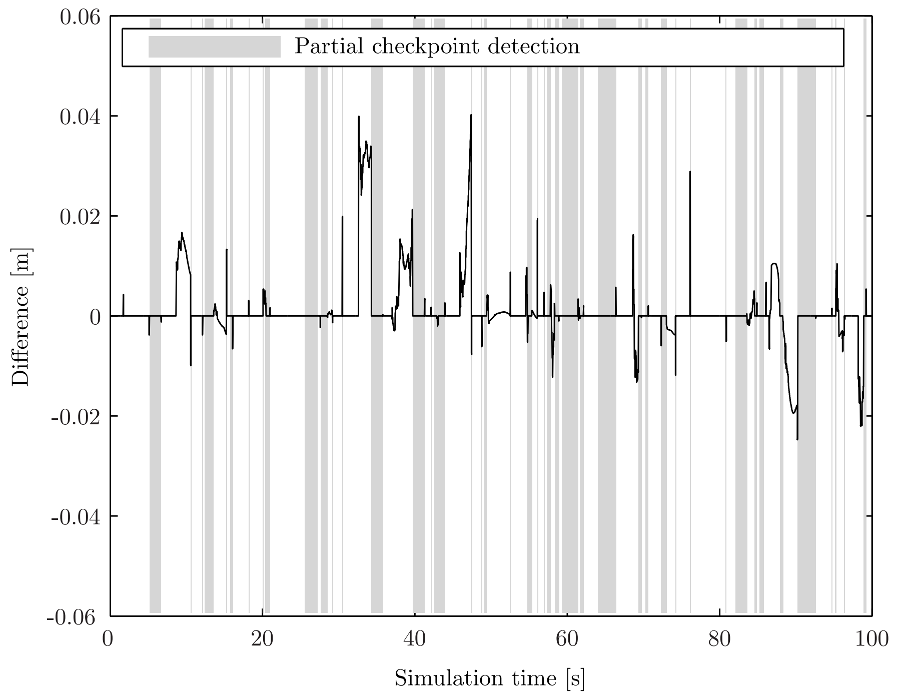
(b) $\pi_1 = 8, \pi_2 = 2.00$ (pt.B).

Figure 2.14: Estimator uncertainty for robot 1 at various connectivity settings.



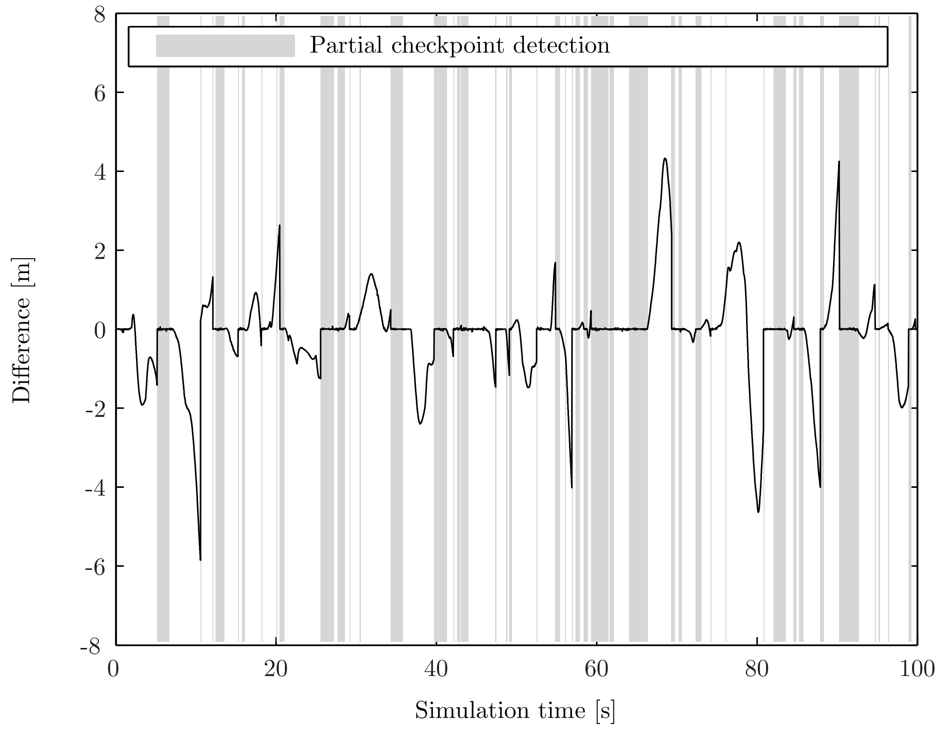
(c) $\pi_1 = 17, \pi_2 = 4.25$ (pt.C). Note that the plots for CSE and DSE are overlapping

Figure 2.14: Estimator uncertainty for robot 1 at various connectivity settings.

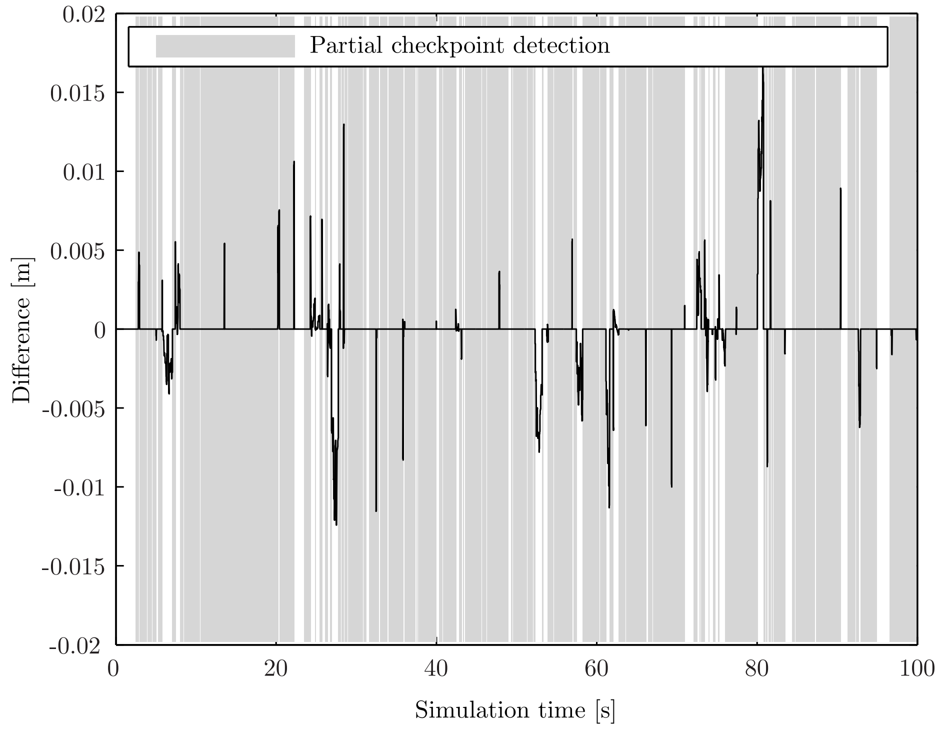


(a) $\pi_1 = 3, \pi_2 = 0.50$ (pt.A).

Figure 2.15: The difference between robot 1's decentralized estimate and the centralized estimate for (a,c,e) its own x-position, and (b,d,f) the x-position of robot 2. Note the difference in scale on the difference.

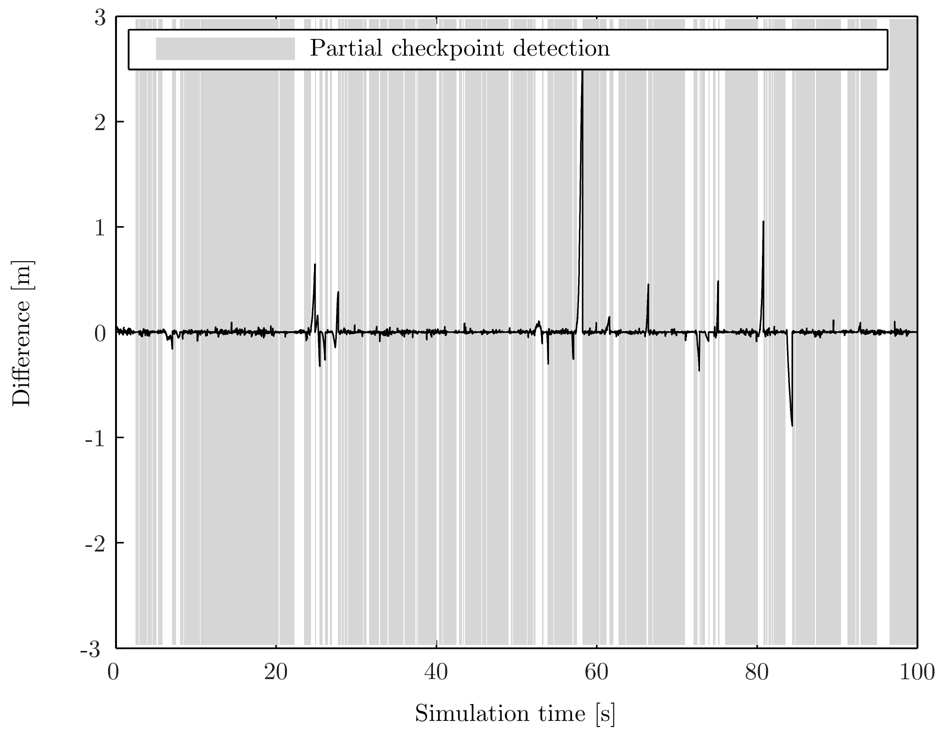


(b) $\pi_1 = 3, \pi_2 = 0.50$ (pt.A).

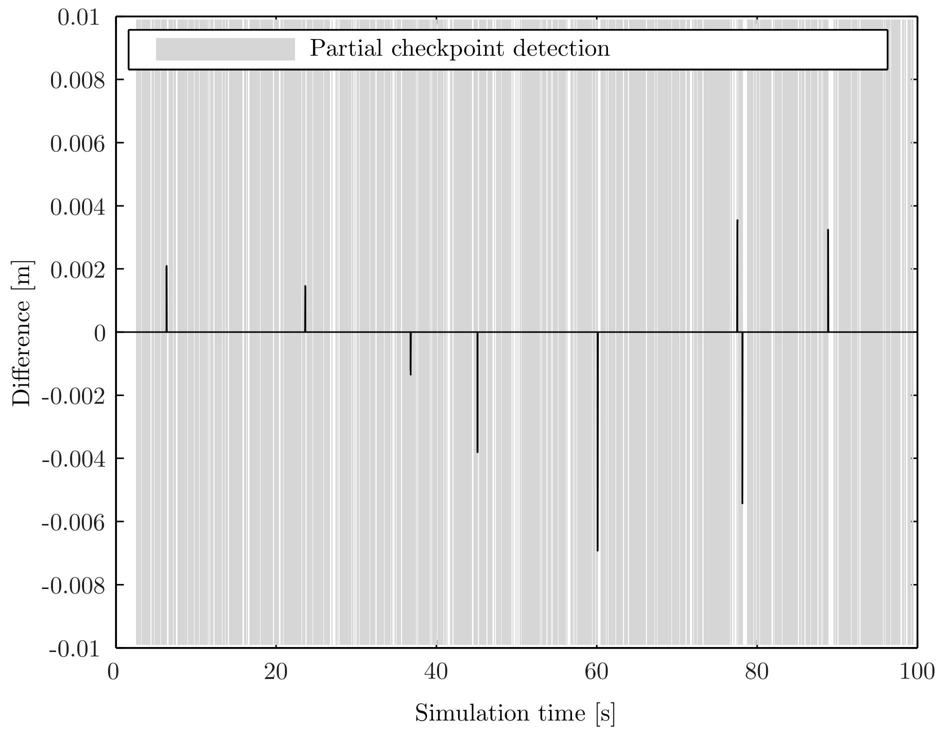


(c) $\pi_1 = 8, \pi_2 = 2.00$ (pt.B).

Figure 2.15: The difference between robot 1's decentralized estimate and the centralized estimate for (a,c,e) its own x-position, and (b,d,f) the x-position of robot 2. Note the difference in scale on the difference.

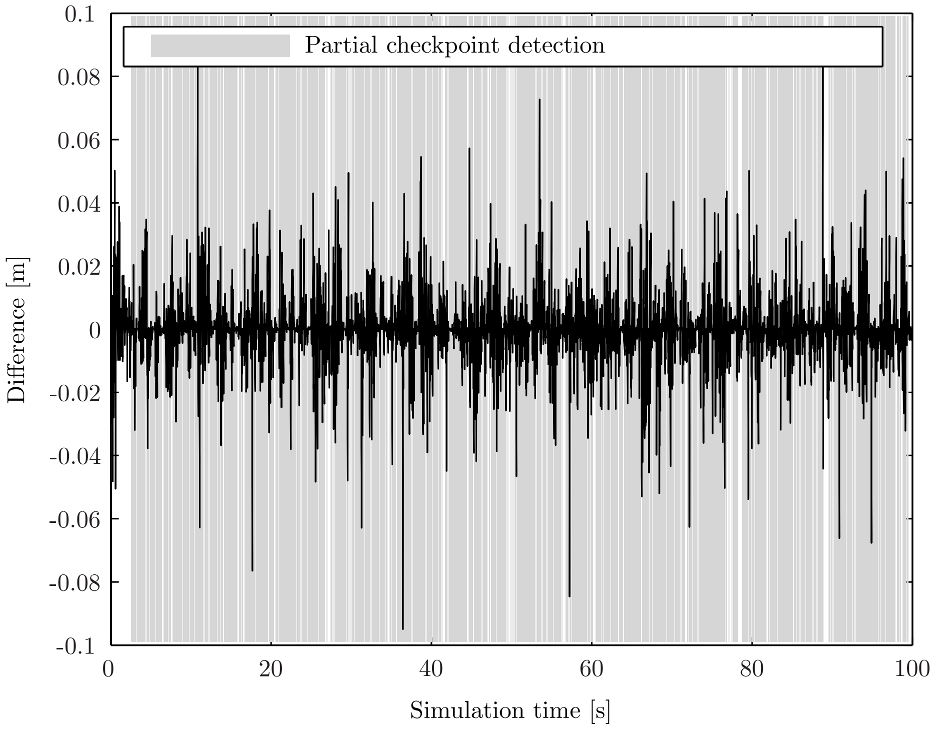


(d) $\pi_1 = 8, \pi_2 = 2.00$ (pt.B).



(e) $\pi_1 = 17, \pi_2 = 4.25$ (pt.C).

Figure 2.15: The difference between robot 1's decentralized estimate and the centralized estimate for (a,c,e) its own x-position, and (b,d,f) the x-position of robot 2. Note the difference in scale on the difference.



(f) $\pi_1 = 17, \pi_2 = 4.25$ (pt.C).

Figure 2.15: The difference between robot 1's decentralized estimate and the centralized estimate for (a,c,e) its own x-position, and (b,d,f) the x-position of robot 2. Note the difference in scale on the difference.

We have also simulated the Cramér-Rao lower bound for uncertainty by evaluating the Jacobians used in the EKF at the true states [87]. This theoretical lower bound for uncertainty is not in the above plots, but we have confirmed that it is slightly lower than that of the centralized state estimator, providing evidence that both the centralized and decentralized estimators that have been implemented are not overconfident.

Next, we will show the difference between the mean of the decentralized state estimates, and that of the centralized state estimates. Since there are a large number of states being estimated (by each robot), we elected to only show a portion of these. For each trial, we will show the difference for robot 1's own x-position estimates (Fig. 2.15a–2.15e) and that of robot 1's estimate of robot 2 (Fig. 2.15b–2.15f). Similar results are observed for all other robots' poses. It is evident that as the frequency of connectivity increases, the difference between the decentralized and centralized state estimates decreases. Furthermore, the difference for a robot's estimate for itself is always closer to the centralized estimate compared to its estimate of another robot. This is because a robot is always aware of its own odometry and measurements but not necessarily that of another robot depending on the evolving network topology.

It can be seen how the detailed results presented above follow the overall trend in localization performance and memory usage observed in Fig. 2.11 and Fig. 2.12. Once again, remember that centralized state estimation is not possible when full connectivity is not guaranteed between robots at each timestep, and the decentralized state estimator presented here is proven to allow the equivalent centralized state estimate to be reached under a dynamic network where robots sporadically communicate. At no time does the network need to be fully connected.

In our performance evaluation, we only compared our centralized-equivalent approach to the centralized EKF estimator. In our review in Section 2.1, we covered some alternative methods that can be applied to robot networks that are never guaranteed to be fully connected. Although we have not implemented these methods to generate quantitative results, we will make a qualitative comparison. One important point to highlight is that our approach is currently the only one that can produce centralized-equivalent estimates in all sparsely-communicating networks, and that do not suffer from cyclic updates and OOSM. As described previously, the approach by Karam et al. [59] made robots maintain estimates of the team by filtering local information. During information exchange, a robot used estimates from other robots as a measurement in performing EKF updates. Cyclic updates were problematic, causing estimates to be over-confident. The approach by Panzieri et al. [60] used multiple EKFs, and each processed a subset of the robot team state. As the correlations between subgroups are ignored, their solution was inconsistent, which they tried to patch by increasing measurement noise. The hierarchical approach by Martinelli [61] also used subgroups and again produced inconsistent estimates. Overall, the results from these alternative methods may produce estimates with lower uncertainty. However, because the estimates are over-confident, they are also sub-optimal compared to our centralized-equivalent approach. The limitations of our approach are the computational, memory storage, and communication complexities. Our approach requires comparatively more storage as we wait for partial checkpoint existence, as well as more computation as we calculate the centralized-equivalent estimate from the previous partial checkpoint occurrence time. These are the costs in obtaining centralized-equivalent estimates in a sparsely-communicating network.

2.6 Localization with a Known Map

Using extensive simulation results we have validated our decentralized state estimation algorithm (Algorithm 1) and its applicability in a decentralized cooperative localization. Recall that we are currently looking at this problem as a simplification from the decentralized SLAM problem by removing the presence of objects in the environment (landmarks) and the need to estimate a map. Before moving on to the decentralized SLAM problem, it is of interest to discuss the problem of cooperative localization with a known map. In this problem, we have a map of where the landmarks are located so there is no need to estimate their positions. Furthermore, we can use the known map to help localize the robots (provided that the robots have the appropriate sensors to make relative measurements to the landmarks). In this case, our decentralized algorithm is still applicable. This is because the idea of information flow, the concept of checkpoints and partial checkpoints, as well as the underlying theory, remain unchanged. The only difference here is the system model (previously shown in (2.1) and (2.2)). In cooperative localization with a known map, we need a measurement model for observing landmarks:

$$\begin{aligned}
\mathbf{x}_{i,k} &= \mathbf{g}\left(\mathbf{x}_{i,k-1}, \mathbf{u}_{i,k}, \epsilon_k\right) \\
\mathbf{y}_{i,k}^{j,i} &= \mathbf{h}\left(\mathbf{x}_{i,k}, \mathbf{x}_{j,k}, \delta_k\right), (\forall j \in N)(d_k^{j,i} \le r_{\mathsf{obs}}) \\
\mathbf{y}_{i,k}^{m,i} &= \mathbf{h_m}\left(\mathbf{x}_{i,k}, \mathbf{x}_{m,k}, \psi_k\right), (\forall m \in M)(d_k^{m,i} \le r_{\mathsf{obs}})
\end{aligned}$$

where:

$\mathbf{h_m}$ is the measurement model

M represents the set of all landmarks (i.e., the map)

$\mathbf{x}_{m,k}$ represents the position of a particular landmark (m)

ψ is the landmark measurement noise

The objective remains as find $\mathsf{bel}(X_k))$ in a decentralized manner. [5]

2.7 Summary

In this chapter, we examined at a simplified version of the decentralized SLAM problem by neglecting the presence of, and the need to estimate a map. This reduced the decentralized SLAM problem into the decentralized localization problem, where robots need to obtain the centralized-equivalent estimate of all robot poses in a decentralized manner.

We began by providing a mathematical formulation of the problem, defined knowledge sets, and their update rules based on communication between robots. We defined the key

[5]Note that localization with a known map is a task that could be accomplished using the same approach for decentralized cooperative localization, but this work has not been carried out.

concept of a checkpoint, allowing all robots to obtain a centralized-equivalent estimate and apply the Markov property (an assumption that future estimates are only dependent on the current estimate, which allows us to discard past information). We also developed theorems regarding its existence. However, checkpoints are impractical in the sense that in order to detect their existence, an outside observer is required. This led to us looking at a network from the perspective of a single robot. We introduced the concept of a partial checkpoint, which indicates when a single robot can obtain a centralized-equivalent estimate of all robot poses. We also developed several theorems for detecting the existence of a partial checkpoint.

In relating partial checkpoints to checkpoints, we were able to show that a robot's decision to invoke the Markov property does not affect another robot's ability to do the same. Furthermore, if all robots invoke the Markov property whenever possible (when a partial checkpoint exists) based on their own knowledge, we can be sure that all robots can still obtain a centralized-equivalent estimate.

Using the theorems we presented, an algorithm was developed allowing each robot to obtain the centralized-equivalent estimate whenever possible. At every timestep, the algorithm produces a current estimate (which may be temporary), and a centralized-equivalent estimate if possible. Note that the time of the centralized equivalent estimate may not be the current time (i.e., there may be a delay) due to network connectivity. The algorithm is scalable in the sense that the number of robots in the network does not need to be known initially, but only when communication is bi-directional, and when the communication range limit is greater than the measurement range limit.

We validated our decentralized algorithm through simulations. We showed that memory usage (although large compared to the centralized estimator) is limited by exploiting the Markov property at partial checkpoints. In our simulations, we also looked at how various factors (captured by dimensionless variables) affect performance of our decentralized state estimator in comparison to the centralized estimator, and how this relates to network connectivity. Results show that the performance of our decentralized state estimator begins to approach that of the centralized state estimator when a phase transition occurs in the frequency of network connectivity. It must be stressed again that a centralized state estimator will not be able to produce an estimate unless we assume full network connectivity at all times.

In the upcoming chapter, we will look at the decentralized SLAM problem without simplifications. Many of the key theorems that we developed in this chapter will be carried forward and applied to the decentralized SLAM problem.

Chapter 3

Decentralized Cooperative Simultaneous Localization and Mapping

Teamwork is the ability to work together toward a common vision. The ability to direct individual accomplishments toward organizational objectives. It is the fuel that allows common people to attain uncommon results.

Andrew Carnegie (1835–1919)
industrialist, businessman, and entrepreneur.

In the previous chapter, we examined a simplified version of the decentralized cooperative SLAM problem, where we removed the requirement for robots to estimate the map (position of landmarks). This simplification transformed the decentralized cooperative SLAM problem into the decentralized cooperative localization problem. It was done so that we could focus our attention on the consequence of sporadic network connectivity on information flow. In this chapter[1], we will restore the requirement for robots to estimate a map. That is, each robot must estimate the poses (i.e., position and orientation) of all robots, as well as the position of all observed landmarks. We will first provide the mathematical formulation of the decentralized cooperative SLAM problem. Then under the assumption that the communication network is never guaranteed to be fully-connected,

[1]The work this chapter was presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems 2010 [88]. It was also accepted for publication in the Journal of Intelligent Robotic Systems [89].

we will define the conditions under which the centralized-equivalent estimate (for all robot poses and landmark positions) can be obtained. We will see how these conditions relate to checkpoints and partial checkpoints, which enables us to apply the theorems developed from the decentralized cooperative localization problem to the decentralized cooperative SLAM problem. To validate our approach to decentralized cooperative SLAM, we will go beyond using simulations, and use a hardware experiment dataset collected using a fleet of real robots.

3.1 An Overview of Advances in Cooperative SLAM

One of the earliest works on cooperative SLAM was by Fenwick et al. [90], who generalized the single-robot EKF SLAM approach for the multi-robot scenario. This is essentially a centralized approach, which requires a fully-connected communication network. Using the information form of the EKF, Nettleton et al. [91] devised a decentralized algorithm that takes advantage of the additive update property, which allows the incorporation of observations into an estimate in any order. However, as pointed out by Rosencrantz et al. [92], this only works for states that do not change over time and where motion predictions are not necessary (i.e., only the static map is estimated). Nettleton et al. [93] acknowledges this, because when we estimate the states of robots and landmarks, motion predictions affect the estimate uncertainty for the entire system. Later, Nettleton et al. [93] also presented a SLAM algorithm where robots only communicate sub-map information to reduce communication bandwidth. However, this no longer produces centralized-equivalent estimates. Furthermore, as Reece and Roberts [94] point out, the use of the *Covariance Intersection* method as part the algorithm in [93] yields highly conservative estimates. In addition, the algorithm can only handle Gaussian noise as it is fundamentally based on the *Extended Information Filter*.

Thrun et al. [95] introduced the *Sparse Extended Information Filter* (SEIF) for SLAM, which is an approximation of the centralized-equivalent estimate. This was extended to the multi-robot SLAM problem, where the robots are not aware of each other's starting pose, and overlapping local maps from each robot are combined to form the global map. Similarly, Ko et al. [96], Zhou and Roumeliotis [97] and Wang et al. [98] also came up with approaches that involve merging of maps between robots. In the approach proposed in [96], a robot tries to localize itself in another robot's map to confirm their relative positions before merging their maps. In [97], relative measurements between robots, as well as landmarks in two robots' maps are used to align and merge maps. Note that although the approaches described above do not require a fully-connected network at all times, they do not produce centralized-equivalent estimates, which is something that we try to achieve.

Howard et al. [99] presented a novel approach to multi-robot SLAM that uses manifold maps instead of a planar map for a two-dimensional environment. In their approach, each robot maintains its own manifold map until it encounters another robot, at which point their maps are merged. The focus of this work is mainly on the mapping aspect, and a robot only needs to localize itself (as opposed to localizing all robots in the system). Howard [100] also looked at performing multi-robot SLAM using particle filters. In this approach, robots are unaware of each other's initial pose and first perform state estimation in a decentralized manner (i.e., each robot acts as an independent system). When robots encounter each other for the first time, their individual maps are combined into a common map using relative pose information. The notion of a virtual robot travelling backward in time is introduced to allow for the incorporation of information gathered by a robot before the common map is merged. When all robots have encountered each other, the state estimation process becomes centralized, and requires a fully-connected communication network.

3.2 Mathematical Formulation

Before we show the mathematical formulation, it is important to note that there exist different variations of the cooperative SLAM problem. One variation only requires robots to localize themselves while estimating a map (such as [96] for example). However our formulation requires each robot to obtain estimates for the position of all known landmarks, as well as the poses of all robots in the system.

The mathematical formulation for decentralized cooperative SLAM is similar to that of decentralized cooperative localization, with the addition of landmarks. As in cooperative localization, let N represent the set containing the unique identification indices of all robots. In addition to this, let M represent the set that contains the unique identification indices of all landmarks (the map). The system model is as follows:

$$\begin{aligned}
\mathbf{x}_{i,k} &= \mathbf{g}\left(\mathbf{x}_{i,k-1}, \mathbf{u}_{i,k}, \epsilon_k\right), \ \forall i \in N \\
\mathbf{x}_{m,k} &= \mathbf{x}_{m,k}, \ \forall m \in M \\
\mathbf{y}_{i,k}^{j,i} &= \mathbf{h}\left(\mathbf{x}_{i,k}, \mathbf{x}_{j,k}, \delta_k\right), \ \forall j \in N, \ d_k^{j,i} \leq r_{\mathsf{obs}} \\
\mathbf{y}_{i,k}^{m,i} &= \mathbf{h}_{\mathbf{m}}\left(\mathbf{x}_{i,k}, \mathbf{x}_{m,k}, \psi_k\right), \ \forall m \in M, \ d_k^{m,i} \leq r_{\mathsf{obs}}
\end{aligned}$$

Note that we are still using a general system model (which may be nonlinear), where for timestep k:

$\mathbf{x}_{i,k}$ $(i \in N)$ represents the state (pose) of robot i
$\mathbf{x}_{m,k}$ $(m \in M)$ represents the state (position) of the stationary landmark m
$\mathbf{u}_{i,k}$ represents the odometry information of robot i
$\mathbf{g}(\cdot)$ is the state transition function for the robots
ϵ_k represents the process noise
$\mathbf{y}_{i,k}^{j,i}$ represents the measurement (e.g., range/bearing) of robot j with respect to robot i
$\mathbf{y}_{i,k}^{m,i}$ represents the measurement of landmark m with respect to robot i
$\mathbf{h}(\cdot)$ is the robot measurement function
$\mathbf{h_m}(\cdot)$ is the landmark measurement function
δ_k is the robot measurement noise
ψ_k is the landmark measurement noise
$d_k^{j,i}$ is the distance between robot i and robot j
$d_k^{m,i}$ is the distance between robot i and landmark m
r_{obs} is the measurement range limit

With the inclusion of landmarks, we need to redefine some of the sets introduced in Chapter 2. Let

$$X_k = \{\mathbf{x}_{i,k}, \mathbf{x}_{m,k} | i \in N, m \in M_k\},$$

represent the set of all states known to exist at timestep k, where M_k is the set of landmarks that have been observed by at least one robot up to time k. Let

$$X_{i,k} = \{\mathbf{x}_{j,k}, \mathbf{x}_{m,k} | j \in N_{i,k}, m \in M_{i,k}\},$$

represent the states of all robots and landmarks known to robot i up to time k, where $N_{i,k}$ is the set of robots known to robot i up to time k, and $M_{i,k}$ is the set of landmarks known to robot i up to time k. Let

$$Y_{i,k} = \{\mathbf{y}_{i,k}^{j,i}, \mathbf{y}_{i,k}^{m,i} | j \in N, m \in M, d_k^{j,i} \leq r_{\mathsf{obs}}, d_k^{m,i} \leq r_{\mathsf{obs}}\}$$

represent the set of measurements from robot i to all robots and landmarks within observation range.

In the previous chapter, when we examined the decentralized cooperative localization problem, we assumed that $R_{i,k}$ represents the set of robots that robot i is connected with at timestep k (i.e., as in Fig. 2.1a). For the decentralized cooperative SLAM problem, we can assume either of the information exchange modes apply. That is, we can also define $R_{i,k}$ as the set of robots within communication range of robot i as in Fig. 2.1b:

$$R_{i,k} = \{j | j \in N, d_k^{i,j} \leq r_{\mathsf{comm}}\}$$

We will continue to adhere to the probabilistic approach as we are dealing with a stochastic system. The expression for the centralized belief (which is a function that describes the joint likelihood of the states in X_k),

$$\mathsf{bel}(X_k) := p\left(X_k | \mathsf{bel}(X_0), \{\mathbf{u}_{i,1:k}, Y_{i,1:k} | i \in N\}\right),$$

remains the same as in the cooperative localization case, but note here that the belief is now over the state of all robots and known landmarks.

To keep track of what each robot knows, we will make use of knowledge sets again, with the update rules defined in (2.3) and (2.4). However, we will assume at the initial timestep that each robot knows the total number of robots in the team. This is a subtle but important difference between decentralized cooperative SLAM and decentralized cooperative localization, where a robot only needs to know of its own existence, and can look for partial checkpoints based on the robots known to it (provided that $r_{\mathsf{comm}} \geq r_{\mathsf{obs}}$). We will show later in this chapter that this initial knowledge requirement is necessary to guarantee that the centralized-equivalent estimate is obtainable by all robots in the decentralized cooperative SLAM problem. In addition to knowing the total number of robots in the team, each robot also begins with an estimate of its initial pose, $S_{i,0} \supseteq \mathsf{bel}(\mathbf{x}_{i,0})$. These pose estimates are all expressed within the same reference frame.

In performing state estimation for single-robot SLAM, the system of robot and landmarks is unobservable unless the initial (absolute) pose of the robot with respect to the system's reference frame can be determined [101]. This can be accomplished by using some prior information. The system can be made observable by directly providing the estimator with the initial pose of the robot. Alternatively, by knowing the absolute positions for a number of landmarks (2 in the case of planar two-dimensional SLAM) and making observations to these landmarks, the system also becomes observable. In the case where prior information is not available, we can perform relative SLAM, such that the initial robot pose is used as the reference frame for the system. Although the system is not observable, we still seek a SLAM solution, which we must interpret carefully given the lack of prior knowledge. We can extend the above discussion to multi-robot SLAM. One way of ensuring that the system of multiple robots and landmarks is observable is to provide the estimator with the initial poses of all the robots. The availability of these initial pose estimates is an assumption that we make in our centralized-equivalent approach to make our system observable.

Once again, we would like to make use of the *Markov property*,

$$p\left(X_k|\mathsf{bel}(X_0), U_{1:k}, Y_{1:k}\right) = p\left(X_k|\mathsf{bel}(X_{k-1}), U_k, Y_k\right),$$

to reduce computational requirements and memory use by allowing state estimation to be performed recursively. The set $Y_{1:k}$ now includes both inter-robot measurements and measurements made to landmarks. We saw from Theorem 3.1 that in decentralized cooperative localization, robots can invoke the Markov property based on their local knowledge without consideration of other robots' knowledge. Furthermore, we ensured that all other robots can perform the same. We will show that this is also the case for decentralized cooperative SLAM.

Our objective for decentralized cooperative SLAM is for each robot to find $\mathsf{bel}(X_{i,k})$ in a decentralized manner, given that the robots are only occasionally exchanging information with one another in a sparsely connected and dynamic network.

3.3 Obtaining the Centralized-Equivalent Estimate

It is important not to forget why we are interested in obtaining the centralized-equivalent estimate. This is because operating in a sparsely-communicating network in a decentralized manner introduces the possibility of cyclic updates and out-of-sequence measurements. These in turn may lead to inconsistent estimates. It must be acknowledged that the rules that we use for updating knowledge sets help by preventing the duplication of information in a knowledge set, but we still need to ensure that the same piece of information is not used more than once in calculating an estimate. By adopting a centralized-equivalent approach, we can always avoid these problems. Most important is the fact that obtaining a centralized-equivalent estimate allows us to make use of the Markov property to limit memory usage.

In Chapter 2, we introduced the concepts of a checkpoint and a partial checkpoint. These events indicate when a team of robots, or when an individual robot, is able to obtain the centralized-equivalent estimate and apply the Markov Property in the decentralized cooperative localization problem. We have also put forward a number of theorems regarding the existence of checkpoints and partial checkpoints, which ultimately allowed us to conclude that all robots should calculate the centralized-equivalent estimate and apply the Markov property based on local knowledge. Although this action appears greedy, all robots are still guaranteed to be able to obtain the centralized-equivalent estimate (provided that the knowledge set update rules are followed when robots communicate). In this chapter, we would like to extend the concepts of checkpoint and partial checkpoint to the decentralized cooperative SLAM problem. One option is for us to prove all the theorems again for the decentralized cooperative SLAM case, but this will be largely a repetition of the previous proofs. The reason for this is because landmarks are passive objects in the workspace that do not produce any measurements, and are incapable of relaying information. Instead, we will claim that the theorems from Chapter 2 also apply to the decentralized cooperative SLAM problem, and explain why this is the case.

3.3.1 The Global Perspective

We will begin as in Chapter 2, where we first examined the robot network as an outside observer of the system that is able to see when each robot makes a measurement (to a landmark or another robot). The outside observer is also able to see when communications occur between robots. First, we want to show that the concept of a checkpoint can be applied to the decentralized cooperative SLAM problem. That is, when a checkpoint occurs, all robots are able to obtain the centralized-equivalent estimate in decentralized cooperative SLAM.

Suppose that all robots can calculate the centralized-equivalent estimate in decentralized cooperative SLAM at some timestep, k_c. This implies that robots can obtain

$$\mathsf{bel}(X_{k_c}) = p\left(X_{k_c}\middle| \{\mathsf{bel}(X_{j,0}), \mathbf{u}_{j,1:k_c}, Y_{j,1:k_c} | j \in N\}\right),$$

or more generally, robots can obtain

$$\begin{cases} p\left(X_{k_c}\middle| \left\{\mathsf{bel}(X_{j,k_{s,j}}), \mathbf{u}_{j,k_{s,j}+1:k_c}, Y_{j,k_{s,j}+1:k_c} | j \in N\right\}\right), \text{ if } k_{s,j} < k_c \\ p\left(X_{k_c}\middle| \left\{\mathsf{bel}(X_{j,k_{s,j}}) | j \in N\right\}\right), \text{ if } k_{s,j} = k_c. \end{cases}.$$

If we examine the dependencies (estimates, odometry, and measurement data) of the above distribution, we can conclude that these dependencies must be in each robot's knowledge set. That is,

$$\begin{aligned} S_{i,k_e} &\supseteq \begin{cases} \{\mathsf{bel}(X_{j,k_{s,j}}), \mathbf{u}_{j,k_{s,j}+1:k_c}, Y_{j,k_{s,j}+1:k_c} | j \in N\}, \text{ if } k_{s,j} < k_c \\ \{\mathsf{bel}(X_{j,k_{s,j}} | j \in N)\}, \text{ if } k_{s,j} = k_c \end{cases} \\ &\supseteq \begin{cases} \{\mathsf{bel}(X_{j,k_{s,j}}), \mathbf{u}_{j,k_{s,j}+1:k_c}, Y_{j,k_{s,j}+1:k_c} | j \in N\}, \text{ if } k_{s,j} < k_c \\ \{\mathsf{bel}(X_{k_c}) | j \in N\}, \text{ if } k_{s,j} = k_c \end{cases}. \end{aligned}$$

The definition of a checkpoint (Definition 4 from Chapter 2) states that:

A *checkpoint*, $C(k_c, k_e)$, is an event that occurs at the checkpoint time, k_c, that first comes into existence at k_e, in which the knowledge set for each robot i contains for all j:

1. the previous state estimate of robot j at some timestep, $k_{s,j} \leq k_c$,
2. all the odometry and measurement data of robot j from timestep $k_{s,j}$ to k_c.

Note that this is the exact same information that is required to obtain the centralized-equivalent estimate in decentralized cooperative SLAM. Therefore, we can conclude that the existence of a checkpoint allows all robots to obtain the centralized-equivalent estimate in decentralized cooperative SLAM. More formally, when $C(k_c, k_e)$ exists, all robots can obtain $\mathsf{bel}(X_{k_c})$. Recall that the difference between the checkpoint existence time, k_e, and checkpoint occurrence time, k_c, can be interpreted as the time delay in obtaining the centralized-equivalent estimate. This is a consequence of robots having to operate in a sparsely-communicating network that is never guaranteed to be fully connected. Next we need to show that the theorems related to checkpoints from Chapter 2 also apply to the decentralized cooperative SLAM case. We will not prove these theorems again for the decentralized cooperative SLAM case since the proofs will look almost identical to the decentralized cooperative localization case. Rather, we will claim that these theorems apply to the decentralized cooperative SLAM problem and provide arguments as to why this is the case.

Claim 4.1: *Theorem 1.1, Lemma 1.1, and Theorem 1.2 apply to the decentralized cooperative SLAM problem.*

Reasoning. Theorem 1.2 states that checkpoint $C(k_c, k_e)$ exists if and only if a path exists from node $v(i, k_c)$ to node $v(j, k_e)$ on the information flow graph $\mathcal{G}_{k_c,k_e}$, $(\forall i, j)$. Recall that an information flow graph represents the path in which information is carried forward in time and exchanged between robots[2]. Since landmarks are passive objects and do not communicate or relay information, their presence does not alter the information flow graph in any way. Therefore, this theorem applies in both the decentralized cooperative localization and the decentralized cooperative SLAM scenarios.

Lemma 1.1 is concerned with the order of the occurrence times and existence times for two checkpoints. It states that if $C(k_{c_1}, k_{e_1})$ and $C(k_{c_2}, k_{e_2})$ exist, and $k_{e_1} \neq k_{e_2}$. Then $k_{c_1} < k_{c_2}$ if and only if $k_{e_1} < k_{e_2}$. The argument in the proof of this lemma uses the information flow graph. Again, since the graph topology does not change between the

[2]In some of the figures presented so far, we explicitly drew out other events such as measurements between robots, but these are only for explanation purposes and play no part in information flow.

decentralized cooperative localization and the decentralized cooperative SLAM scenarios, this lemma applies to the decentralized cooperative SLAM problem.

Theorem 1.2 provides a practical method to detect the existence of a checkpoint. It states that the checkpoint $C(k_c, k_e)$ exists if and only if the knowledge set of each robot at k_e contains $\mathbf{u}_{j,k_c}$ or $\mathsf{bel}(\mathbf{x}_{j,k_c}), (\forall j)$. Hence, we only need to search for a subset of the information required to obtain the centralized-equivalent estimate to determine the existence of a checkpoint. As we have already shown, these dependencies do not change from the decentralized cooperative localization to the decentralized cooperative SLAM case (i.e., we still need a prior estimate with the measurements and odometry data from all robots up to the time of the new centralized-equivalent estimate). Therefore, this theorem is also applicable to the decentralized cooperative SLAM problem. □

In summary, what we have shown so far is that the concept of a checkpoint applies not only to the decentralized cooperative localization problem, but also to decentralized cooperative SLAM. This implies that an outside observer can determine if all robots can obtain the centralized-equivalent estimate in cooperative decentralized SLAM by checking for the existence of a checkpoint. In other words, we have generalized the notion of a checkpoint. We have also reasoned that Theorems 1.2, Lemma 1.1, and Theorem 1.2 from Chapter 2 also apply to the decentralized cooperative SLAM problem, which provides us with a way to detect the existence of a checkpoint. However, we have seen before that the need for a outside observer defeats the purpose of having a decentralized system. Therefore, we need to again examine if we can eliminate the need of the outside observer to obtain centralized-equivalent estimates.

3.3.2 The Local Perspective

Once again, we have to acknowledge the fact that a robot's perception of the network topology may not be as up to date as that of an outside observer. This was illustrated in Fig 2.4 in Chapter 2, and we introduced the concept of a partial checkpoint to indicate when a single robot can obtain the centralized-equivalent estimate for all robots in decentralized cooperative localization. We want to show that partial checkpoints can also be used in decentralized cooperative SLAM.

The following is very similar to the argument we used to show that checkpoints are applicable in decentralized cooperative SLAM. Suppose that robot i can calculate the centralized-equivalent estimate in decentralized cooperative SLAM at some timestep, $k_{c,i}$. This implies that robot i can obtain

$$\mathsf{bel}\left(X_{k_{c,i}}\right) = p\left(X_{k_{c,i}} \middle| \left\{\mathsf{bel}\left(X_{j,0}\right), \mathbf{u}_{j,1:k_{c,i}}, Y_{j,1:k_c} | j \in N\right\}\right)$$

or more generally, robot i can obtain

$$\begin{cases} p\left(X_{k_{c,i}} \middle| \left\{\mathsf{bel}\left(X_{j,k_{s,j}}\right), \mathbf{u}_{j,k_{s,j}+1:k_{c,i}}, Y_{j,k_{s,j}+1:k_{c,i}} | j \in N\right\}\right), \text{ if } k_{s,j} < k_{c,i} \\ p\left(X_{k_{c,i}} \middle| \left\{\mathsf{bel}\left(X_{j,k_{s,j}}\right) | j \in N\right\}\right), \text{ if } k_{s,j} = k_{c,i}. \end{cases}.$$

If we examine the dependencies (estimates, odometry, and measurement data) of the above distribution, we can conclude that these dependencies must be in robot i's knowledge set. That is,

$$\begin{aligned} S_{i,k_{e,i}} &\supseteq \begin{cases} \left\{\mathsf{bel}\left(X_{j,k_{s,j}}\right), \mathbf{u}_{j,k_{s,j}+1:k_{c,i}}, Y_{j,k_{s,j}+1:k_{c,i}} | j \in N\right\}, \text{ if } k_{s,j} < k_{c,i} \\ \left\{\mathsf{bel}\left(X_{j,k_{s,j}}\right) | j \in N\right\}, \text{ if } k_{s,j} = k_{c,i} \end{cases} \\ &\supseteq \begin{cases} \left\{\mathsf{bel}\left(X_{j,k_{s,j}}\right), \mathbf{u}_{j,k_{s,j}+1:k_{c,i}}, Y_{j,k_{s,j}+1:k_{c,i}} | j \in N\right\}, \text{ if } k_{s,j} < k_{c,i} \\ \left\{\mathsf{bel}\left(X_{j,k_c}\right) | j \in N\right\}, \text{ if } k_{s,j} = k_c \end{cases}. \end{aligned}$$

The definition of a partial checkpoint (Definition 5 from Chapter 2) states that:

A partial checkpoint, $C_p(k_{c,i}, k_{e,i})$, is an event that occurs for robot i at time $k_{c,i}$, that first comes into existence at $k_{e,i}$, in which the knowledge set for robot i contains for all j:

1. the previous state estimate of robot j at some timestep, $k_{s,j} \leq k_{c,i}$,
2. all the odometry and measurement data of robot j from timestep $k_{s,j}$ to $k_{c,i}$.

This is the exact same information that is required for a single robot, i, to obtain the centralized-equivalent estimate in decentralized cooperative SLAM. Therefore, we can conclude that the existence of a partial checkpoint allows a single robot to obtain the centralized-equivalent estimate in decentralized cooperative SLAM. In other words, when $C(k_{c,i}, k_{e,i})$ exists for robot i, then robot i can obtain $\mathsf{bel}(X_{k_c})$. Next we will show that the theorems related to partial checkpoints from Chapter 2 also apply to the decentralized cooperative SLAM case. Following our previous approach, we will not prove these theorems for the decentralized cooperative SLAM case since the proofs will look almost identical to the decentralized cooperative localization case. Instead, we will claim that these theorems apply to the decentralized cooperative SLAM problem and provide arguments as to why this is the case.

Claim 5.2: *Theorem 2.1, Lemma 2.1, and Theorem 2.2 apply to the decentralized cooperative SLAM problem.*

Reasoning. Theorem 2.1 states that partial checkpoint $C(k_{c,i}, k_{e,i})$ exists if and only if a path exists from node $v(j, k_{c,i})$ to node $v(i, k_{e,i})$ on the information flow graph $\mathcal{G}_{k_{c,i},k_{e,i}}, (\forall j \in N)$. Since landmarks are passive objects and do not communicate or relay information, their presence does not alter the information flow graph in any way. Therefore, this theorem applies in both the decentralized cooperative localization and the decentralized cooperative SLAM scenarios.

Lemma 2.1 examines the order of the occurrence times and existence times for two partial checkpoints. It states that if $C_p(k_{c_1,i}, k_{e_1,i})$ and $C_p(k_{c_2,i}, k_{e_2,i})$ exist, and $k_{e_1,i} \neq k_{e_2,i}$. Then $k_{c_1,i} < k_{c_2,i}$ if and only if $k_{e_1,i} < k_{e_2,i}$. The argument in the proof of this lemma also uses the information flow graph. Again, since the graph topology does not change between the decentralized cooperative localization and the decentralized cooperative SLAM scenarios, this lemma applies to the decentralized cooperative SLAM problem.

Theorem 2.2 provides a practical method of detecting the existence of a partial checkpoint. It states that the partial checkpoint $C_p(k_{c,i}, k_{e,i})$ exists for robot i if and only if the knowledge set of robot i at k_e contains $\mathbf{u}_{j,k_c}$ or $\mathsf{bel}(X_{j,k_c}), (\forall j)$. Hence, we only need to search for a subset of the information required to obtain the centralized-equivalent estimate to determine the existence of a checkpoint. As we have already shown, these dependencies do not change from the decentralized cooperative localization to the decentralized cooperative SLAM case (i.e., we still need a prior estimate with the measurements and odometry data from all robots up to the time of the new centralized-equivalent estimate). Therefore, this theorem is also applicable to the decentralized cooperative SLAM problem. □

In summary, the concept of a partial checkpoint can be generalized and applies to both the decentralized cooperative localization problem and the decentralized cooperative SLAM. Any robot can obtain the centralized-equivalent estimate in decentralized cooperative SLAM by checking for the existence of a partial checkpoint. The partial checkpoint detection methods also generalizes to cover the decentralized cooperative SLAM case. As we have seen in Chapter 2, robots can apply the Markov property after obtaining a centralized-equivalent estimate based on local knowledge while ensuring that all other robots can also obtain the centralized-equivalent estimate. This was proven by Lemma 3.1, Theorem 3.1, and Theorem 3.2, and we want to show that these also apply in decentralized cooperative SLAM.

Claim 5.3: *Lemma 3.1, Theorem 3.1, and Theorem 3.2 apply to the decentralized cooperative SLAM problem.*

Reasoning. Lemma 3.1 tells us that when partial checkpoints exist with the same occurrence time for all robots, then a checkpoint exists at the same occurrence time. The proof of this theorem compares the knowledge sets of robots when a checkpoint exists and when partial checkpoints exist for individual robots. Since we have established that checkpoints and partial checkpoints can be generalized (without changes) from the decentralized cooperative localization scenario to cover the decentralized cooperative SLAM problem, this Lemma applies to the decentralized cooperative SLAM problem as well.

Theorem 3.1 is the key theorem that allows our estimation framework to be decentralized. It states that a robot's decision to invoke the Markov property and discard information (that it has used to generate the centralized-equivalent estimate) does not affect the ability of other robots to perform the same (i.e., it does not affect the existence of checkpoints). The proof of this theorem looks at the knowledge sets of robots that communicate with a robot that has applied the Markov property. These knowledge sets are compared to the knowledge sets of robots that do not communicate with the robot the has applied the Markov property. Although the content of these knowledge sets are different, we can still conclude that partial checkpoints exist for all robots (i.e., all the robots can still obtain the centralized-equivalent estimate) by using the definition of a partial checkpoint. The intuition behind this discovery is that robots can communicate their centralized-equivalent estimates to each other instead of passing measurement and odometry data. Since we have not changed the definition of checkpoints or partial checkpoints, we can conclude that this theorem is applicable to decentralized cooperative SLAM.

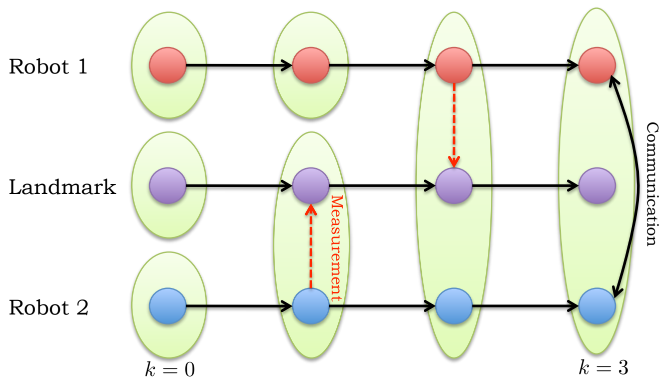
Finally, Theorem 3.2 states that all robots should apply the Markov property whenever a partial checkpoint is detected (i.e., based on their own local knowledge), and this will allow all robots to obtain centralized-equivalent estimates. This theorem is only an extended interpretation of Theorem 3.1, and therefore, also applies to decentralized cooperative SLAM. $\square$

To reiterate, a robot can detect whether it can obtain a centralized-equivalent estimate in decentralized cooperative SLAM by checking for the existence of a partial checkpoint, which we have generalized from the decentralized cooperative localization problem. Because of this, we can apply Theorems 3.1 and 3.2 to the decentralized cooperative SLAM problem. The important implication here is that for both the decentralized cooperative localization and decentralized cooperative SLAM problems, a robot can apply the Markov property and discard information that it no longer needs without considering what other robots have in their knowledge sets. In other words, a robot does not need to keep track of the estimates and measurements known to another robot. Yet, we can still guarantee that all robots can obtain the centralized-equivalent estimate. The result of this is that this makes the decentralized cooperative SLAM algorithm similar to the decentralized cooperative localization algorithm. However we need to consider the necessary initial knowledge of each robot, which differ from decentralized cooperative localization.

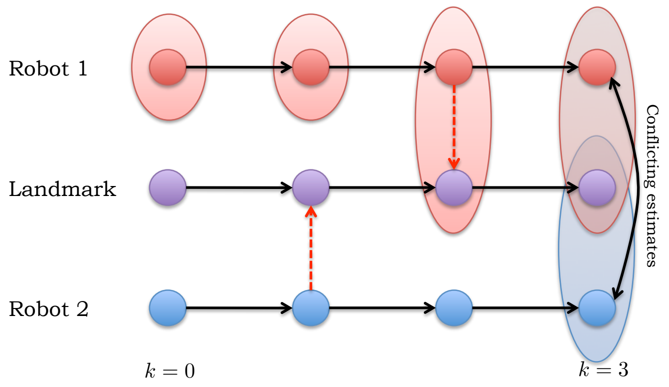
3.3.3 Initial Knowledge of Robots

Although we have extended the applicability of the theorems developed for decentralized cooperative localization to decentralized cooperative SLAM, these two problems do not share the same requirement on the initial knowledge of robots. In decentralized cooperative localization, when we have the communication range limit greater than the observation (measurement) range limit, it is unnecessary for a robot to know the total number of robots in the team and yet it can still obtain the centralized-equivalent estimate (i.e., subgroups of robots can be treated as individual systems) according to Theorem 3.3. Obtaining the centralized-equivalent estimate in decentralized cooperative SLAM is more restrictive, as the number of robots in the system needs to be known to all robots. To explain this, suppose that we separate the robots into subgroups that initially do not know of each other's existence for the decentralized cooperative SLAM problem. In this case, we are not guaranteed that the subgroup estimates are statistically independent due to the presence of landmarks. Consider Fig. 3.1 as an example, where there are two sub-groups consisting of a single robot each. Acting independently, each robot's self estimate will be correlated with the landmark, but they will not be aware that the landmark has been observed by both robots (i.e., the estimate of both robots and the landmark should all be correlated) until they encounter each other. Each robot will also apply the Markov property at every timestep because they only know of their own existence. When the two robots finally encounter each other, they will not be able to merge their estimates to form the centralized-equivalent estimate. To formalize the above idea, we have the following theorem (which is a counterpart to Theorem 3.3 in Chapter 2):

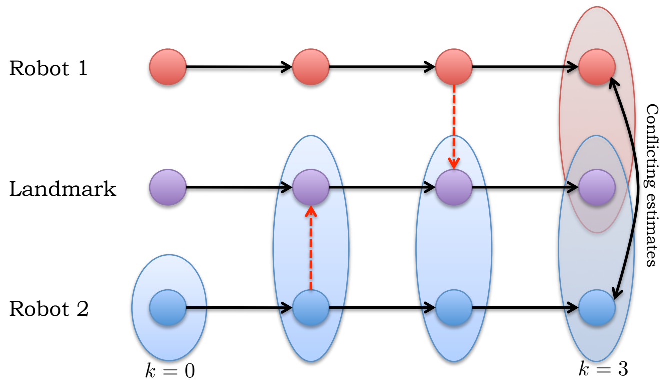
Theorem 5.1: *In decentralized cooperative SLAM, assume robot i does not initially know the total number of robots in the team. If robot i detects a partial checkpoint, $C_p(k_{c,i}, k_{e,i})$, based on robots known to it, $N_{i,k_{e,i}}$, and applies the Markov Property when the partial checkpoint exists, then robot i is not guaranteed to obtain the centralized-equivalent estimate for all robots and known landmarks when it knows of the existence of all robots.*



(a) Correlations in the centralized estimates.



(b) Correlations in robot 1's decentralized estimates.



(c) Correlations in robot 2's decentralized estimates.

Figure 3.1: In the above information flow graphs with measurements explicitly labelled, the correlations between the estimates of Robot 1, Robot 2, and a landmark are shown as shaded ellipses for: a) the centralized estimator, which have the estimates correctly correlated, b) the decentralized estimates made by robot 1 assuming that it initially only knows of its own existence, and c) the decentralized estimates made by robot 2 assuming that it initially only knows of its own existence. In the latter two cases, the robots will have applied the Markov property at every timestep. When the two robots finally communicate with each other, they will possess decentralized estimates that cannot be used to recover the centralized-equivalent estimate.

Proof. Since we want to show that the centralized-equivalent estimate is not guaranteed to be obtainable by robot i, we only need to show one example of when this is the case. We know that robot i knows of its own existence, and therefore, $i \in N_i$. Now suppose robot i takes a measurement, $\mathbf{y}^{m,i}_{i,k_m}$, of a landmark, m[3]. Suppose that another robot, $n \in \overline{N_i}$ (not known to robot i), also takes a measurement, $\mathbf{y}^{m,n}_{n,k_m}$, of the same landmark. We can show that robot i will know of landmark m when it detects the existence of the partial checkpoint $C_p(k_m, k_{e,i})$:

$$\begin{aligned} & \mathbf{y}^{m,i}_{i,k_m} \text{ exists} \\ \Rightarrow\ & \mathbf{y}^{m,i}_{i,k_m} \in S^{-}_{i,k_m} \\ \Rightarrow\ & \mathbf{y}^{m,i}_{i,k_m} \in S_{i,k_m} \\ \Rightarrow\ & \mathbf{y}^{m,i}_{i,k_m} \in S_{i,k_{e,i}} \\ \Rightarrow\ & m \in M_{i,k_{e,i}} \end{aligned}$$

Robot i however, will not know that robot n exists or that it has also made a measurement of landmark m, unless robots i and n were within the communication range when the measurement was made of landmark m at timestep k_m. This is not guaranteed to happen because we do not enforce that robots must remain within communication range of each other. Mathematically, suppose $\mathbf{y}^{m,n}_{n,k_m}$ exists, and that $d^{i,n}_k > r_{\mathsf{comm}}$:

$$\begin{aligned} & d^{i,n}_k > r_{\mathsf{comm}} \\ \Rightarrow\ & \mathbf{y}^{m,n}_{n,k_m} \in S_{n,k_m} \text{and } \mathbf{y}^{m,n}_{n,k_m} \notin S_{i,k_{e,i}} \\ \Rightarrow\ & n \notin N_{i,k_{e,i}} \end{aligned}$$

Now if robot i detects partial checkpoints based on robots known to it and applies the Markov property when a partial checkpoint exist, then its estimate, $\mathsf{bel}(X_{i,k_m})$, cannot be used to calculate the centralized-equivalent-estimate for all robots and known landmarks using (2.5) because of the correlation (i.e., the measurement made by robots i and n of landmark m) of which robot i does not know. In other words,

$$\begin{aligned} & \mathsf{bel}\left(X_{i,k_m}, \overline{X_{i,k_m}}\right) \neq \mathsf{bel}\left(X_{i,k_m}\right) \mathsf{bel}\left(\overline{X_{i,k_m}}\right) \\ \Rightarrow\ & \mathsf{bel}\left(X_{i,k}, \overline{X_{i,k}}\right) \neq \mathsf{bel}\left(X_{i,k}\right) \mathsf{bel}\left(\overline{X_{i,k}}\right), k \geq k_m \end{aligned}$$

Note here that we cannot calculate the centralized-equivalent estimate from the origi-

[3]This proof will also work if we start with the assumption that another robot $j \in N_i$ measures landmark m. However, we chose to proceed with the simpler case by having robot i make the measurement.

nal measurement and odometry data since they have been discarded when the Markov property was applied. Therefore, not only is robot i unable to use $\mathsf{bel}(X_{i,k_m})$ to determine the centralized-equivalent estimate for all robots and known landmarks at timestep k_m, future estimates calculated by recursive filtering also cannot be used to recover the centralized-equivalent estimate for all robots and known landmarks. $\square$

Corollary 5.1: *In decentralized cooperative SLAM, all robots must initially know the total number of robots in the team to guarantee that all robots can obtain the centralized-equivalent estimate for all robots and known landmarks in decentralized cooperative SLAM.*

Proof. From Theorem 5.1, we know that a robot is not guaranteed to obtain the centralized-equivalent estimate for all robots and known landmarks if the number of robots in the team is initially unknown, and if it detects partial checkpoints based on the number of robots known to it. Even if only one robot in the team does not initially know the total number of robots, there is the possibility that its estimate (which cannot be used to recover the centralized-equivalent estimate for all robots and known landmarks) will be passed to other robots. On the other hand, suppose that all robots initially know the total number of robots in the team, and only use the existence of partial checkpoints (based on having the required information from all robots) to determine when the centralized-equivalent estimate is obtainable (and when the Markov property can be applied). In this case, all robots are guaranteed to obtain the centralized-equivalent estimate for all robots and known landmarks. $\square$

In summary, in decentralized cooperative SLAM we cannot make use of the notion of independent subgroups, and we cannot guarantee that a robot can obtain the centralized-equivalent estimate (over all robots and known landmarks) without detecting a partial checkpoint based on information from all robots, which requires the robot to know the number of robots in the network. This is not a requirement in decentralized cooperative localization, under the condition that the communication range limit is greater than or equal to the observation range limit. However, due to the presence of landmarks, this 'loophole' that can be exploited in decentralized cooperative localization to allow partial checkpoint detection based on a subset of robots cannot be further exploited in decentralized cooperative SLAM. Although this may seem restrictive, we must remember that this is because we want all robots to be able to obtain the centralized-equivalent estimate in a sparsely-communicating robot network that is never guaranteed to be fully-connected.

Furthermore, we are no longer working with the reduced (robot-only) decentralized cooperative localization problem.

An alternative way of ensuring that the centralized-equivalent estimate is obtainable is to keep all robots from applying the Markov property (i.e., have all robots retain all information in their knowledge set through time). This, however, is impractical as computation and memory usage will increase indefinitely. Hence, we need to allow robots to know when they can apply the Markov property, which requires knowing the number of robots in the system.

To recapitulate the content of this section, we have generalized the concepts of a checkpoint and a partial checkpoint, allowing them to be applicable to the decentralized cooperative SLAM problem. The existence of a checkpoint is equivalent to the moment when all robots can obtain the centralized-equivalent estimate at the checkpoint occurrence time. Similarly, the existence of a partial checkpoint for a particular robot is equivalent to when it is possible for that robot to obtain the centralized-equivalent estimate. From this, we are then able to apply Theorems 3.1 and 3.2, which allow robots to apply the Markov property without consideration of other robots, and still ensure that all other robots can obtain the centralized-equivalent estimate. However, in decentralized cooperative SLAM, we require robots to initially know the total number of robots in the system. Through Theorem 5.1 and Corollary 5.1, we showed that the presence of landmarks makes knowing the number of robots in the team a mandatory initial requirement. Fig. 3.2 is an illustrative summary of the relationships between the theorems we have introduced up to this point.

3.4 Applying the Theory

In this section, we will present our decentralized cooperative SLAM algorithm, which is guaranteed to work in a sparsely-communicating robot network, and provide each robot with centralized-equivalent estimates over all robots and landmarks. We must stress again that this algorithm does not require robots to keep track of what other robots know, and each robot can decide to invoke the Markov property based on their local knowledge. Surprisingly, the algorithm is simpler in comparison to the decentralized cooperative localization algorithm (Algorithm 1 from Chapter 2) due to the requirement on initial knowledge that each robot needs to initially know the total number of robots in the team. Following the presentation of our algorithm, we will provide a computational complexity analysis, which is subsequently followed by simulation and hardware experiment results.

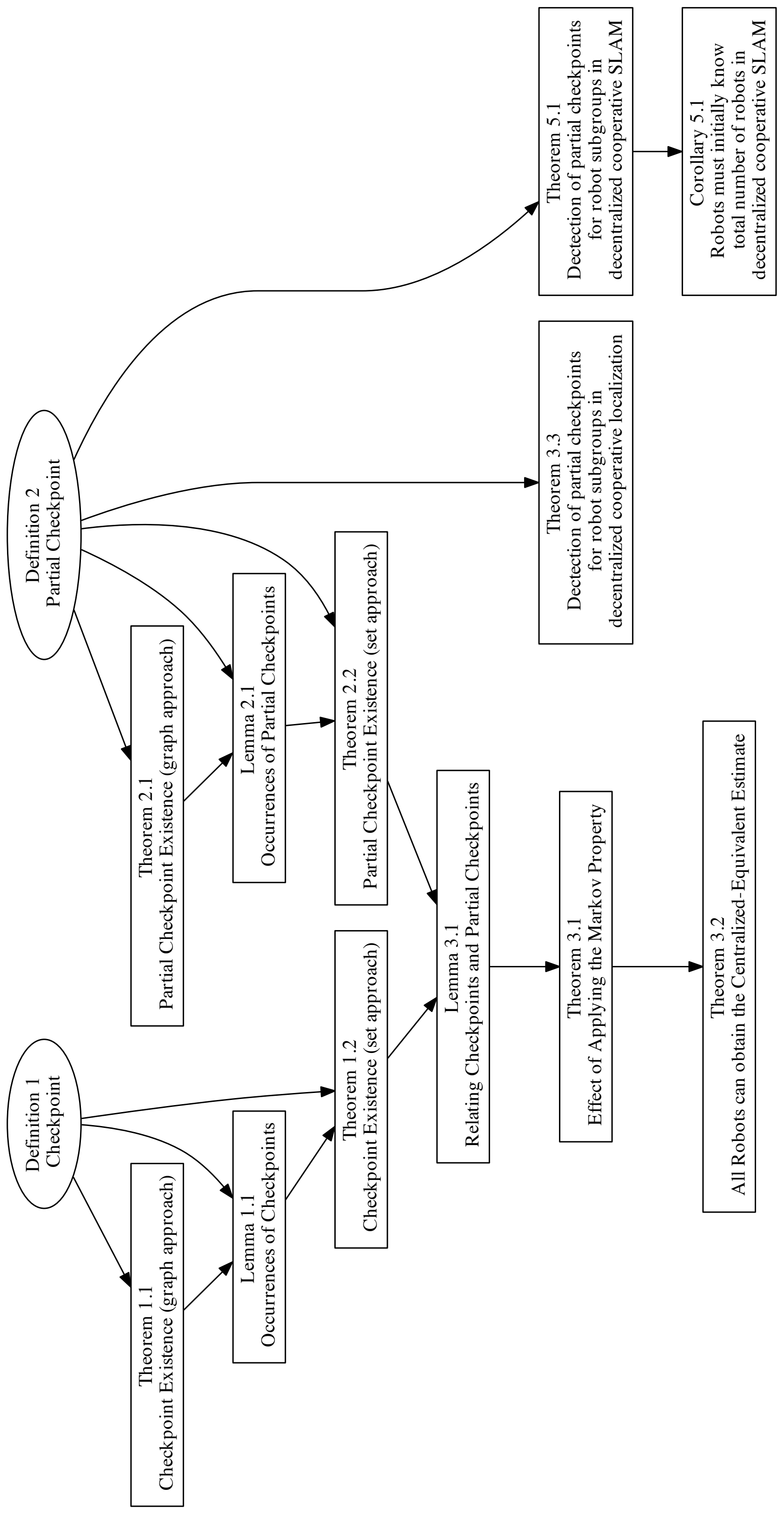

Definition 1
Checkpoint
Theorem 1.1
Checkpoint Existence (graph approach)
Lemma 1.1
Occurrences of Checkpoints
Theorem 1.2
Checkpoint Existence (set approach)
Definition 2
Partial Checkpoint
Theorem 2.1
Partial Checkpoint Existence (graph approach)
Lemma 2.1
Occurrences of Partial Checkpoints
Theorem 2.2
Partial Checkpoint Existence (set approach)
Lemma 3.1
Relating Checkpoints and Partial Checkpoints
Theorem 3.1
Effect of Applying the Markov Property
Theorem 3.2
All Robots can obtain the Centralized-Equivalent Estimate
Theorem 3.3
Dectection of partial checkpoints for robot subgroups in decentralized cooperative localization
Theorem 5.1
Dectection of partial checkpoints for robot subgroups in decentralized cooperative SLAM
Corollary 5.1
Robots must initially know total number of robots in decentralized cooperative SLAM

Figure 3.2: A graphical summary of the relationship between theorems introduced in chapters 2 and 3.

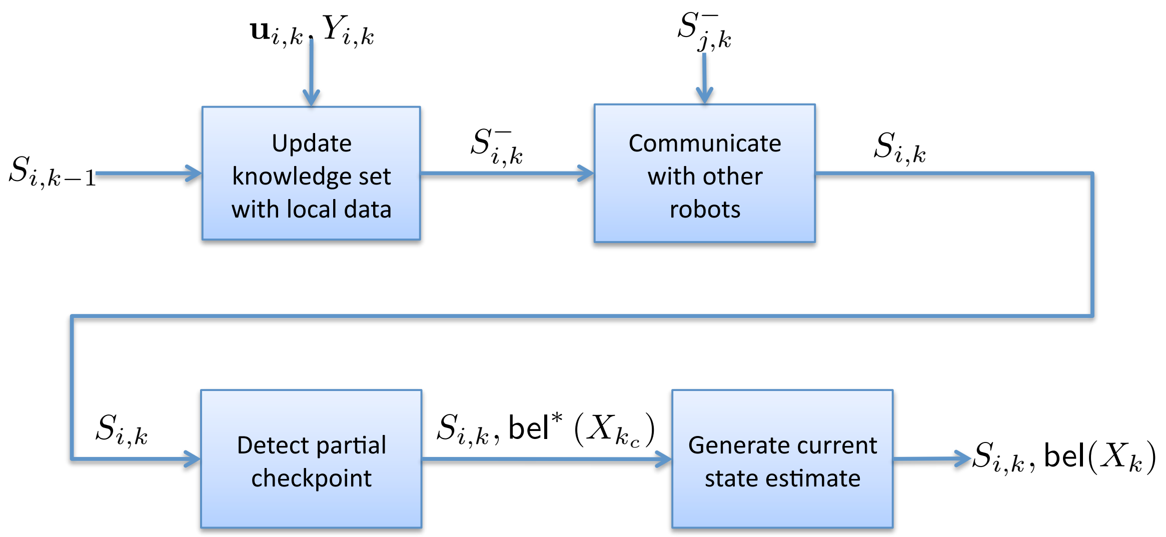

$\mathbf{u}_{i,k}, Y_{i,k}$
$S^-_{j,k}$
$S_{i,k-1}$
Update knowledge set with local data
$S^-_{i,k}$
Communicate with other robots
$S_{i,k}$
$S_{i,k}$
Detect partial checkpoint
$S_{i,k}, \text{bel}^*(X_{k_c})$
Generate current state estimate
$S_{i,k}, \text{bel}(X_k)$

Figure 3.3: The decentralized cooperative SLAM algorithm, which provides centralized-equivalent state estimates of all robots and map features whenever possible.

3.4.1 The Decentralized Cooperative SLAM Algorithm

Algorithm 2 is our decentralized cooperative SLAM algorithm, developed by combining the theory provided in Chapter 2, and section 3.3 of this chapter. Fig. 3.3 is a graphical representation of the algorithm.

```
Algorithm 2: Decentralized cooperative SLAM($k$, $\mathbf{u}_{i,k}$, $Y_{i,k}$, $S_{i,k-1}$, $S_{j,k}$ $(i \in N)(\forall j)(d_k^{j,i} \leq r_{\text{comm}})$)
1 $S_{i,k} \leftarrow S_{i,k-1} \cup \{\mathbf{u}_{i,k}\} \cup \{Y_{i,k}\} \bigcup_{j \in R_{i,k}} S^-_{j,k}$
2 $K \leftarrow \{k_r\}$such that$\{\mathbf{u}_{i,k_r} | i \in N\} \in S_{i,k}$
3 $k_{c,i} \leftarrow \max(K)$
4 $\tilde{S}_{i,k_{c,i}} \leftarrow S_{i,k} - \{\mathbf{u}_{j,k_r}, Y_{j,k_r} | j \in N\}\ \ (\forall k_r > k_{c,i})$
5 $\text{bel}^*\left(X_{k_{c,i}}\right) \leftarrow p\left(X_{k_{c,i}} | \tilde{S}_{i,k_{c,i}}\right)$
6 $S_{i,k} \leftarrow S_{i,k} \cup \text{bel}^*\left(X_{k_{c,i}}\right)$
7 $S_{i,k} \leftarrow S_{i,k} - \{\mathbf{u}_{j,k_r}, Y_{j,k_r}, \text{bel}^*\left(X_{k_r}\right) | j \in N\}\ \ (\forall k_r \leq k_c)$
8 $\text{bel}\left(X_k\right) \leftarrow p\left(X_k | S_{i,k}\right)$
9 return $\{\text{bel}\left(X_k\right), S_{i,k}\}$
```

Algorithm 2 executes at every timestep on all robots. On line 1, we update the knowledge set of robot i by implementing Equations (2.3) and (2.4). On lines 2 and 3, we search for the latest partial checkpoint. Line 4 defines the subset of knowledge that includes all information up to the partial checkpoint time. On line 5, we calculate an estimate for the partial checkpoint time. Note that bel^* indicates the centralized-equivalent estimate. On line 6, we proceed to discard information replaceable by bel^*

(invoking the Markov property), and enter the newly-calculated belief into the knowledge set on line 7. On line 8, we use all available information in the knowledge set to produce the current state estimate. Note that because the centralized-equivalent estimate can be delayed, we produce a temporary estimate until the centralized-equivalent estimate can be obtained. For the temporary estimate, robot i assumes robot j maintains its last known velocity if odometry data are not available. Due to this assumption, temporary estimates for robots that have been out of communication range for an extended period of time may have diverged from the real robot pose. Nevertheless, we are guaranteed that the centralized-equivalent estimate can be recovered when communication is reestablished.

Aside from being able to generate the centralized-equivalent estimate in a network that is never guaranteed to be fully connected, another feature of our cooperative decentralized SLAM algorithm is that we can use a variety of recursive filtering and data association methods within our framework (on line 5 and 8). The framework is not dependent on specific filtering methods such as the EKF, but the underlying assumption is that the Markov property is valid and applicable (i.e., any recursive approximation of the Bayes filter could be used within our framework).

One will notice that the overall structure of our decentralized cooperative SLAM algorithm is very similar to the decentralized cooperative localization algorithm when Fig. 3.3 is compared to Fig. 2.8. However, the actual algorithm listing is much longer and more complex for Algorithm 1. This difference is attributed to the initial knowledge requirement in decentralized cooperative SLAM, which requires every robot to initially know the total number of robots in the team. This prevents robots from starting as individual and independent sub-systems. As such, we no longer need to handle the merging of beliefs from multiple sub-systems when they encounter each other for the first time, and this simplifies the algorithm for decentralized cooperative SLAM.

3.4.2 Complexity Analysis

Complexity analysis of our decentralized cooperative SLAM algorithm is difficult because it is not possible to predict the connectivity of the dynamic robot network. Overall, the complexity of the algorithm is at least on the order of the filtering and data association methods implemented, and the computational requirement for a robot will increase as the number of timesteps since its last partial checkpoint (i.e., $k - k_c$) increases. Let us consider the worst-case scenario as shown in Fig. 3.4 where we have n robots and m landmarks. In this scenario, we have every robot measuring every other robot and all landmarks at every timestep. However, one robot does not communicate with the rest of the team. We will assume that the EKF [79] is used as the filtering method. The state of the system has a dimension that is proportional to the number of robots, n, and the number landmarks, m. At each timestep the number of measurements is on the order of n^2+nm, and the computational complexity for processing each measurement is quadratic with respect to the number of measurements. This implies that computational complexity is $O(n^4+n^3m)$ for the centralized estimator in the worst-case scenario. Furthermore, storing the covariance matrix and measurements requires memory usage of $O\left((n+m)^2\right)$.

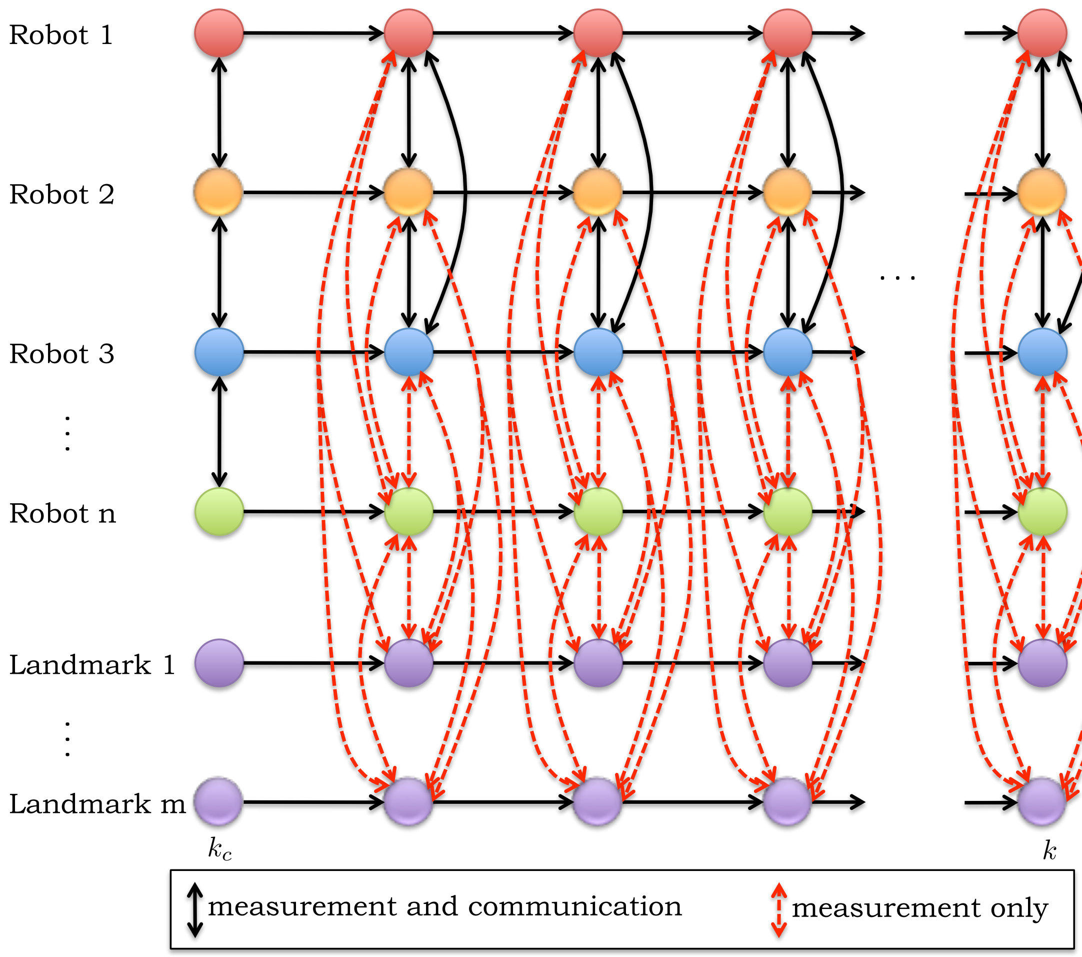


Figure 3.4: The worst-case scenario in decentralized cooperative SLAM in terms of memory usage and computational complexity.

For the same worst-case scenario, the complexity for calculating the current estimate with our decentralized cooperative SLAM algorithm is $O\left((k-k_c)(n^4+n^3m)\right)$ when a new partial checkpoint is discovered. The $k-k_c$ (delay) factor comes from the fact that we need to run the recursive filter from time k_c to the current time k. For the case where no new partial checkpoint is discovered, and by knowing the state estimate from the previous timestep $(k-1)$, computational complexity is $O(n^4+n^3m)$ (i.e., the same as the centralized estimator). Memory usage requirement for the decentralized estimator is $O\left((k-k_c)(n+m)^2\right)$. Again, the $k-k_c$ factor comes from the fact that we need to keep all the information since the previous partial checkpoint time to ensure that the centralized-equivalent estimate is obtainable. Lastly, the communication bandwidth requirement is $O((k-k_c)(n^3+n^2m))$ for our decentralized approach. In practice, we can reduce communication bandwidth by having each robot remember the latest information sent to other robots (but note that this does not imply having to know what each robot has in their knowledge set).

In comparison to the decentralized cooperative localization algorithm, the complexity analysis above is very similar. However, one must not forget that in decentralized cooperative SLAM, the number of states can be vastly greater than in the decentralized cooperative localization case due to the presence of landmarks. In either case, there is an additional cost for having the guarantee that the centralized-equivalent estimate is obtainable by all robots. This is the cost of operating in a network that is never guaranteed to be fully connected.

3.4.3 Simulations

We will use a simulation as a first validation of Algorithm 2. In the scenario, illustrated in Fig. 3.5, the robots are moving in circles at constant forward and angular velocities, and the communication range is limited such that a 'weaving' pattern emerges from the communication linkages established by the robots in both circles. This also prevents the network from ever being fully connected. As a consequence, there will be a delay to when the centralized-equivalent estimate for the current timestep can be obtained (i.e., we need to wait until a partial checkpoint occurs for the current timestep). This also implies that the estimates shown are the temporary estimates made at each timestep with the information available to the robot. Recall that we assume other robots maintain their last known forward and angular velocities if this information is not available. Nevertheless, our algorithm guarantees that the centralized-equivalent estimate can always be recovered (when a partial checkpoint does indeed exist). Fig. 3.6 shows several components of the (current, and temporary) decentralized estimate from Robot 1 (red). Note that even though there is a delay to when the centralized-equivalent estimate is obtainable, its (current and temporary) estimate of its own x-position as well as a landmark's x-position is very close to the centralized estimate. Robot 1's estimate of another robot, however, exhibits greater differences due to the loss of communication. Although this simulation scenario is simple, it demonstrates that our algorithm can be used in a network that is never fully connected. In the following section, we will implement our decentralized cooperative SLAM algorithm on real hardware.

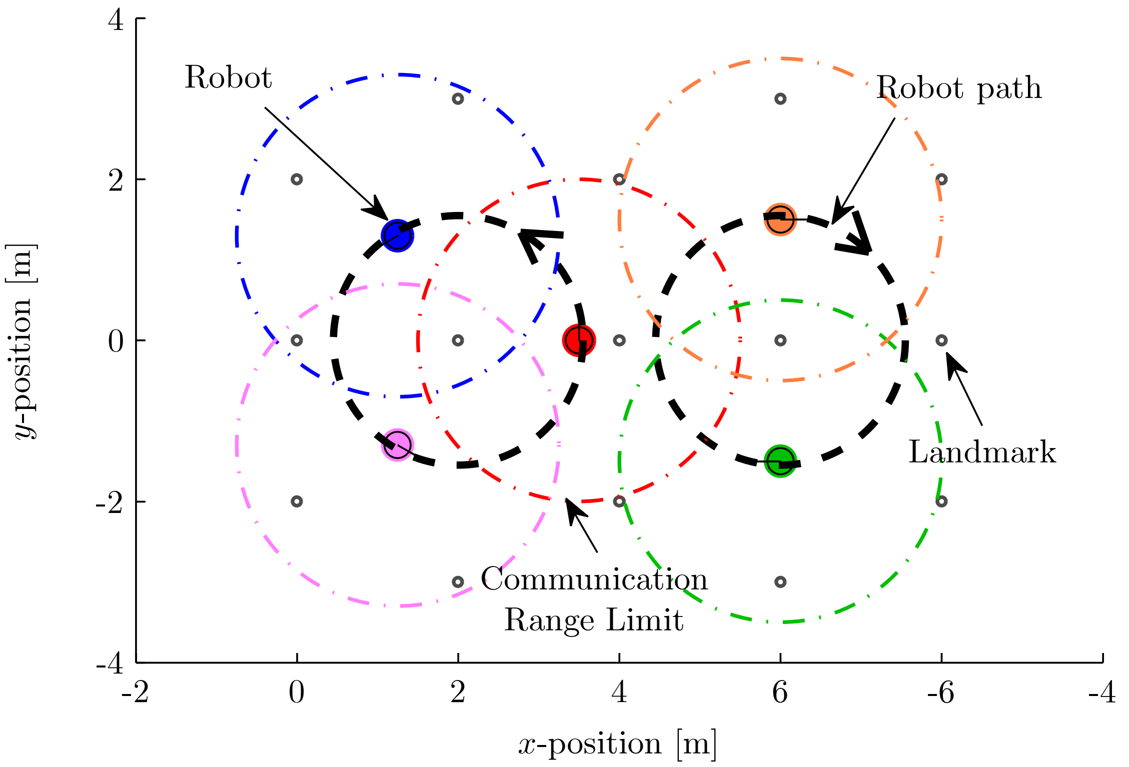
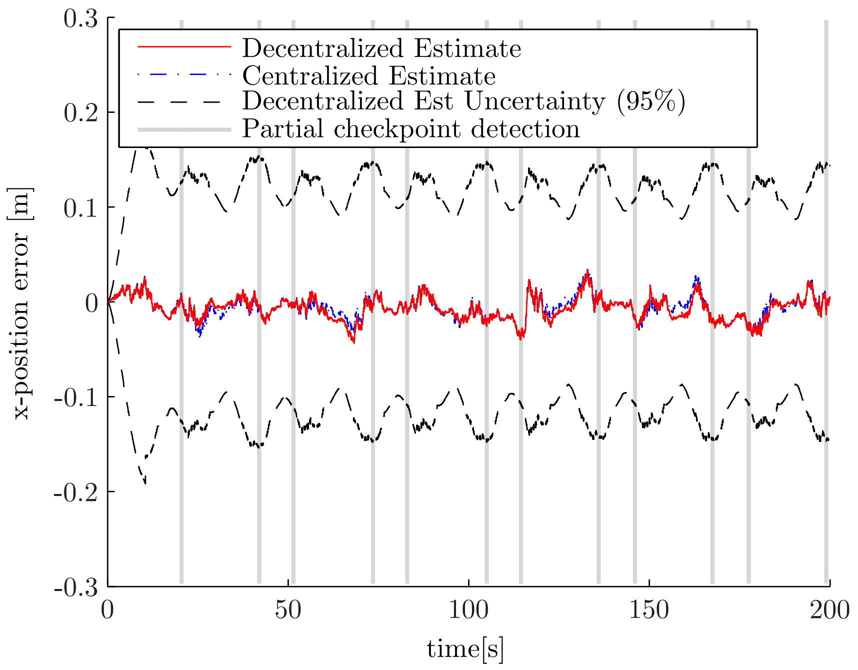


Figure 3.5: A simulation of a weaving robot network that is never fully connected.

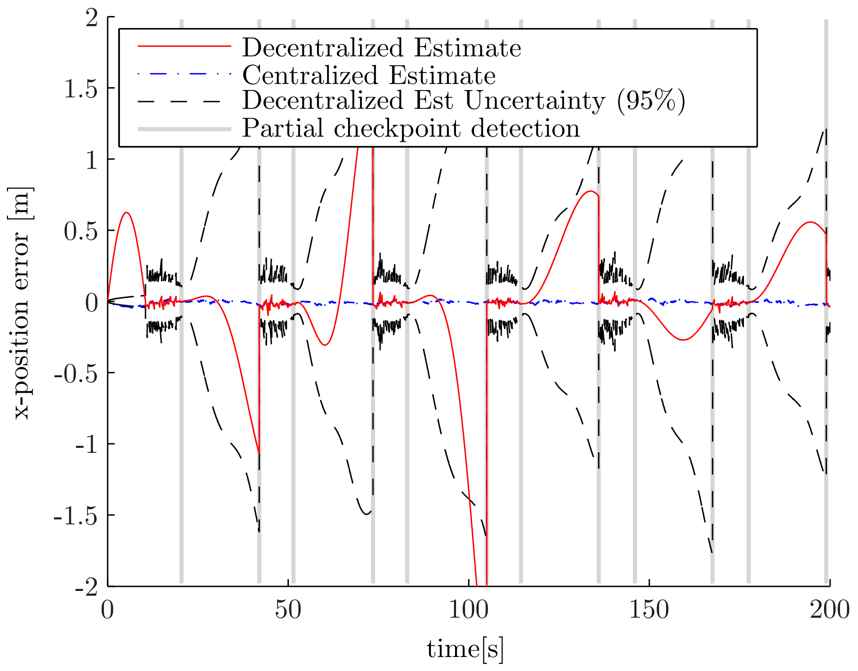
3.5 Hardware Experiment Dataset

To provide a more realistic evaluation of our decentralized cooperative SLAM algorithm, we have created a two-dimensional multi-robot cooperative localization and mapping dataset[4]. The dataset consists of 9 sub-datasets, each ranging between 15 to 70 minutes in duration. It is produced using a fleet of 5 mobile robots in an indoor workspace with 15 static landmarks (which are repositioned for each sub-dataset). Each sub-dataset

[4]This dataset has been published in the International Journal of Robotics Research[102]

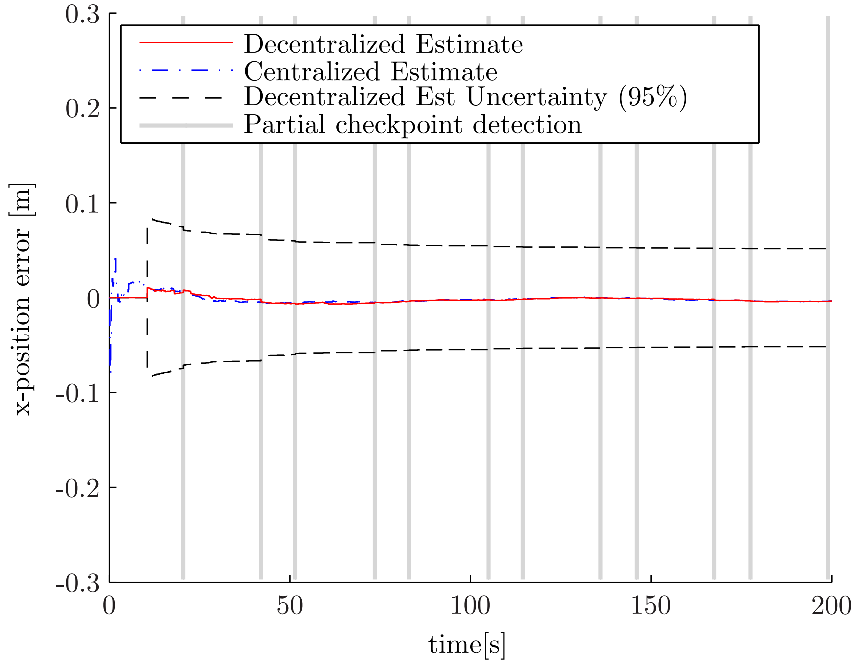


(a) Robot 1's estimate of its own x-position.



(b) Robot 1's estimate of robot 3's x-position.

Figure 3.6: Components of robot 1's decentralized estimate for the scenario shown in Fig. 3.5



(c) Robot 1's estimate of a landmark's x-position.

Figure 3.6: Components of robot 1's decentralized estimate for the scenario shown in Fig. 3.5

contains time-stamped odometry (velocities) and range-bearing measurements that are logged by each robot. In addition, by using a 10-camera motion capture system, we are able to obtain precise groundtruth data for all robot poses (position and orientation), as well as groundtruth landmark positions (which is rarely available in SLAM problems). Fig. 3.7 is a photograph taken during the production of a dataset.

We will now provide a more in-depth description of the workspace and equipment used to collect the dataset, the data collection process, and how the dataset is used to validate our decentralized cooperative SLAM algorithm.

3.5.1 Data Collection

The multi-robot cooperative localization and mapping dataset is produced in a rectangular indoor space with approximate dimensions of 15m×8m. The floor in this space is visually flat. For each sub-dataset, 15 landmarks and 5 robots are placed in the workspace. For the duration of each sub-dataset, each robot logs its own odometry data while driving to randomly generated waypoints in the workspace while avoiding obstacles (landmarks and other robots). When another robot or landmark is in the field of view of a robot, a range and bearing measurement is made and logged. Throughout the entire

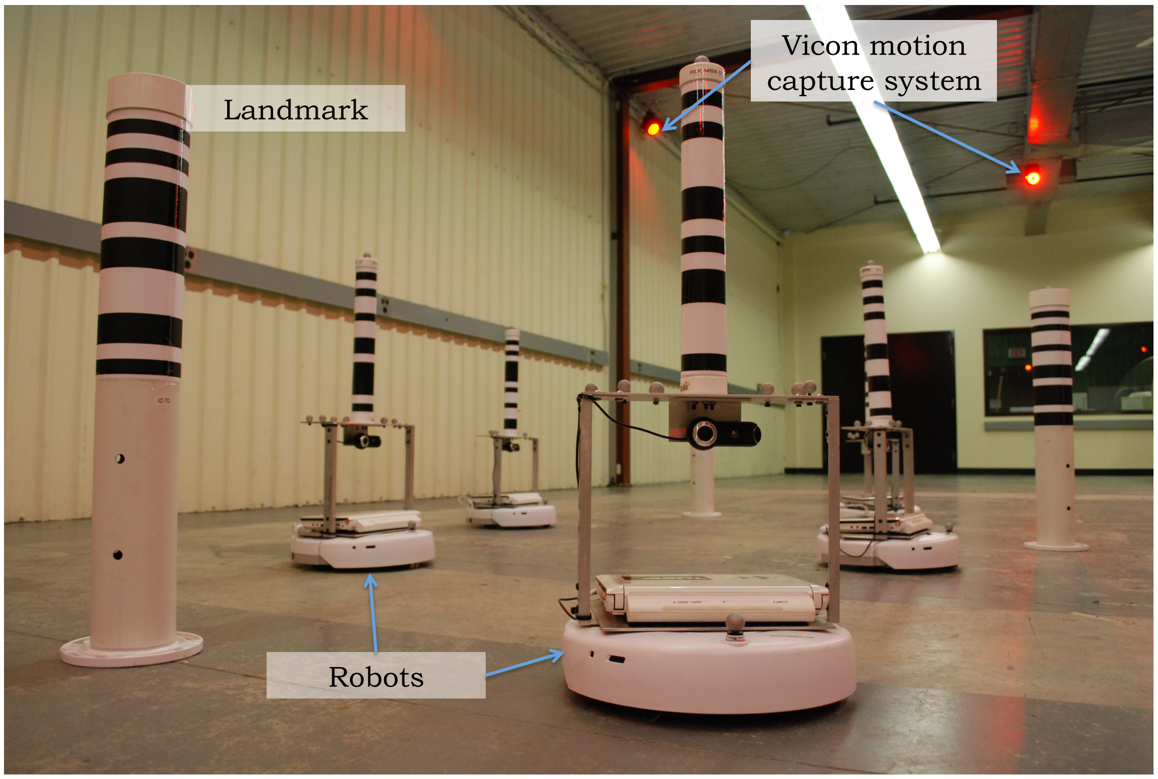


Figure 3.7: The data collection process for the multi-robot cooperative localization and mapping dataset uses a fleet of 5 robots. Groundtruth data for all landmark positions and robot poses is provided by a 10-camera Vicon motion capture system.

data collection process, a 10-camera Vicon motion capture system monitors and logs the pose of each robot, as well as the position of all landmarks, for groundtruth data.

The Robots

The fleet of 5 robots used to produce the dataset are identical in construction (see Fig. 3.8), and are built from the iRobot Create (two-wheel differential drive) platform. Each robot is equipped with a computer, which interfaces with a monocular camera that serves as the primary sensing instrument. Conveniently, the camera is mounted at a location where the measurement coordinate frame coincides with the robot's body frame in 2-d, as shown in Fig. 3.9. A cylindrical barcode tube is installed on each robot and is centered above the robot's body frame. Each barcode pattern is 30cm in height and allows for the identification of a robot. Range and bearing measurements can be made when the barcode is detected by another robot.

In terms of software, each robot runs the *Player*[5] server [103]. This software is re-

[5]Player is a open-source software which serves as a network server over which hardware and software

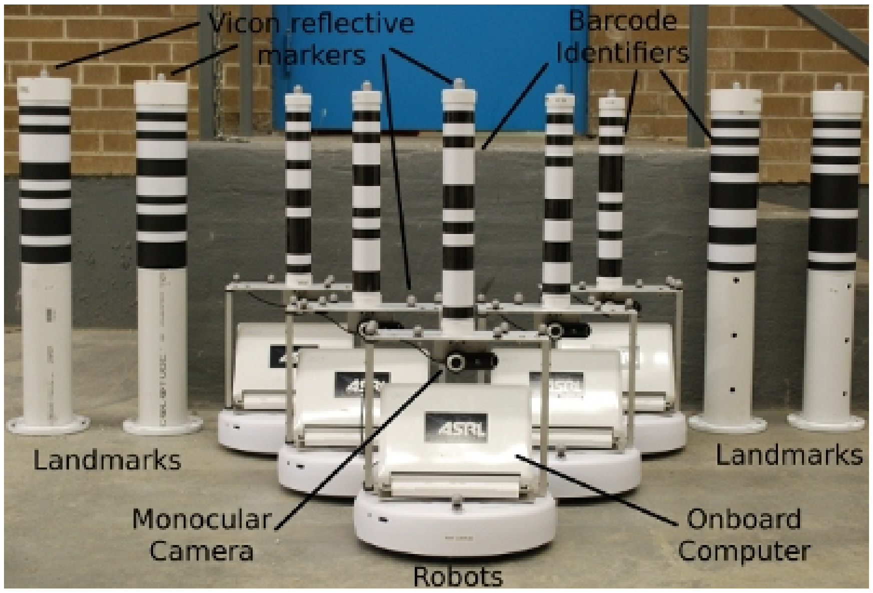


Figure 3.8: The fleet of 5 robots and a sample of the landmarks used in the making of the multi-robot cooperative localization and mapping dataset. The barcode patterns are used in producing range-bearing measurements from the camera mounted on each robot.

sponsible for performing motion control, obstacle avoidance, image processing (for measurements), as well as data logging through a collection of custom-coded and default drivers.

Landmarks

Cylindrical tubes similar to the ones installed on each robot (but larger in diameter) serve as landmarks. Each landmark has an unique barcode pattern that is 30cm in height for identification and measurement purposes.

Odometry

Forward velocity (along the x-axis of the robot body frame) commands, v, and angular velocity commands (rotation about the z-axis of the robot body frame using the right-hand rule), ω, are logged at an average of 67Hz as odometry data. The maximum forward velocity of a robot is 0.16m/s, and the maximum angular velocity is 0.35rad/s.

abstracted as standard interfaces can interact with each other.

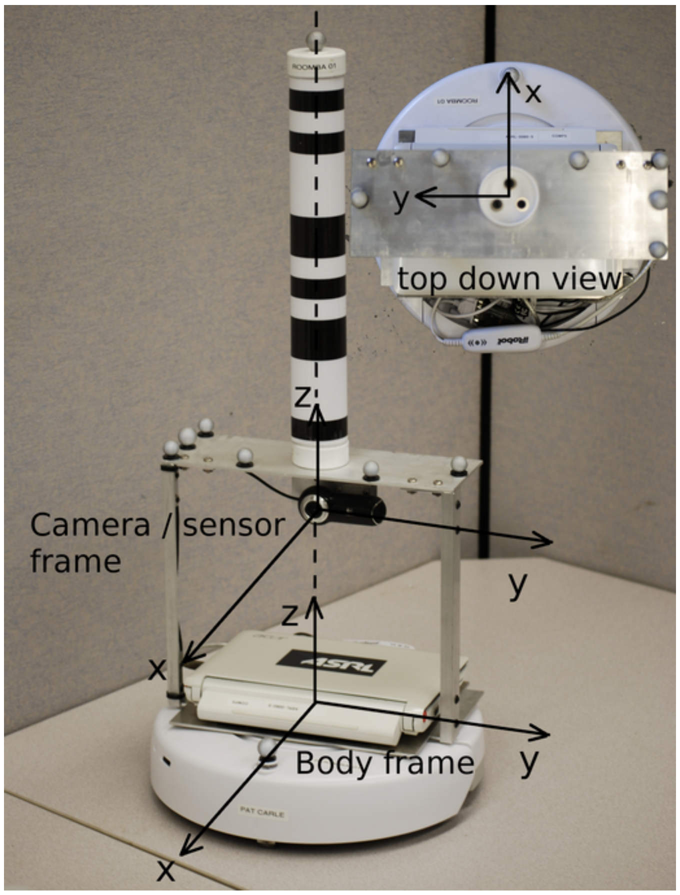


Figure 3.9: Reference frames for the robot body, and the sensor (camera). The frames are aligned such that the transformation between them only consists of a translation in the z direction.

Measurements

During dataset production, images captured by the monocular camera on each robot at frame rate of 10Hz and a resolution of 960×720 pixels are processed to extract range and bearing measurements from barcode patterns in the field of view. The cameras have a field of view of approximately 60 degrees. The camera on each robot is individually calibrated using a planar checkerboard [104] to obtain the parameters necessary to undistort images. A process that primarily relies on edge detection is used on the undistorted gray-scale images for detecting barcodes. Fig. 3.10 shows the typical output of the barcode detection process. Range measurements are obtainable since the vertical focal length and the barcode height are known. Bearing measurements correspond to the horizontal position of a barcode in an image. The sources of measurement error include: lighting conditions, slight tilting of the robot on the presumed-leveled floor, and motion blur.

Each barcode pattern begins (from top to bottom) with a black-white-black start

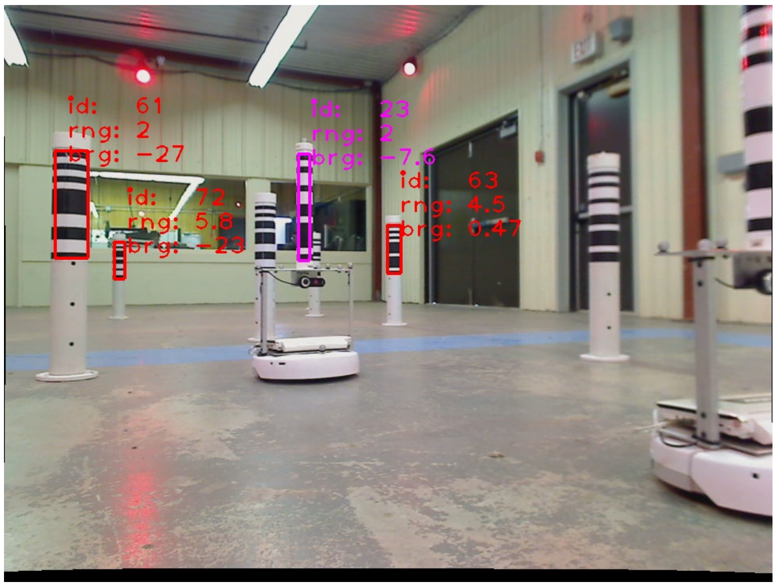


Figure 3.10: An example of barcode detection from an undistorted image. A detected barcode provides a range-bearing measurement, as well as a unique identification signature.

code, followed by a series of alternating black and white bars, which encodes 2 digits in accordance with the *Universal Product Code (UPC)* standard. To reduce the chance of misreading a barcode, the 2 digits that identify a robot must add up to a checksum of 5, while the digits for a landmark must sum to 7 or 9. In rare instances, mislabelled barcodes have been found when barcodes partly occlude one another. These instances are easy to detect using the groundtruth data.

Occlusions

For sub-dataset 9, barriers were placed in the workspace between landmarks to occlude the robots' views, as shown in Fig. 3.11. This resulted in a decrease in the number of measurements made by the robots, and also a decrease in the maximum measurement range. Fig. 3.12 is a map of the barriers in the workspace.

Groundtruth

A Vicon motion capture system consisting of 10 (MX-40+) cameras provides groundtruth data for the dataset. The groundtruth coordinate frame also serves as the inertial reference frame, in which the positions of landmarks (x, y), and robot poses (x, y, θ) are reported. A unique constellation of reflective markers on each robot allows the system to identify and track the local body frame (position of orientation) of each robot in 3d

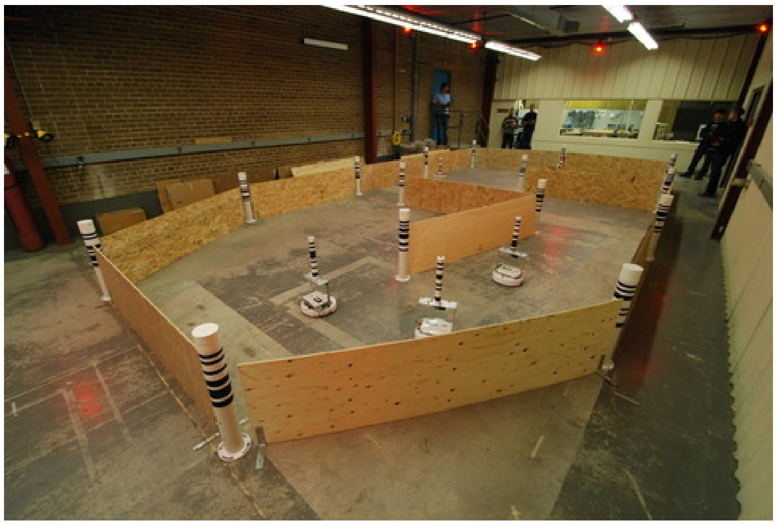


Figure 3.11: In sub-dataset 9, data collection was performed with barriers in the workspace to increase occlusions for measurements.

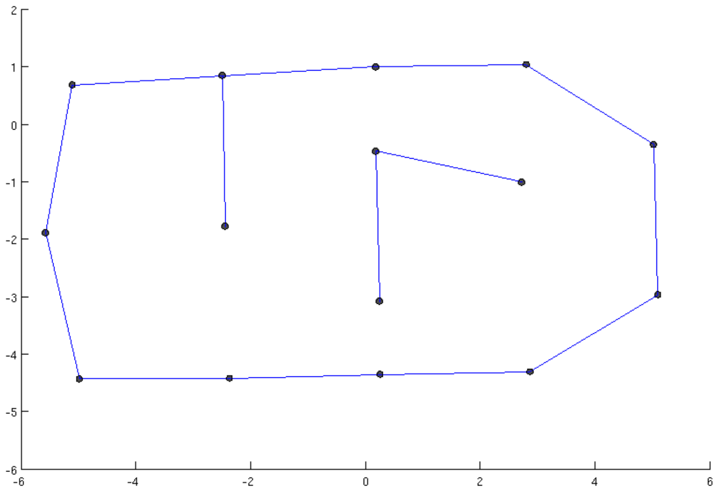


Figure 3.12: A map of the barriers in sub-dataset 9 used for measurement occlusions.

at 100 Hz. Reflective markers are centered on the top of each landmark cylinder so that landmark positions are also recorded. Vicon claims that the accuracy of their tracking system is on the order of 1×10^{-4}m, and a study (on an older system) [105] appears to support this claim. However, for the workspace used in producing the datasets, we believe that positional accuracy is on the order of 1×10^{-3}m due to the specific configuration of the cameras in the workspace, as well as the ability of the Vicon system to track numerous moving bodies simultaneously. Occasionally, the Vicon system fails to identify a robot from its constellation of markers. The log entries of these occasions (identifiable by observing zero values for all position and rotation components) have been conveniently removed from the dataset. In rare occasions, the Vicon system will also make an error fitting the proper model to the constellation of markers representing a robot. These instances (detectable by physically impossible changes in position and orientation) have also been removed from the groundtruth data. Since our datasets are in 2d, position information in the global z direction, as well as body pitch and roll angles, have been excluded.

Time Synchronization

A *Network Time Protocol* (NTP) daemon is used to synchronize the clocks between the computers and each robot, as well as the groundtruth data logging computer. The performance of the NTP daemon is checked prior to the collection of each sub-dataset. For each sub-dataset, the average error reported by the NTP daemon on each computer is on the order of 1×10^{-3}s, while the maximum error is on the order of 1×10^{-2}s. Based on this, we consider the clocks on all computers to be synchronized. Timestamps are applied to all odometry data, range and bearing measurements, groundtruth data, and camera images. We have also accounted for delays in logging measurements due to image processing time.

3.5.2 Data Usage

In validating our decentralized cooperative SLAM algorithm with the hardware dataset, we imposed communication range limits on the robots by using their groundtruth positions. We also limited communication intervals to every 0.5s (i.e., robots waited for 0.5s between exchanging knowledge sets using (2.3) and (2.4)). The observation range, r_{obs}, was also limited to 3m by ignoring measurements that have a greater range. We validated our algorithm on each of the sub-datasets. Each validation trial is considered to be completed when we have processed all the measurement data available.

Similarly to the validation of our decentralized cooperative localization algorithm in Chapter 2, the EKF was used again as the filtering method in our decentralized cooperative SLAM algorithm for estimating robot poses and landmark positions. The motion and measurement models, (2.6, 2.7, 2.8), presented previously in Section 2.5.1, were used in propagating and updating the estimate mean and covariance. Calibration data with groundtruth information were collected separately from the hardware dataset for the purpose of determining the noise parameters in the motion and measurement models. Using groundtruth information, we determined the errors between velocity commands and actual velocities. Similarly, we examined the errors between groundtruth-predicted measurements and the actual measurements. From the error statistics, we adjusted for biases in our inputs and measurements, and determined the covariance associated with each noise parameter. Due to estimator inconsistency with the EKF [106], we examined the normalized estimator error squared *normalized-estimator-error-squared* (NEES) using data separate from our experimental dataset. The covariances for input and measurement noises were inflated until a satisfactory level of estimator consistency was achieved. This gave us the following values:

$$\boldsymbol{\epsilon}_k \sim \left(\mathbf{0}, \begin{bmatrix} 5.075\ {v_{i,k}}^2 & 0 \\ 0 & 0.345\left(\frac{\mathrm{rad}}{\mathrm{s}}\right)^2 \end{bmatrix}\right), \quad \boldsymbol{\delta}_k \sim \left(\mathbf{0}, \begin{bmatrix} 0.0215\ \mathrm{m}^2 & 0 \\ 0 & 0.01\ \mathrm{rad}^2 \end{bmatrix}\right)$$

In order for us to focus on the state estimation aspect of decentralized cooperative SLAM, we performed data association using barcode identification (i.e., we assumed known data association). Note, however, that our framework allows for the implementation of other data association techniques if data association is unknown. We will examine the topic of data association in more detail in Chapter 4. In SLAM, the system state needs to be augmented in the estimator when a new landmark is observed. The augmented state of the system can be partitioned as

$$\begin{bmatrix} \mathbf{x}'_k & \mathbf{x}_{m,k} \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} \mathbf{x}_{j,k} & \mathbf{x}_{i,k} & \mathbf{x}_{m,k} \end{bmatrix}^{\mathrm{T}},$$

where $\mathbf{x}'_k$ is the state of the system before augmentation, and $\mathbf{x}_{m,k}$ is the state of the newly observed landmark. The state $\mathbf{x}'_k$ can be further partitioned, such that $\mathbf{x}_{i,k}$ represents the state of the robot that observed the new landmark, and $\mathbf{x}_{j,k}$ represents all other states in the system before augmentation. The state of the new landmark is defined as

$$\mathbf{x}_{m,k} = \begin{bmatrix} x_{m,k} \\ y_{m,k} \end{bmatrix} = \mathbf{f}(\mathbf{x}_{i,k}, \mathbf{y}^{m,i}_{i,k}) = \begin{bmatrix} x_{i,k} \\ y_{i,k} \end{bmatrix} + \begin{bmatrix} r^{m,i}_{i,k} \cos\left(\theta_{i,k} + \phi^{m,i}_{i,k}\right) \\ r^{m,i}_{i,k} \sin\left(\theta_{i,k} + \phi^{m,i}_{i,k}\right) \end{bmatrix},$$

where $\mathbf{y}^{m,i}_{i,k} = \left[r^{m,i}_{i,k}, \phi^{m,i}_{i,k}\right]$ is the measurement. If we represent the covariance of $\mathbf{x}'_k$ as

$$\boldsymbol{\Sigma}' = \begin{bmatrix} \boldsymbol{\Sigma}_{j,j} & \boldsymbol{\Sigma}_{j,i} \\ \boldsymbol{\Sigma}_{i,j} & \boldsymbol{\Sigma}_{i,i} \end{bmatrix},$$

then the covariance matrix for the augmented system becomes

$$\begin{bmatrix} \boldsymbol{\Sigma}_{j,j} & \boldsymbol{\Sigma}_{j,i} & \boldsymbol{\Sigma}_{j,i}{\mathbf{F}_{\mathbf{x}'_k}}^{\mathrm{T}} \\ \boldsymbol{\Sigma}_{i,j} & \boldsymbol{\Sigma}_{i,i} & \boldsymbol{\Sigma}_{i,i}{\mathbf{F}_{\mathbf{x}'_k}}^{\mathrm{T}} \\ \mathbf{F}_{\mathbf{x}'_k}\boldsymbol{\Sigma}_{i,j} & \mathbf{F}_{\mathbf{x}'_k}\boldsymbol{\Sigma}_{i,i} & \mathbf{F}_{\mathbf{x}'_k}\boldsymbol{\Sigma}_{i,i}{\mathbf{F}_{\mathbf{x}'_k}}^{\mathrm{T}} + \mathbf{F}_{\mathbf{y}^{m,i}_{i,k}}\mathbf{Q}_k{\mathbf{F}_{\mathbf{y}^{m,i}_{i,k}}}^{\mathrm{T}} \end{bmatrix},$$

where,

$$\mathbf{F}_{\mathbf{x}'_k} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}'_k}, \quad \mathbf{F}_{\mathbf{y}^{m,i}_{i,k}} = \frac{\partial \mathbf{f}}{\partial \mathbf{y}^{m,i}_{i,k}},$$

and $\mathbf{Q}_k$ is the covariance for the measurement noise [6].

3.6 Performance Analysis

The communication range is an important factor to consider when we interpret the results from our decentralized estimator. As communication range increases, we can expect the percentage of time that the network is fully connected to also increase. Furthermore, we can expect that robots have more opportunities to communicate with each other. For each of the sub-dataset, the percentage of time that the network is fully connected is shown in Fig. 3.13. The zero-to-one transition phenomenon observed in the figure is typical for a *fixed-radius-random-graph-model* [85], which we are using for generating our robot network. We have indicated on the figure the data that correspond to sub-dataset 1 of our experiments, as we will be examining the performance of our decentralized estimator for this test in detail. We picked this test in particular because it displayed the lowest percentage of full connectivity time for all communication range limits out of all the sub-datasets (i.e., sub-dataset 1 is the worst-case).

We will begin by examining the results from test 1 in detail. In particular, we will look at robot 1's estimates. Detailed results from the other 4 robots are similar to those of robot 1 and are not shown. Fig. 3.14 shows the result of our decentralized cooperative SLAM algorithm at 30 minutes into test 1. In this case, the communication range was limited to 2m, resulting in the network being fully connected only 21% of the time.

Fig. 3.15 displays the memory usage of robot 1 in running the decentralized cooperative SLAM algorithm on sub-dataset 1, for the cases where the communication range limit is limited to 1m, 2m, and 3m. Referring to Fig. 2.10, these cases correspond to 0.6%, 20.7%, and 52.7% of full network connectivity, respectively. Fig. 3.15 also shows the hypothetical memory usage of robot 1, had it never applied the Markov property. In using our decentralized cooperative SLAM algorithm, we are able to limit memory usage because our algorithm makes use of the Markov property. Temporary increases in memory usage are caused by the loss of connectivity to other robots in the network (i.e., Robot 1 needs to retain information since its last partial checkpoint occurrence time). These increases are followed by sharp decreases, which signal that Robot 1 has

[6]Refer to [107] for a more in-depth explanation of state augmentation in SLAM.

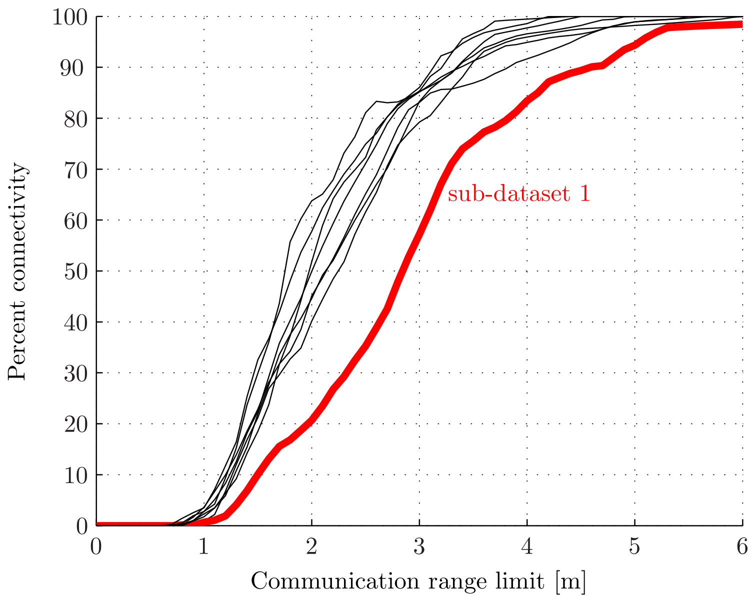


Figure 3.13: Percentage of time when the network is fully connected as a function of communication range limit in each dataset.

applied the Markov property (i.e., Robot 1 has detected a new partial checkpoint). Note, however, that decreases in memory usage do not necessarily imply that the network is fully connected. For the case where $r_{\mathsf{comm}} = 1$m, the network is only fully connected 0.6% of the time. Yet, Robot 1 is still able to occasionally apply the Markov property because information from other robots can be relayed. As the communication range limit increases, we can see that memory usage decreases, and the Markov property is applied more frequently. This is because robots have more opportunities to communicate with each other as communication range limit increases. Similar trends in memory usage are observed for all other robots, and in all other tests.

Fig. 3.16 shows the amount of information communicated with robot 1 in test 1 for cases with different communication range limits. The amount of information communicated remains relatively low for most timesteps in all cases. Instances where there is a significant increase in the amount of information communicated indicate that robot 1 has just re-established connection with one or more robots (i.e., the robots are updating each other with the latest information in their knowledge sets). As the communication range limit decreases, robots are more likely to lose connection with other robots and experience longer durations of disconnection. This explains why more information is exchanged when communication is re-established.

We will compare the performance of our algorithm against that of the centralized EKF-SLAM algorithm. In order for the centralized estimator to work, we allow it to cheat

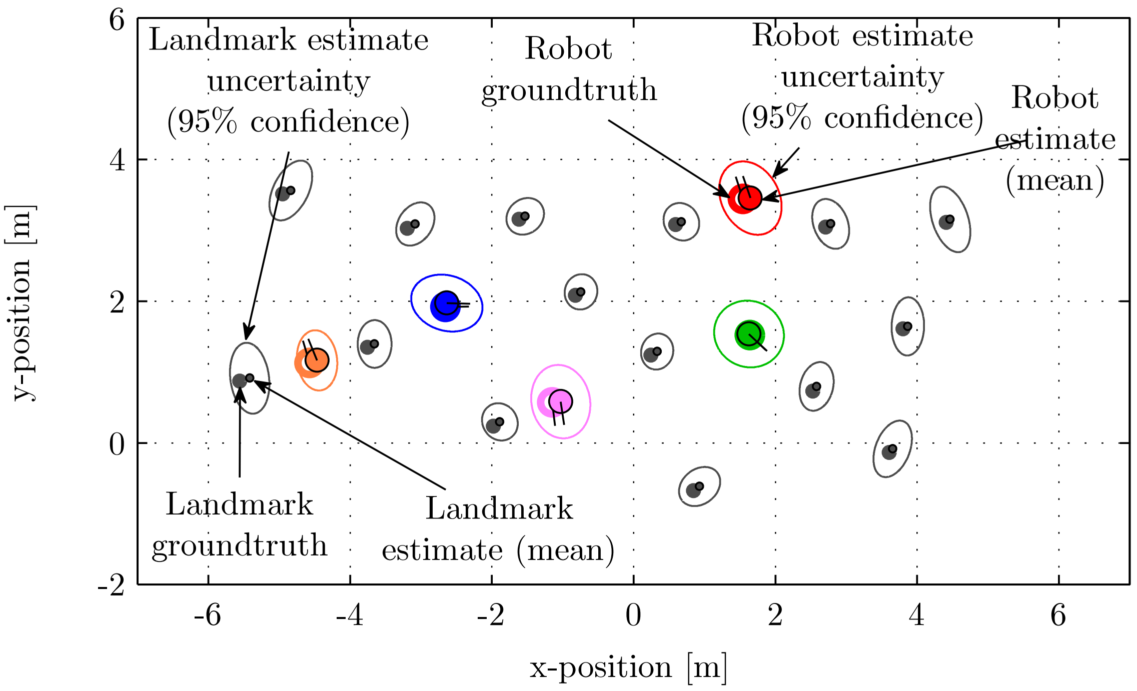


Figure 3.14: A graphical representation of the results from sub-dataset 1 after 1800s. The ellipses over all robots and landmarks are projections of the decentralized estimate calculated by Robot 1 (the red robot). The communication range was limited to 2m, and the observation range was limited to 3m.

by ignoring the communication constraints (i.e., the robot network is fully connected at all times for the centralized estimator). Figs. 3.17, 3.18, and 3.19 are a collection of graphs showing several components of robot 1's estimate error for the cases where the communication range limit is limited to 1m, 2m, and 3m. The components shown include robot 1's estimate error of its own x-position, robot 1's estimate error of robot 3's x-position, and robot 1's estimate error of a landmark's x-position. These components shown are representative of a robot's estimate error of its own pose (for which odometry information is always available), a robot's estimate error of another robot's pose (for which odometry data are not always available), and a robot's estimate error of a landmark position. In each of the plots, we also compare with the centralized estimator error. It is important to remember that we are allowing the centralized estimator to cheat by ignoring the communication restrictions. The uncertainty regions shown on each plot are projections of the 95% confidence ellipsoid for the full-state decentralized estimate (i.e., including all robots and landmarks). The grey areas in the plots indicate the timesteps at which partial checkpoints were detected. In other words, these are instances when robot 1 is able to obtain centralized-equivalent estimates. However, note that what is

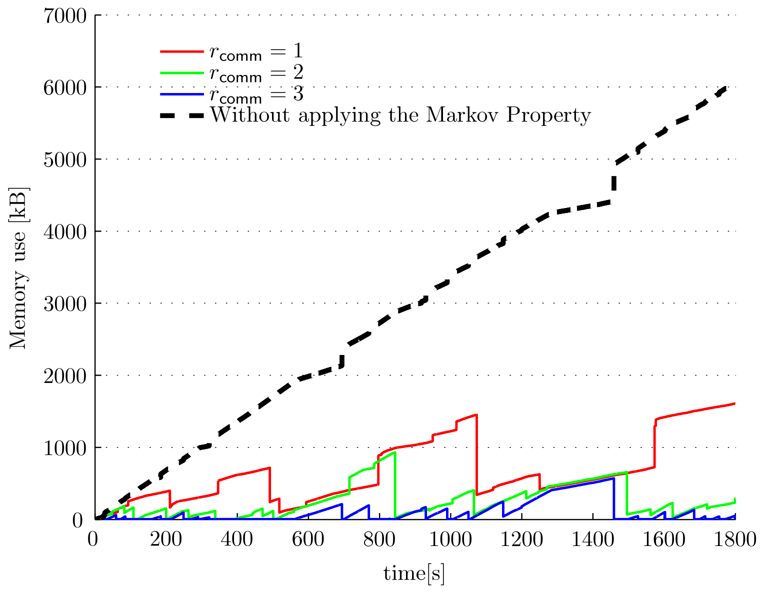


Figure 3.15: Memory usage of robot 1 in test 1.

shown is the existence of partial checkpoints at the indicated timesteps. The occurrence time of these partial checkpoints may be delayed depending on the network connectivity. Hence, even if partial checkpoint existence is detected, the estimate shown may still be a temporary estimate.

In Fig. 3.17 and 3.19 , we can see that robot 1's self estimate and its estimate of a landmark match closely with the centralized state estimates, even in the $r_{\text{comm}} = 1$m communication range limit case where the network is only fully connected 0.6% of the time. Larger differences can be observed between the centralized and decentralized for robot 1's estimate of robot 3's x-position in Fig. 3.18. Instances of larger differences occur due to the loss of communication between robot 1 and 3. During this time, robot 1 assumes the last known velocity of robot 3 to calculate an estimate. However, this estimate is only temporary as discussed in Section 3.4. The difference between the decentralized and centralized estimate decreases eventually, when robot 1 communicates with robot 3 again, or obtains information of robot 3 through another robot. As the communication range limit increases, the duration and frequency of communication loss between robot 1 and robot 3 decrease. This leads to smaller differences observed between the decentralized and centralized estimates. Estimation errors that are similar to the above figures are observed in the state estimates of the other robots. Note here that we cannot expect the decentralized and centralized estimates to be the same unless a partial checkpoint occurs at the latest timestep (i.e., if a robot communicates with all other robots at the current timestep). In all other cases, the partial checkpoint occurrence

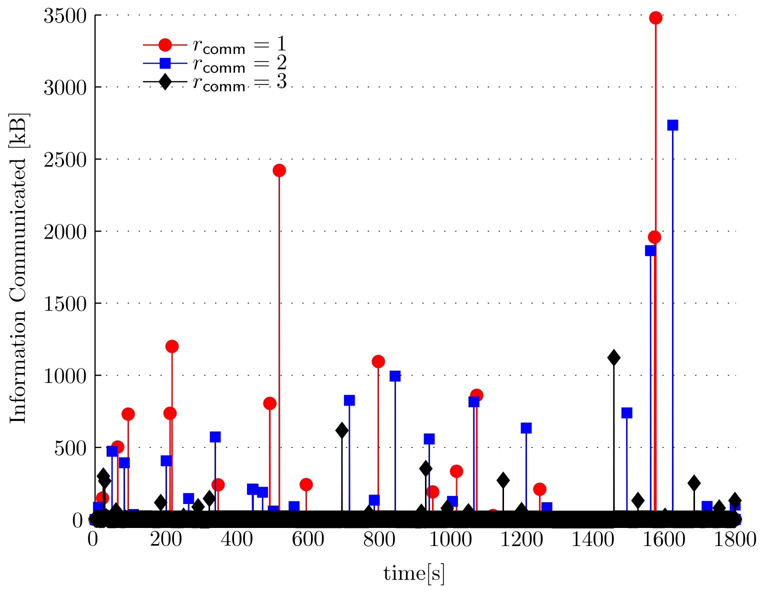
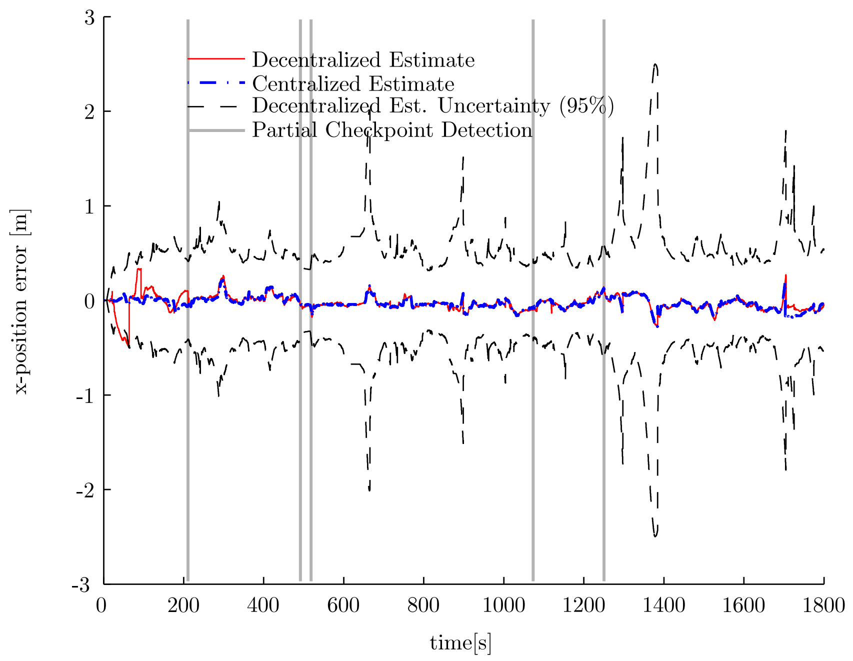


Figure 3.16: The amount of information exchanged with robot 1 in test 1.

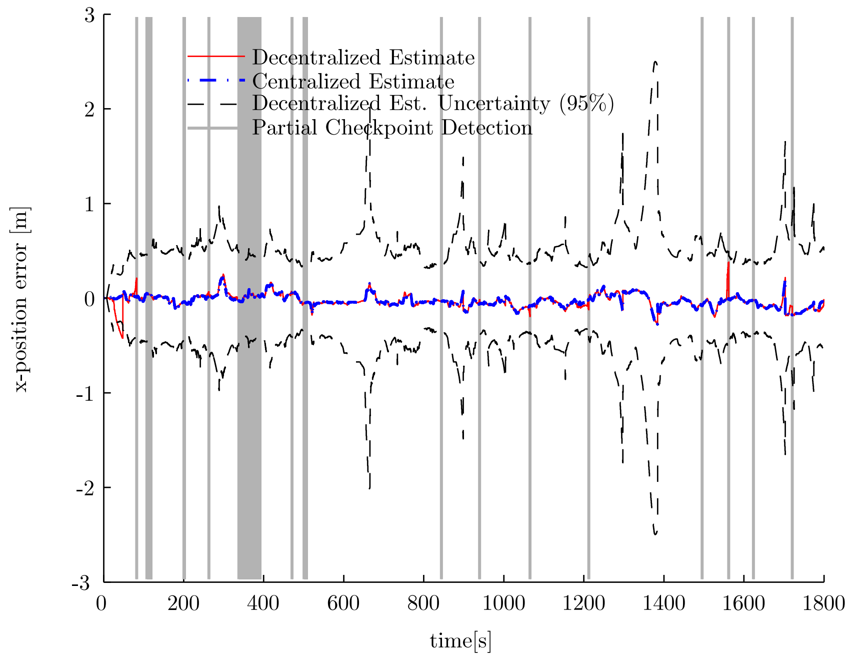
time (i.e., the time of the centralized-equivalent estimate) is delayed, and we need to use a temporary estimate for the current timestep.

We will now turn our attention to looking at the overall results from all the datasets. Again, we tested our decentralized cooperative SLAM algorithm using communication range limits of 1m, 2m, and 3m. Each data point in Fig.3.20 corresponds to either the average memory use by all robots (red circles) or the maximum memory use out of all robots (blue squares) in a particular test, under a particular communication range limit. A line of best fit is also plotted for both average memory use and maximum memory use to illustrate the general trend of the data. We can observe that as communication range decreases (or as the percentage of time that the network is fully connected increases), both average and maximum memory use decrease. This corresponds with the trend observed in Fig. 3.15. To reiterate, this trend is due to the robots applying the Markov property more frequently as communication range limit increases.

Fig. 3.21 shows the average amount of information communicated between robots during each communication interval (every 0.5s), as well as the maximum amount of information communicated out of all robots. Again, each data point corresponds to a particular test and is restricted to a particular communication range limit. The average amount of information communicated remains low regardless of the percentage of time that the network is fully connected. The maximum amount of information communicated increases as the percentage of time that the network is fully connected decreases. The reason for this, as explained previously, is because robots are likely to lose connection

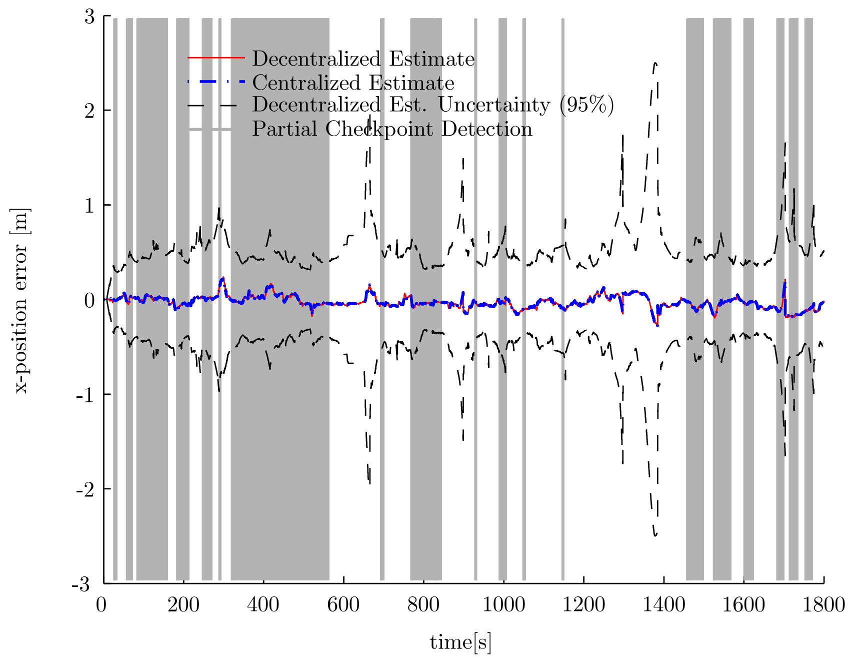


(a) Estimate of its own x-position, $r_{\mathsf{comm}} = 1$.



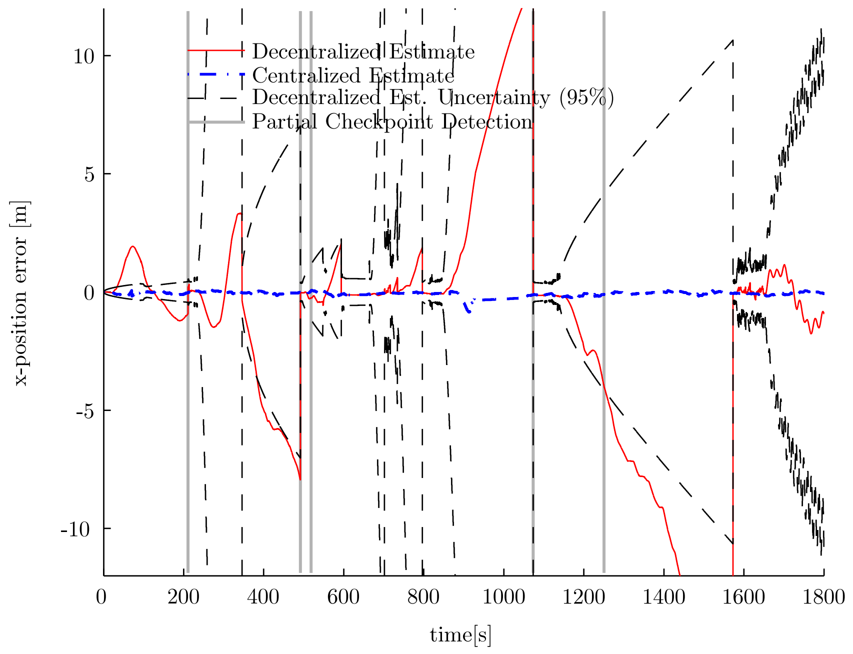
(b) Estimate of its own x-position, $r_{\mathsf{comm}} = 2$.

Figure 3.17: Robot 1's decentralized estimate of its own x-position at various communication range limits.



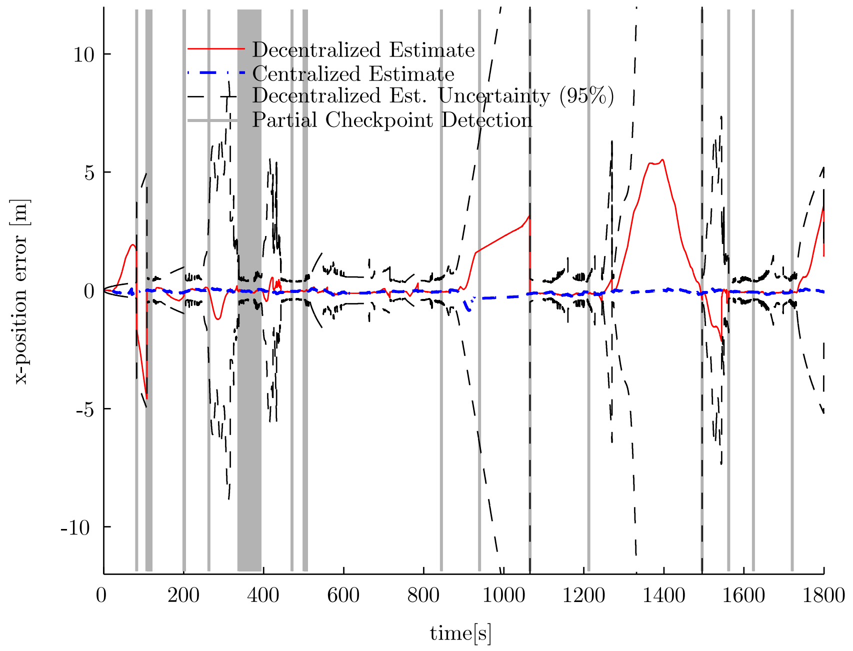
(c) Estimate of its own x-position, $r_{\mathsf{comm}} = 3$.

Figure 3.17: Robot 1's decentralized estimate of its own x-position at various communication range limits.

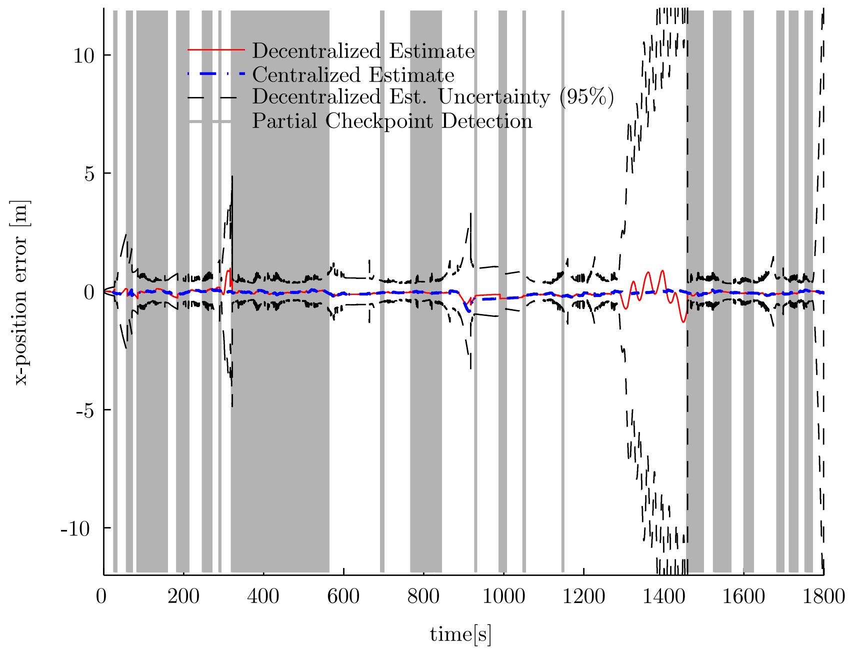


(a) Estimate of robot 3's x-position, $r_{\mathsf{comm}} = 1$.

Figure 3.18: Robot 1's decentralized estimate of another robot's x-position at various communication range limits.

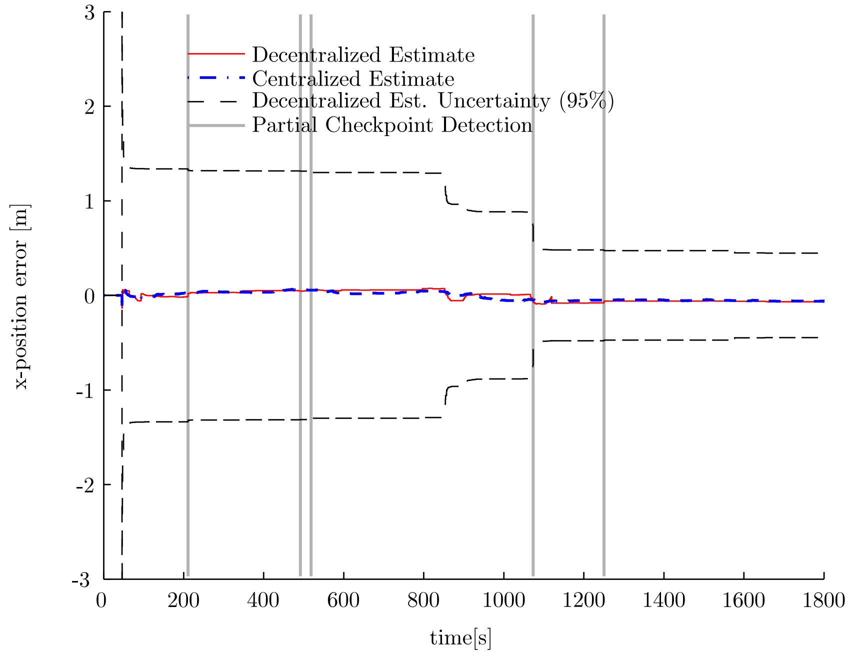


(b) Estimate of robot 3's x-position, $r_{\mathsf{comm}} = 2$.

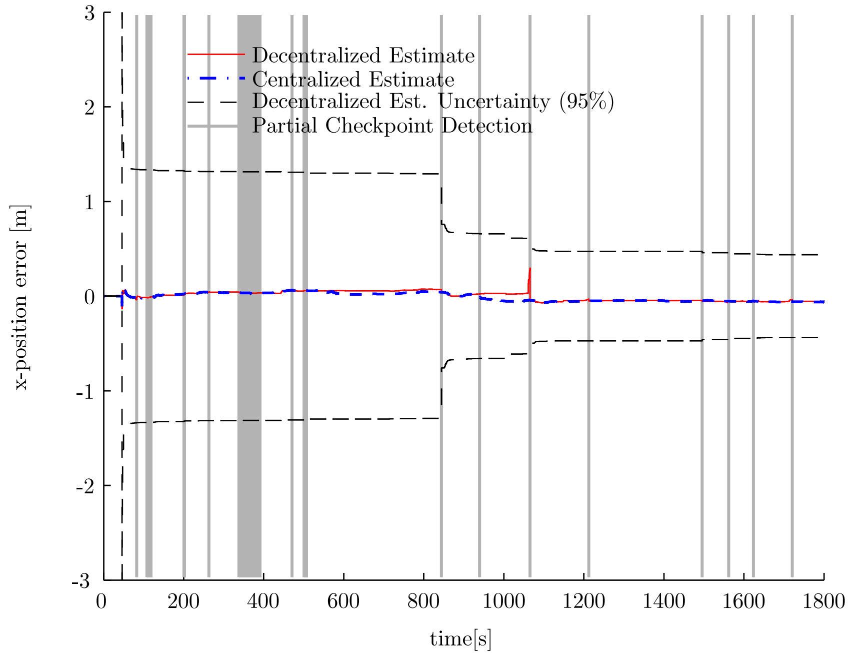


(c) Estimate of robot 3's x-position, $r_{\mathsf{comm}} = 3$.

Figure 3.18: Robot 1's decentralized estimate of another robot's x-position at various communication range limits.

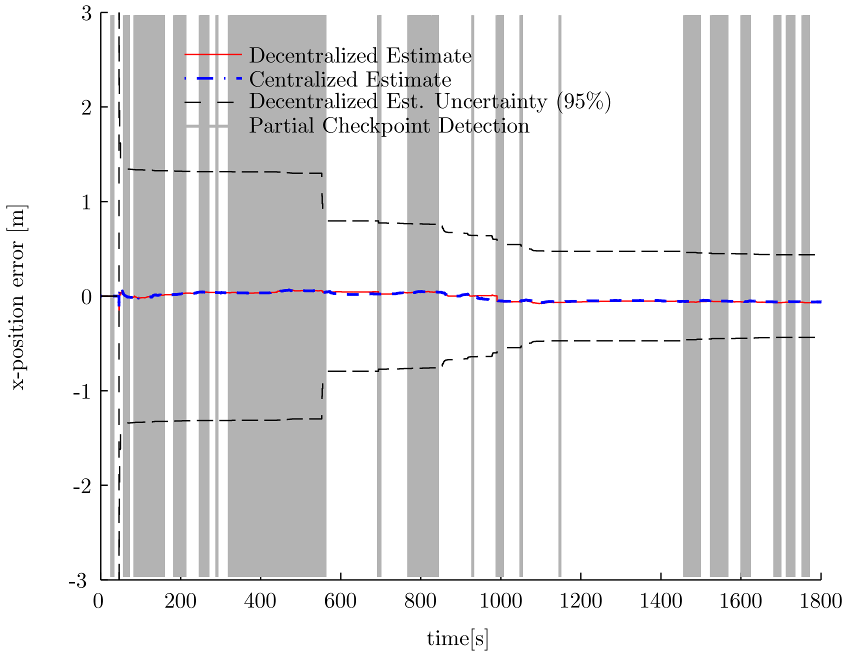


(a) Estimate of a landmark's x-position, $r_{\mathsf{comm}} = 1$



(b) Estimate of a landmark's x-position, $r_{\mathsf{comm}} = 2$

Figure 3.19: Robot 1's decentralized estimate of the x-position of a landmark at various communication range limits.



(c) Estimate of a landmark's x-position, $r_{\mathsf{comm}} = 3$

Figure 3.19: Robot 1's decentralized estimate of the x-position of a landmark at various communication range limits.

with other robots and experience longer duration of disconnection as the communication range limit decreases. These observations are consistent with the results in Fig. 3.16. Although Fig. 3.21 shows that the amount of information communicated can become very high when the communication range limit is low, this is only the case if robots have to update each other with the newest information within one communication interval. In reality, robots are usually within communication range of each other over multiple communication intervals. Hence, the exchange of information could be spread out over this duration.

Next, we will look at the differences between the decentralized and centralized estimates in all 8 tests. Fig. 3.22a-3.22c show the *root-mean-squared* (rms) differences between the decentralized and centralized x-position estimates, $e_{x,\mathsf{rms}}$, y-position estimates, $e_{y,\mathsf{rms}}$, and θ-orientation estimates, $e_{\theta,\mathsf{rms}}$, respectively. In each plot, we also distinguish between robots' estimates of themselves (red circle), robots' estimates of all other teammates (blue square), and robots' estimates of all landmarks (black diamond). Lines of best fit are plotted to provide a sense of the general trends in the data. These trends are consistent with what was observed in the detailed results from test 1 presented earlier.

In general, for a robot's decentralized estimates of its own pose, there is very little difference compared to the centralized estimates. Even as communication range limit de-

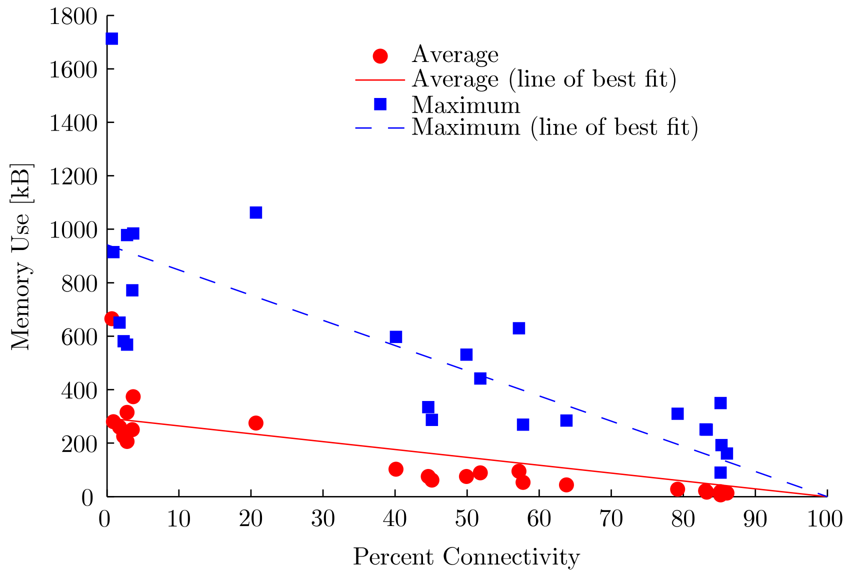


Figure 3.20: Average and maximum memory usage as a function of the percentage of time that the network is fully connected.

creases, the differences between the decentralized and centralized estimates only increase slightly. This is because a robot's own odometry and measurements are always known. The increasing differences between the decentralized and centralized self-estimates with decreasing percent connectivity can be explained by the reduced chance for a robot to obtain measurements from other robots as the communication range limit decreases.

For a robot's estimates of other robots' poses, there is on average a greater difference between the centralized and decentralized estimate compared to the error in its self-estimate. This is because the latest odometry data from other robots are not always available for a robot's current state estimate. Furthermore, this difference becomes larger as the communication range limit decreases since the opportunities for communication between robots also decrease. Note, however, that the current state estimate is temporary and can be updated when more information becomes available. Our decentralized cooperative SLAM algorithm ensures that the centralized-equivalent estimate can be obtained later (when a partial checkpoint is detected).

For landmarks, the average differences between the decentralized and centralized estimates follow a similar trend to the robots' self-estimates. The differences are relatively small because landmarks are static (i.e., they do not have odometry data), and are mainly due to a robot not having the latest measurements from other robots. This is an important point, as it indicates that a robot can retain a good estimate of its own state and the location of all landmarks even when network connectivity is sparse. This is useful to

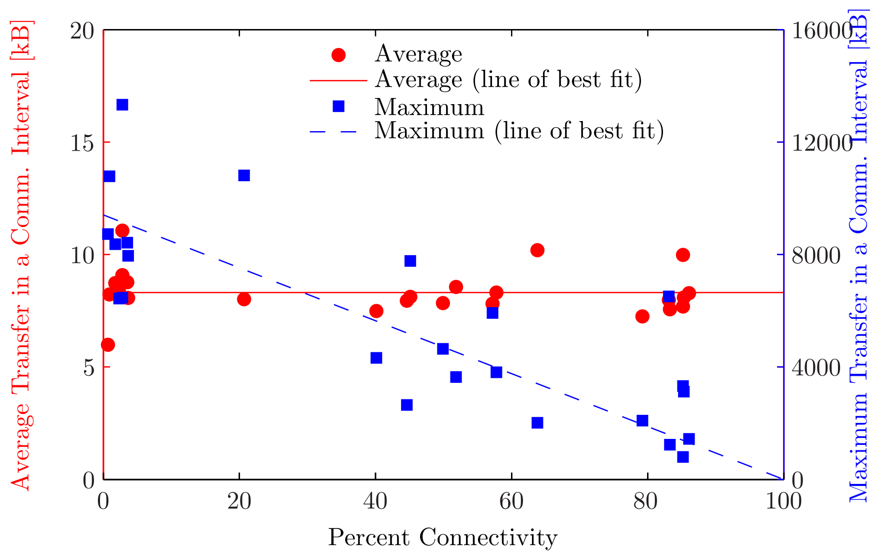


Figure 3.21: Average and maximum amount of information communicate between robots as a function of the percentage of time that the network is fully connected.

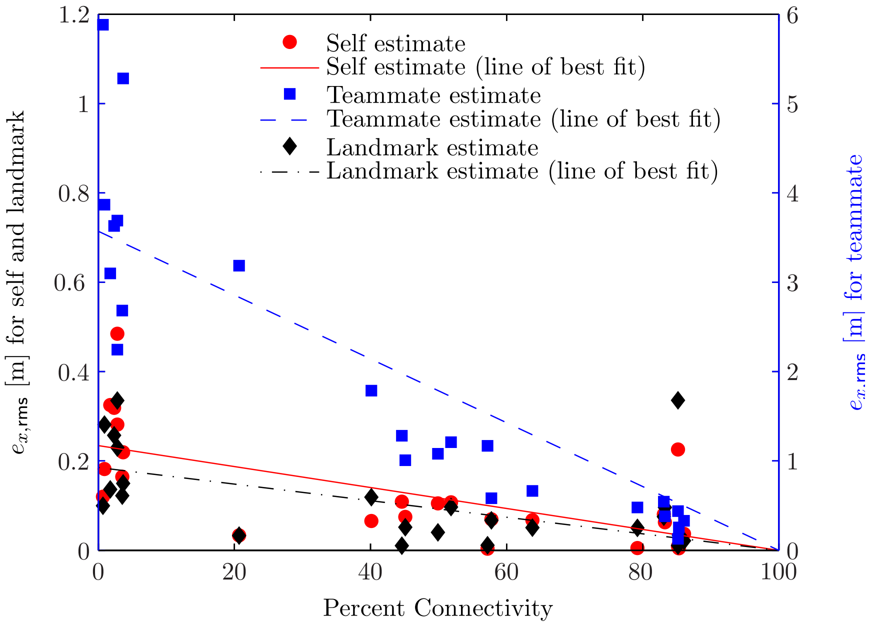
know should robots need to perform path planning in the workspace.

As we have mentioned in Section 2.5.2, computational, storage. and communication complexities still remain the main limitation of our approach in comparison to other cooperative SLAM methods. However, of the methods that can operate in sparsely-communicating networks, our approach is the most accurate in the sense that it is the only one that can produce the centralized-equivalent estimate (even when the network is never fully connected). One additional limitation to our approach is that we require the initial pose estimates to be known (in a common reference frame). Approaches such in [96], [97] and [98] do not have this requirement, and rely on map alignment algorithms for merging estimates when robots encounter each other. To reiterate, these approaches can produce inconsistent estimates.

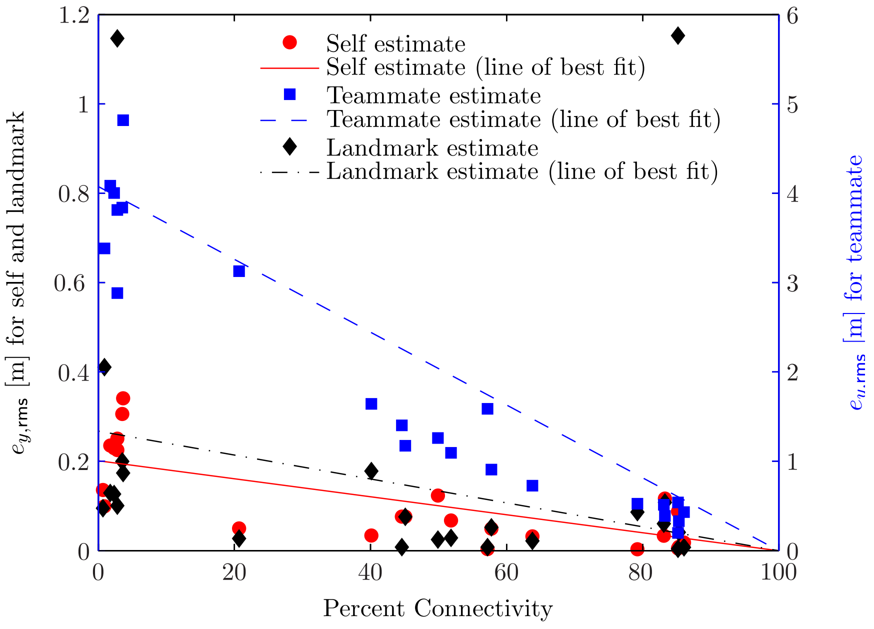
3.7 Summary

In this chapter, we posed the centralized-equivalent decentralized cooperative SLAM problem, where it is necessary for all the robots in a team to obtain centralized-equivalent estimates of known landmark positions and all robot poses, while the communication network is never guaranteed to be fully connected.

In developing our decentralized cooperative SLAM algorithm, we generalized the concepts of a checkpoint and a partial checkpoint, making them applicable to both the decentralized cooperative localization problem, and the decentralized cooperative SLAM

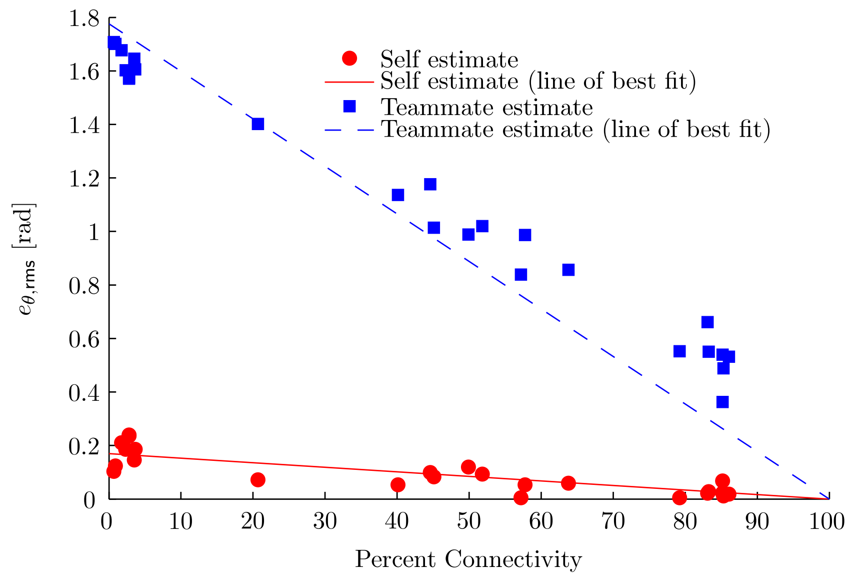


(a) x-position differences.



(b) y-position differences.

Figure 3.22: The rms differences between the decentralized and centralized estimate for a robot's self-estimate, estimates of teammates, and estimates of landmarks, as a function of the percentage of time that the network is fully connected.



(c) θ, orientation, differences.

Figure 3.22: The rms differences between the decentralized and centralized estimate for a robot's self-estimate, estimates of teammates, and estimates of landmarks, as a function of the percentage of time that the network is fully connected.

problem. These events identify when all robots and when a single robot can obtain the centralized-equivalent estimate of the system, respectively. The generalization is important as it allows us to use theorems previously developed for the decentralized cooperative localization problem in Chapter 2. Furthermore, we proved that it is necessary in decentralized cooperative SLAM for each robot to initially know the total number of robots in the network. This requirement on initial knowledge is not necessary for the cooperative localization case but is required in decentralized cooperative SLAM. Although this may seem somewhat restrictive, it is necessary for ensuring that the centralized-equivalent estimate is obtainable.

To validate our decentralized cooperative SLAM algorithm, we created an extensive hardware experiment dataset. Using this, we were able to compare the results from our algorithm against those from a centralized estimator (which we allowed to cheat by ignoring communication constraints), and evaluated the performance of our algorithm under different communication range limits. Our results show how memory usage is limited in our algorithm due to use of the Markov property. The results also show that the centralized-equivalent estimate can always be recovered after a period of poor network connectivity. In terms of estimation accuracy, our overall results show that a robot's estimate of its own pose and the position of landmarks are close to the centralized estimate even with a low percentage of time in which the network is fully connected. Furthermore, the accuracy of a robot's estimate of another robot depends on the percentage of time in which the network is fully connected.

One aspect of SLAM that we only touched lightly upon in this chapter is the problem of data association. That is, determining with which robot or landmark a measurement corresponds with. This task is important as improper data association often leads to a divergence between the estimate and the actual robot and landmark states in most SLAM algorithms (even single robot ones). This can then lead to a unrecoverable failure in the localization and mapping system [107]. In validating our decentralized cooperative SLAM algorithm in this chapter, we made the task of data association easy by allowing robots to use the barcode identification numbers on each robot and landmark. We also claimed that our algorithm will still work if we assume unknown data association. However, the quality of the resulting estimate will depend largely on the data association method used, and how well it does in correctly assigning correspondences. In practice, it is useful to keep track of multiple data association hypotheses [108] as a preventive measure against data association failures (i.e., a bad hypothesis can be discarded and we can continue to perform state estimation using the remaining hypotheses). However, maintaining multiple hypotheses also increases computational complexity (as we are essentially running a separate filter for each hypothesis), and memory use. In the following chapter, we will examine how data association hypotheses can be distributed amongst a team of robots to make better use of the computational resources available in a network of robots.

Chapter 4

Distributed and Decentralized Cooperative Simultaneous Localization and Mapping

Competition has been shown to be useful up to a certain point and no further, but cooperation, which is the thing we must strive for today, begins where competition leaves off.

Franklin D. Roosevelt (1882–1945)
former president of the U.S.A.

In this chapter, we will examine how a robot team can make better use of the computational resources available in performing state estimation[1]. Previously in Chapter 3, we identified that data association (i.e., associating a particular measurement with the correct robot or landmark) is an important aspect in SLAM, and that it is critical for measurements to have the correct correspondences. The consequence of a poor data association is estimator divergence [107], and this type of failure essentially causes a robot to be unable to localize and to have a map that is no longer usable. To reduce the chance of this occurring, multiple estimates using different data association hypotheses can be maintained [108] so that we have the opportunity to discard and replace estimates made with bad data associations. The cost of this, however, is higher computational complexity and memory usage. With a multi-robot system, we have potentially more computational

[1]The work this chapter was presented at the IEEE International Conference on Robotics and Automation 2011 [109].

resources that can be used for maintaining multiple estimates because each robot is equipped with its own computer. In other words, we can potentially distribute the work amongst all the robots. However, this is a complex task under the assumption that the communication network is never guaranteed to be fully connected. At this point, it will be useful to review the following terms:

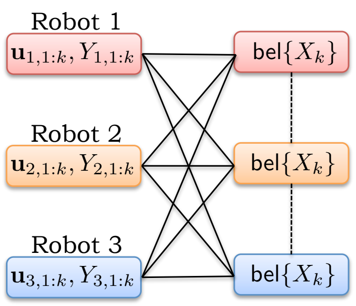
Definition 2: *A* decentralized multi-robot system *is a group of robots in which the computation for a task can be performed by one or more robot in the team.*

Definition 3: *A* distributed multi-robot system *is a group of robots in which the computation for a task is divided amongst the robots in the team.*

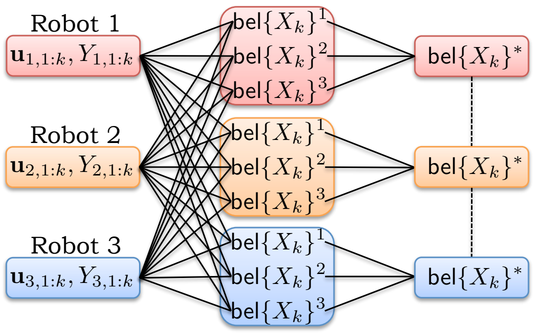
Although there have been numerous publications on cooperative localization, and cooperative SLAM, few have considered a combined decentralized and distributed approach. Previously in Chapter 2 and 3 we reviewed some of the more notable advances on these two problems. These include the work by Roumeliotis and Bekey [55], who performed distributed multi-robot localization by decomposing the EKF into multiple filters that can perform the prediction step of the EKF locally on each robot. More recently, Nerurkar et al. [58] performed cooperative localization using a distributed MAP estimator. Schizas et al. [110] performed distributed estimation based on reduced-dimensionality observations from a sensor network, where the measurements are processed by a central node. Having a fully connected network is a key requirement for the works above. In contrast, we do not make any assumptions on network connectivity. Most closely related to the work in this chapter is the study by Ko et al. [96], who performed multi-robot SLAM in a non-fully connected network. However, the centralized-equivalent estimate cannot be obtained unless the network is fully connected.

Our idea for distributing the computations for multiple estimates with different data association hypotheses is illustrated in Fig. 4.1. In the previous chapters, we assumed known data association. Therefore, robots do not need to maintain multiple estimates as the centralized-equivalent estimates that they obtain are always calculated based on correct correspondences. Hence, as shown in Fig. 4.1a, each robot will calculate their own centralized-equivalent estimate, $\mathsf{bel}\{X_k\}$, when they have received the necessary information from other robots (i.e., when a partial checkpoint exists). Note that it is also possible for a robot to communicate a centralized-equivalent estimate that it has calculated to another robot; in fact, we rely on this to allow robots to apply the Markov property without having to worry what other robots know at the time.

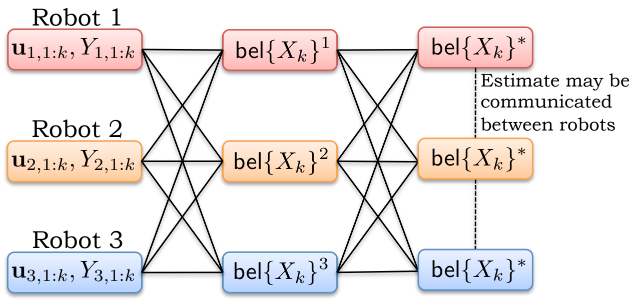
Suppose now that we want to maintain multiple estimates with different data as-


Robot 1
$\mathbf{u}_{1,1:k}, Y_{1,1:k}$
Robot 2
$\mathbf{u}_{2,1:k}, Y_{2,1:k}$
Robot 3
$\mathbf{u}_{3,1:k}, Y_{3,1:k}$
$\mathsf{bel}\{X_k\}$
$\mathsf{bel}\{X_k\}$
$\mathsf{bel}\{X_k\}$

(a) Decentralized approach with known data association



(b) Decentralized and non-distributed approach, maintaining multiple estimates using different data association hypotheses.


Robot 1
$\mathbf{u}_{1,1:k}, Y_{1,1:k}$
Robot 2
$\mathbf{u}_{2,1:k}, Y_{2,1:k}$
Robot 3
$\mathbf{u}_{3,1:k}, Y_{3,1:k}$
$\mathsf{bel}\{X_k\}^1$
$\mathsf{bel}\{X_k\}^2$
$\mathsf{bel}\{X_k\}^3$
$\mathsf{bel}\{X_k\}^*$
Estimate may be communicated between robots

(c) Decentralized and distributed approach, maintaining multiple estimates using different data association hypotheses.

Figure 4.1: Comparing distributed and non-distributed approaches. Same-coloured boxes represent the quantities obtained by a certain robot. Solid lines show the dependencies required in obtaining an estimate. A dashed line indicates that an estimate can be obtained by the sharing of informations. Note that in (b) and (c), we must evaluate, $\mathsf{bel}\{X_k\}^1, \mathsf{bel}\{X_k\}^2, \mathsf{bel}\{X_k\}^3$, which are estimates based on different sets of data association hypotheses. Only after evaluating these intermediate products can we obtain the centralized-equivalent estimate.

sociation hypotheses using our existing decentralized approach. Again, our motivation here is to reduce the chance of estimator divergence and catastrophic failures by relying on more than just one estimate. When a robot acquires the necessary information to calculate the centralized-equivalent estimate (i.e., when a partial checkpoint exists for that robot), it will want to generate a number of different estimates. For illustrative purposes, suppose a robot maintains three different estimates based on different sets of data association hypotheses, shown as $\mathsf{bel}\{X_k\}^1, \mathsf{bel}\{X_k\}^2, \mathsf{bel}\{X_k\}^3$ in Fig. 4.1b. From these estimates, the robot will select the best estimate, based on some defined metric, as the centralized-equivalent estimate, $\mathsf{bel}\{X_k\}^*$. Since each robot needs to repeat this when they detect a partial checkpoint, there will be redundant calculations, and there will exist multiple copies of the same estimates on each robot.

To reduce the amount of redundancy, we want to have the calculations of estimates with different data association hypotheses distributed amongst the team of robots, as shown in Fig. 4.1c. When a partial checkpoint is detected by a robot, it will select a subset of estimates to calculate. For illustrative purposes, in Fig. 4.1c, each robot calculates one estimate, which is based on a different set of data association hypotheses than the estimates produced by its teammates. When each robot has completed their share of the calculations, they will share their estimates with each other. From those, they will select one (based on some defined metric) as the centralized-equivalent estimate.

With the distributed approach in Fig. 4.1c, robots need to 'synchronize' twice to obtain the centralized-equivalent estimate. The first time is to acquire the information to perform the calculations for estimates based on different data association hypotheses, and the second time is to share their calculated estimates so that one can be selected as the centralized-equivalent estimate. Hence, in comparison to the non-distributed approach (in Fig. 4.1b, where robots only need to synchronize once), we are reducing redundancy by adding a second layer of information synchronization.

The notions of a checkpoint and a partial checkpoint have already been defined from previous chapters for information synchronization purposes. The existence of a partial checkpoint enables a robot to know when it has gathered the information required to calculate a centralized-equivalent estimate. In Chapter 2, we used the notions of checkpoint and partial checkpoint for the decentralized cooperative localization problem. Then in Chapter 3, we extended the application of these concepts to the decentralized cooperative SLAM problem. In this chapter, we want to again use the notion of a checkpoint and a partial checkpoint and examine how they can facilitate the distribution of estimate calculations based on different data association hypotheses. However, as noted above, we need a second layer of synchronization to produce the centralized-equivalent estimate. Hence we will introduce (later in this chapter) a couple of new notions, which are similar to a checkpoint and a partial checkpoint. Following our previous development approach, we will first examine how the evaluation of multiple estimates based on different data association hypotheses can be distributed in a network from a global perspective. Then we will do the same from the local perspective of a robot, which will lead us to the development of our distributed and decentralized cooperative SLAM algorithm. This algorithm (which we will present in Section 4.3.1), maintains the property of allowing all robots to obtain the centralized-equivalent estimate whenever possible, while assuming that the network for communication is never guaranteed to be fully connected. In such a sparse network, a robot may not always have the latest odometry and measurements from other robots. Our approach allows robots to obtain a temporary (localization and map) estimate at the current timestep using information available locally, but we also ensure that the centralized-equivalent estimate can always be recovered by all robots at a later time. Additionally, we again do not require a robot to keep track of what other robots know when it applies the Markov property to discard past information. For validating this new algorithm, we will use the hardware experiment dataset that we created for validating the decentralized cooperative SLAM algorithm, and assume unknown data associations for the measurements. As part of the evaluation, we will compare the difference between a distributed, and a non-distributed approach. Again, we are motivated by the better utilization of computational resources available in the network of robots.

4.1 Mathematical Formulation

We will use the same general system model for our multi-robot system as in Chapter 3:

$$\begin{aligned}
\mathbf{x}_{i,k} &= \mathbf{g}\left(\mathbf{x}_{i,k-1}, \mathbf{u}_{i,k}, \epsilon_k\right), \ \forall i \in N \\
\mathbf{x}_{m,k} &= \mathbf{x}_{m,k}, \ \forall m \in M \\
\mathbf{y}_{i,k}^{j,i} &= \mathbf{h}\left(\mathbf{x}_{i,k}, \mathbf{x}_{j,k}, \delta_k\right), \ \forall j \in N, \ d_k^{j,i} \leq r_{\mathsf{obs}} \\
\mathbf{y}_{i,k}^{m,i} &= \mathbf{h_m}\left(\mathbf{x}_{i,k}, \mathbf{x}_{m,k}, \psi_k\right), \ \forall m \in M, \ d_k^{m,i} \leq r_{\mathsf{obs}}
\end{aligned}$$

where for timestep k:

$\mathbf{x}_{i,k}$ $(i \in N)$ represents the state (pose) of robot i

$\mathbf{x}_{m,k}$ $(m \in M)$ represents the state (position) of the stationary landmark m

$\mathbf{u}_{i,k}$ represents the odometry information of robot i

$\mathbf{g}(\cdot)$ is the state transition function for the robots

ϵ_k represents the process noise

$\mathbf{y}_{i,k}^{j,i}$ represents the measurement of robot j with respect to robot i

$\mathbf{y}_{i,k}^{m,i}$ represents the measurement of landmark m with respect to robot i

$\mathbf{h}(\cdot)$ is the robot measurement function

$\mathbf{h_m}(\cdot)$ is the landmark measurement function

δ_k is the robot measurement noise

ψ_k is the landmark measurement noise

$d_k^{j,i}$ is the distance between robot i and robot j

$d_k^{m,i}$ is the distance between robot i and landmark m

r_{obs} is the measurement range limit

We will also be using the sets X_k, and $Y_{i,k}$ in our mathematical development. To review,

$$X_k = \{\mathbf{x}_{i,k}, \mathbf{x}_{m,k} | i \in N, m \in M_k\},$$

represents the set of all states known to exist at timestep k, where M_k is the set of landmarks that has been observed by at least one robot up to time k.

$$Y_{i,k} = \{\mathbf{y}_{i,k}^{j,i}, \mathbf{y}_{i,k}^{m,i} | j \in N, m \in M, d_k^{j,i} \leq r_{\mathsf{obs}}, d_k^{m,i} \leq r_{\mathsf{obs}}\}$$

represents the set of measurements from robot i to all robots and landmarks within observation range. For communication, we can again assume that either of the modes illustrated in Fig. 2.1 can be used. That is, we can assume that a robot can communicate with any robot with which it is currently connected, or we can assume that a robot can only communicate with other robots within its own communication range.

Previously, we have provided an expression for the centralized belief over X_k. Here, we will define the centralized belief over a time interval $k \in [k_1, \ldots, k_2]$, where $0 \leq k_1 \leq k_2$, which is represented by a probability density function, $p(\cdot)$:

$$\mathsf{bel}\left(X_{k_1:k_2}\right) := p\left(X_{k_1:k_2} | \mathsf{bel}\left(X_0\right), \{\mathbf{u}_{i,1:k_2}, Y_{i,1:k_2} | i \in N\}\right),$$

which is conditioned on the initial belief, $\mathsf{bel}\left(X_0\right)$, past odometry data, and past range and bearing measurements. Note that we can obtain the belief over one timestep, $\mathsf{bel}\left(X_k\right)$, if we let $k_1 = k_2 = k$. The reason for expressing the belief over a time interval will become apparent as we develop our distributed and decentralized cooperative SLAM algorithm. As a hint, this is related to the second information synchronization event that needs to occur with a distributed approach.

We will continue to use the knowledge set, $S_{i,k}$, to keep track of the information available to each robot. The update rules for the knowledge sets, as defined by (2.3) and (2.4) also remain the same as before. As we have established in Chapter 3, in decentralized cooperative SLAM we need to ensure that all robots initially know the total number of robots in the team. This is necessary to ensure that all robots can obtain the centralized-equivalent estimate. Hence, a robot will detect partial checkpoints based on all the robots in the team.

If we apply the Markov property (to allow for recursive state estimation, reduced computational cost, and memory usage), we can express the centralized belief as

$$p\left(X_{k_1:k_2} | \mathsf{bel}\left(X_{k_1-1}\right), \{\mathbf{u}_{i,k_1:k_2}, Y_{i,k_1:k_2} | i \in N\}\right).$$

Note here that the centralized belief is still expressed over a time interval. Since we are operating in a sparsely-communicating robot network, the Markov property can only be applied once a robot obtains sufficient information regarding other robots through communication (i.e., when the robot can calculate the centralized-equivalent estimate). Furthermore, each robot must ensure that other robots will no longer require any past information that will be discarded when applying the Markov property. In decentralized cooperative SLAM, a robot can apply the Markov property when it detects the existence of a checkpoint. For distributed and decentralized cooperative SLAM, we need to make some changes and introduce some additional concepts to handle the need for double synchronization, as illustrated in Fig. 4.1.

With the non-distributed decentralized cooperative SLAM algorithm, each robot is responsible for obtaining their own estimate of the system. This inevitably causes redundancy as robots make the same calculations for the same estimate (Fig. 4.1a). This redundancy becomes more profound if we want to maintain multiple estimates based on different data association hypotheses (Fig. 4.1b). It is therefore desirable and computationally advantageous to reduce this redundancy by distributing the computation amongst the robots in the team. We want to enable robots to calculate state estimates based on different data association hypotheses, which can be evaluated in parallel across the robot network (Fig. 4.1c). For instance, if we use a PF [53], each robot can contribute by handling the weight calculations for a subset of particles (i.e., particles may have different data association hypotheses). We define the computations performed by a robot towards a centralized-equivalent state estimate as follows:

Definition 6: *The* distributed contribution, $\mathsf{bel}\left(X_{k_1:k_2}\right)^i$*, is the computation performed by robot* i *for the centralized-equivalent estimate* $\mathsf{bel}\left(X_{k_1:k_2}\right)$*, and it is dependent on the quantities* $\left\{\mathsf{bel}\left(X_{k_1-1}\right), \mathbf{u}_{i,k_1:k_2}, Y_{i,k_1:k_2} | i \in N\right\}$.

Note here that each robot's distributed contribution requires the same input (i.e., the same previous estimate, with the same odometry and measurement data). However, a random selection of data association hypotheses are used for each distributed contribution. Hence, two distributed contributions are different unless the exact same set of data association hypotheses are selected. Note also that the same inputs that are used to calculate a distributed contribution allows us to obtain the centralized-equivalent estimate, $\mathsf{bel}\left(X_{k_2}\right)$, in our previous non-distributed approach. From all robots' distributed contributions, we want to be able to select one (which we think did the best job in assigning proper measurement correspondences) as the centralized-equivalent estimate. For this purpose, we implicitly associate a metric, s^i, with each distributed contribution:

$$\mathsf{bel}\left(X_{k_1:k_2}\right)^i \leftarrow \{\mathsf{bel}\left(X_{k_1:k_2}\right)^i, s^i\}.$$

The metrics are used in determining how distributed contributions are used to produce a *consensus estimate*:

Definition 7: *The* consensus estimate, $\mathsf{bel}\left(X_{k_1:k_2}\right)^*$, *is equivalent to the centralized-equivalent estimate* $\mathsf{bel}\left(X_{k_1:k_2}\right)$, *and is only produced by the distributed contributions from all robots through an arbitration function* $f^a(\cdot)$ *that is known to all robots a priori:*

$$f^a : \{\mathsf{bel}\left(X_{k_1:k_2}\right)^i | i \in N\} \mapsto \mathsf{bel}\left(X_{k_1:k_2}\right)^* \tag{4.1}$$

As an example, the distributed contribution $\mathsf{bel}\left(X_{k_1:k_2}\right)^i$ may represent the estimate produced by robot i using a certain set of data association hypotheses, with the metric $\mathbf{s}^i$ representing the product of the likelihoods of all the measurements used in the estimate. The arbitration function, $f^a(\cdot)$, can be defined such that when the distributed contributions from all robots have been obtained, the one with the highest metric is selected as the consensus estimate, $\mathsf{bel}\left(X_{k_1:k_2}\right)^*$.

Note that the arbitration function is general, and can encompass the case where distributed computation is unnecessary (i.e., when all robots consider the same data association hypothesis as in Fig. 4.1a, and a robot's distributed contribution is just interpreted as the centralized-equivalent estimate):

$$f^a(\mathsf{bel}\left(X_{k_1:k_2}\right)^i) = \mathsf{bel}\left(X_{k_1:k_2}\right)^i = \mathsf{bel}\left(X_{k_1:k_2}\right)^*, i \in N.$$

Our objective in this chapter is to determine how cooperative SLAM can be solved as a recursive filtering problem in a distributed and decentralized fashion. Furthermore, this needs to be done in a dynamic and sparsely-communicating robot network that is never guaranteed to be fully connected. Additionally, we want all robots to obtain the (centralized-equivalent) consensus estimate (of all robots and known landmarks) whenever possible.

4.2 Operating in a Dynamic and Sparse Network

In the previous chapter, we presented a solution for decentralized cooperative SLAM where robots are able to compute the centralized-equivalent estimate in a non-distributed manner while operating in a network that is never fully connected. Distributing the solution in a sparse network creates a more complex situation because not only do robots have to detect when they have sufficient information to calculate their distributed contributions, they also need to reach a consensus (centralized-equivalent) estimate, $\mathsf{bel}\left(X_k\right)^*$ (recall the double synchronization requirement shown in Fig. 4.1c). In our theoretical development, we will take a similar approach to Chapter 2 and Chapter 3, and first examine our robot network from a global perspective (i.e., that of an outside observer). We will then examine how robots observe the network locally.

4.2.1 The Global Perspective

It is important to know when all robots are able to calculate a distributed contribution, $\mathsf{bel}\left(X_k\right)^i$, as well as the consensus estimate, $\mathsf{bel}\left(X_k\right)^*$. For the purpose of determining when distributed contributions are obtainable by all robots, we need to first review the definition of a checkpoint.

A *checkpoint*, $C(k_c, k_e)$, is an event that occurs at the checkpoint time, k_c, that first comes into existence at k_e, in which the knowledge set of each robot i contains for all j:

1. the previous state estimate, $\mathsf{bel}\left(\mathbf{x}_{j,k_{s,j}}\right)^*$, $k_{s,j} \leq k_c$,
2. all the odometry and measurement data of robot j from timestep $k_{s,j}$ to k_c.

Equivalently, a checkpoint occurs at timestep k_c when $S_{i,k_e} \supseteq \left\{S_{j,k_c} | j \in N\right\}, \forall i \in N$.

The concept of a checkpoint was used in Chapter 2 for the decentralized cooperative localization problem, and was generalized in Chapter 3 for the decentralized cooperative SLAM problem. The existence of $C(k_c, k_e)$ indicates that all robots have the required information in their knowledge sets to individually calculate the centralized-equivalent estimate, $\mathsf{bel}\left(X_{k_c}\right)$, using the same set of data association hypotheses. We now want to use the concept of a checkpoint to indicate when all robots are able to obtain a distributed contribution in a distributed approach to cooperative SLAM. That is, we want to show that when a checkpoint exists, each robot's knowledge up to timestep k_c can be used in potentially random-seeded processes (e.g., such as *Data-Aligned Rigidity-Constrained Exhaustive Search* (DARCES) [111]) performed on each robot to obtain different distributed contributions (and the associated metric, s^i). Once robots have all the distributed contributions, we can then form a (centralized-equivalent) consensus estimate using the arbitration function. To formalize the first part of this (i.e., the first of the two synchronization requirements), we have the following theorem.

Theorem 6.1: *Suppose that the knowledge sets of all robots contain a (centralized-*

equivalent) consensus estimate for some previous timestep k_s (i.e., $S_{i,k_e} \supseteq \{\mathsf{bel}\left(X_{k_s}\right)^*\}, \forall i \in N$). In performing distributed estimation, checkpoint $C(k_c, k_e)$ exists, with $k_s < k_c \leq k_e$, if and only all robots can obtain a distributed contribution, $\mathsf{bel}\left(X_{k_s+1:k_c}\right)^i$, at k_e.

Proof. First assume that $C(k_c, k_e)$ exists. This implies

$$S_{i,k_e} \supseteq \{\mathsf{bel}\left(X_{k_s}\right)^*, \mathbf{u}_{j,k_s+1:k_c}, Y_{j,k_s+1:k_c} | j \in N\}, \forall i \in N.$$

Using Definition 6, we know that all robots can calculate $\mathsf{bel}\left(X_{k_s+1:k_c}\right)^i$.

The reverse argument is also true if we first assume that $\mathsf{bel}\left(X_{k_s+1:k_c}\right)^i$ is obtainable by all robots. This implies that S_{i,k_e} contains the quantities required for $C(k_c, k_e)$ to exist for all robots $i \in N$, according to Definition 4. For an illustrated example of checkpoint existence for distributed and decentralized cooperative SLAM, look ahead to Fig. 4.2. □

Note that when $C(k_c, k_e)$ exists, every robot is able to obtain the distributed contribution over the interval $k_s + 1 : k_c$. We need to remember that the distributed contribution is obtained after communication at the current timestep. Therefore, we must wait for a later timestep to communicate distributed contributions to other robots. We made the estimate time interval explicit because we need to keep estimates for this entire interval (and not just for the latest timestep). The reason for this, as we shall see in Section 4.2.2, is for the second synchronization that needs to occur to generate the consensus estimate. Once we obtain the consensus estimate, we will be able to apply the Markov property to discard used information. To show when the consensus estimate is obtainable by all robots (i.e. the second synchronization), we define a *distributed checkpoint* as follows:

Definition 8: *A* distributed checkpoint, *$C^d(k_s, k_c, k_d)$, where $k_s \leq k_c < k_d$, is an event that first comes into existence at k_d, when $\forall i \in N$, $S_{i,k_d} \supseteq \{\mathsf{bel}\left(X_{k_s:k_c}\right)^j | j \in N\}$, or $S_{i,k_d} \supseteq \{\mathsf{bel}\left(X_{k_s:k_c}\right)^*\}$.*[2]

When each robot has the distributed contributions from all robots, the arbitration function, $f^a(\cdot)$, can be used to obtain the consensus estimate.

Theorem 6.2: *A distributed checkpoint $C^d(k_s, k_c, k_d)$ exists if and only if all robots $i \in N$ can obtain the consensus estimate $\mathsf{bel}\left(X_{k_s:k_c}\right)^*$ at k_d.*

[2]Note that k_c must be less than k_d because after the robots have calculated their distributed contributions, they must wait for the next communication interval to exchange the distributed contributions with each other.

Proof. First, assume that $C^d(k_s, k_c, k_d)$ exists. This implies that

$$S_{i,k_d} \supseteq \begin{cases} \{\mathsf{bel}\left(X_{k_s:k_c}\right)^j \mid j \in N\} \text{ or} \\ \mathsf{bel}\left(X_{k_s:k_c}\right)^* \end{cases}, \forall i \in N.$$

That is, if robots do not already have the consensus estimate, $\mathsf{bel}\left(X_{k_s:k_c}\right)^*$, they can use the arbitration function (4.1) to obtain the consensus estimate from the distributed contributions from all robot.

The reverse argument is also true if we first assume that all robots can obtain $\mathsf{bel}\left(X_{k_s:k_c}\right)^*$. The consensus estimate can be obtained by each robot through direct communication with another robot (i.e., $S_{i,k_d} \supseteq \mathsf{bel}\left(X_{k_s:k_c}\right)^*$). It can also be obtained by the using the arbitration function, which requires

$$S_{i,k_d} \supseteq \left\{\mathsf{bel}\left(X_{k_s:k_c}\right)^j \mid j \in N\right\}.$$

This implies that the distributed checkpoint, $C^d(k_s, k_c, k_d)$, exists at k_d according to Definition 8. □

To review, in order for all robots to obtain the (centralized-equivalent) consensus estimate in distributed and decentralized cooperative SLAM, we must first detect the existence of a checkpoint, which allows each robot to calculate a distributed contribution (i.e., the first synchronization event). Then, we must detect the existence of a distributed checkpoint, which enables robots to either calculate the consensus estimate by applying the arbitration function, or to receive the consensus estimate from another robot through communication (i.e., the second synchronization event). Fig. 4.2 provides a simple example that illustrates this process. However, what we have shown so far is from the perspective of an outside observer of the robot network (i.e., all communication at all timesteps can be observed). Robots do not have this benefit of knowing all the communications that have occurred between other robots. Analogous to our previous approach in the theoretical development of the solutions to decentralized cooperative localization and decentralized cooperative SLAM, we will now examine how distributed estimation can be performed by only considering the local knowledge of a robot.

4.2.2 The Local Perspective

Previously, we reused the concept of a checkpoint, and introduced the new concept of a distributed checkpoint, which applies to all robots in the system, and require an outside

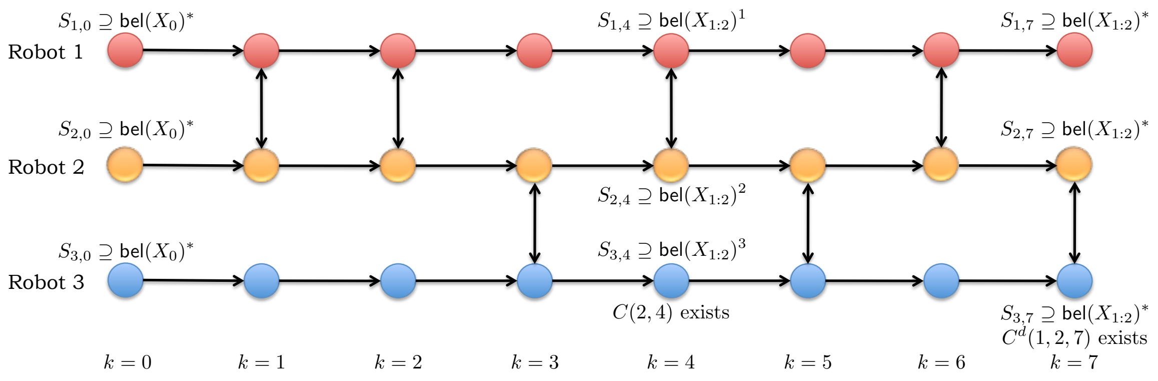

$S_{1,0} \supseteq \mathsf{bel}(X_0)^*$
$S_{1,4} \supseteq \mathsf{bel}(X_{1:2})^1$
$S_{1,7} \supseteq \mathsf{bel}(X_{1:2})^*$
Robot 1
$S_{2,0} \supseteq \mathsf{bel}(X_0)^*$
$S_{2,7} \supseteq \mathsf{bel}(X_{1:2})^*$
Robot 2
$S_{2,4} \supseteq \mathsf{bel}(X_{1:2})^2$
$S_{3,0} \supseteq \mathsf{bel}(X_0)^*$
$S_{3,4} \supseteq \mathsf{bel}(X_{1:2})^3$
Robot 3
$C(2,4)$ exists
$S_{3,7} \supseteq \mathsf{bel}(X_{1:2})^*$
$C^d(1,2,7)$ exists
$k=0$
$k=1$
$k=2$
$k=3$
$k=4$
$k=5$
$k=6$
$k=7$

Figure 4.2: An information flow graph showing the existence of a checkpoint and a distributed checkpoint. Assume that all robots initially have $\mathsf{bel}\,(X_0)^*$ in their knowledge sets. At $k = 4$, All robots are able to obtain distributed contributions for timesteps 1 to 2. Hence, the checkpoint $C(2,4)$ exists at $k = 4$ (i.e., we can find a path on the graph from all the nodes at $k = 2$ to all the nodes at $k = 3$). Note that the distributed contributions are obtained after communication. Therefore, they cannot be communicated to another robot at the same timestep in which they are calculated. At $k = 7$, each robot is able to obtain the distributed contributions from all robots, and calculate the consensus estimate by using the arbitration function. Therefore, $C^d(1,2,7)$ exists.

observer to detect their existence. We will now introduce the corresponding events of a partial checkpoint and a *distributed partial checkpoint* in distributed and decentralized cooperative SLAM for a single robot. We will also explain in this section why it is necessary to maintain distributed contributions over a time interval. Furthermore, as part of the the second synchronization routine in the distributed approach to obtain the consensus estimate, we will introduce the consensus rule, which must be executed on all robots. Finally, we will show how robots can apply the Markov property without considering the knowledge of other robots (similar to what we showed in Chapter 2 for the decentralized cooperative localization problem). First, we should review the definition of a partial checkpoint.

A partial checkpoint, $C_p(k_{c,i}, k_{e,i})$, is an event that occurs at the partial checkpoint time, $k_{c,i}$, that first comes into existence at $k_{e,i}$, in which the knowledge set of robot i contains for all j:

1. the previous state estimate, $\mathsf{bel}\left(\mathbf{x}_{j,k_{s,j}}\right)^*$, $k_{s,j} \leq k_c$,
2. all the odometry and measurement data of robot j from timestep $k_{s,j}$ to k_c.

Equivalently, a partial checkpoint occurs for robot i at timestep k_c when $S_{i,k_e} \supseteq \{S_{j,k_c} | \forall j\}$.

Using this definition, we will show that the existence of a partial checkpoint for a robot corresponds to when that robot can obtain a distributed contribution at the partial checkpoint occurrence time (i.e., this is the first of two synchronizations that happen locally on a robot).

Theorem 7.1: *Suppose that the knowledge set of robot i contains a consensus estimate for some previous timestep k_s (i.e., $S_{i,k_{e,i}} \supseteq \{\mathsf{bel}\,(X_{k_s})^*\}$). In performing distributed estimation, partial checkpoint $C_p(k_{c,i}, k_{e.i})$ exists, with $k_s < k_{c,i} \leq k_{e,i}$, if and only if robot i can obtain the distributed contribution $\mathsf{bel}\left(X_{k_s+1:k_{c,i}}\right)^i$ at $k_{e,i}$.*

Proof. First assume that $C_p(k_{c,i}, k_{e.i})$ exists. This implies

$$S_{i,k_{e,i}} \supseteq \{\mathsf{bel}\,(X_{k_s})^*, \mathbf{u}_{j,k_s+1:k_c}, Y_{j,k_s+1:k_c} \forall j \in N\}.$$

Using Definition 6, we know that robot i can calculate $\mathsf{bel}\left(X_{k_s+1:k_{c,i}}\right)^i$. Now if we first assume that $\mathsf{bel}\left(X_{k_s+1:k_{c,i}}\right)^i$ is obtainable by robot i. This implies that $S_{i,k_{e,i}}$ contains the quantities required for $C_p(k_{c,i}, k_{e.i})$ to exist for robot i, according to Definition 5. □

Once again, at the existence times for these partial checkpoints, the corresponding robot is able to use the information in its knowledge set to calculated a distributed contribution (i.e., an estimate based on a self-chosen set of data association hypotheses). For an examples for partial checkpoint existence, look ahead to Fig. 4.2.

Having completed its share of the work for determining the consensus estimate, a robot now must wait for the distributed contributions from all the other robots, so it can apply the arbitration function, $f^a(\cdot)$, to obtain the consensus estimate (i.e., the second synchronization event that happens locally on a robot). For this, we define the following:

Definition 9: *A* distributed partial checkpoint, *$C_p^d(k_{s,i}, k_{c,i}, k_{d,i})$, where $k_{s,i} \leq k_{c,i} < k_{d,i}$, is an event that first comes into existence at $k_{d,i}$, when $S_{i,k_{d,i}} \supseteq \{\mathsf{bel}\left(X_{k_{s,i},k_{c,i}}\right)^j, \forall j \in N\}$, or $S_{i,k_{d,i}} \supseteq \{\mathsf{bel}\left(X_{k_{s,i},k_{c,i}}\right)^*\}$.*

For the existence of a distributed partial checkpoint, we have the following theorem:

Theorem 7.2: *A distributed partial checkpoint $C_p^d(k_{s,i}, k_{c,i}, k_{d,i})$ exists for robot i if and only if robot i can obtain the consensus estimate $\mathsf{bel}\left(X_{k_{s,i}:k_{c,i}}\right)^*$ at $k_{d,i}$.*

Proof. First, assume that $C_p^d(k_{s,i}, k_{c,i}, k_{d,i})$ exists. This implies that

$$S_{i,k_d} \supseteq \left\{ \begin{array}{l} \{\mathsf{bel}\left(X_{k_{s,i}:k_{c,i}}\right)^j | j \in N\} \text{ or} \\ \mathsf{bel}\left(X_{k_{s,i}:k_{c,i}}\right)^* \end{array} \right. .$$

If robot i does not already have the consensus estimate, $\mathsf{bel}\left(X_{k_{s,i}:k_{c,i}}\right)^*$, it can use the arbitration function in equation (4.1) to obtain the consensus estimate from the distributed contributions from all robots.

The reverse argument is also true if we first assume that robot i can obtain $\mathsf{bel}\left(X_{k_{s,i}:k_{c,i}}\right)^*$. This consensus estimate can be obtained by robot i through direct communication with another robot (i.e., $S_{i,k_d} \supseteq \mathsf{bel}\left(X_{k_{s,i}:k_{c,i}}\right)^*$). It can also be obtained by using the arbitration function, which requires

$$S_{i,k_{d,i}} \supseteq \left\{ \mathsf{bel}\left(X_{k_{s,i}:k_{c,i}}\right)^j \middle| \, j \in N \right\}.$$

This implies that the distributed checkpoint, $C_p^d(k_{s,i}, k_{c,i}, k_{d,i})$, exists at $k_{d,i}$ according to Definition 9. □

Look ahead to Fig. 4.2 again for examples of distributed partial checkpoint existence. Not only does the existence of a distributed partial checkpoint for a robot indicate when that robot can obtain the consensus estimate of the system, but it is also an indication of when the Markov property can be applied. However, before we detect distributed partial checkpoints, we need to consider the following. The existence of a partial checkpoint is based on the local knowledge of a robot (which can be different from that of another robot for the same timestep). Therefore, robots may have distributed contributions that do not have matching time intervals due to different partial checkpoint occurrence times. According to our definition of a distributed partial checkpoint, we must obtain the distributed contributions that span the same time interval from all robots before the consensus estimate can be generated. Intuitively, we need to do this to ensure that we make a fair comparison when we apply the arbitration function (i.e., we are not comparing estimates from different timesteps). Hence, some distributed contributions may need to be truncated before the arbitration function is used to produce the consensus estimate. This is the reason (which we have refrained from explaining in detail earlier) for maintaining distributed contributions over a time interval. More formally, to resolve this time interval mismatch problem, we will introduce the following rule on all robots:

Definition 10: *The* consensus rule *is an instruction that is executed on each robot i at time k. It dictates that if*

$$\exists j, \exists m \neq j, \text{ such that } S_{i,k} \supseteq \left\{ \mathsf{bel}\left(X_{k_s:k_{e,j}}\right)^j, \mathsf{bel}\left(X_{k_s:k_{e,m}}\right)^m \right\}, k_{e,j} > k_{e,m},$$

then

$$\begin{aligned} \mathsf{bel}\left(X_{k_s:k_{e,m}}\right)^j &\leftarrow \mathsf{bel}\left(X_{k_s:k_{e,j}}\right)^j, \; \textit{and} \\ S_{i,k_{e,i}} &\leftarrow \left\{S_{i,k_{e,i}} \cup \mathsf{bel}\left(X_{k_s:k_{e,m}}\right)^j\right\} - \mathsf{bel}\left(X_{k_s:k_{e,j}}\right)^j. \end{aligned}$$

In other words, the consensus rule dictates that if robot i has multiple distributed contributions in its knowledge set, it will truncate them according to the distributed contribution with the earliest-ending time, and then update its knowledge set. Note that because we are performing recursive filtering, a truncated estimate does not account for any future measurements. Therefore, future estimates are not affected by the truncation. In practice when we implement this rule on a robot, we can make it part of the arbitration function. Fig. 4.3 contains the same information flow graph as in Fig. 4.2, but it shows when each robot locally detects partial checkpoints and distributed partial checkpoints. In this example, note that the consensus rule is applied to truncate robot 1's distributed contribution from $\mathsf{bel}\left(X_{1:3}\right)^1$ to $\mathsf{bel}\left(X_{1:2}\right)^1$.

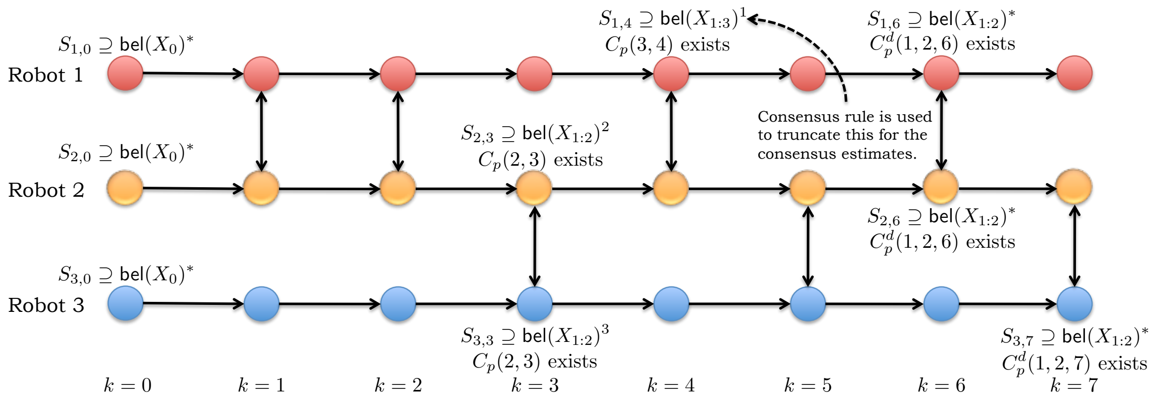

$S_{1,0} \supseteq \mathsf{bel}(X_0)^*$
Robot 1
$S_{1,4} \supseteq \mathsf{bel}(X_{1:3})^1$
$C_p(3,4)$ exists
$S_{1,6} \supseteq \mathsf{bel}(X_{1:2})^*$
$C_p^d(1,2,6)$ exists
Consensus rule is used to truncate this for the consensus estimates.
$S_{2,0} \supseteq \mathsf{bel}(X_0)^*$
Robot 2
$S_{2,3} \supseteq \mathsf{bel}(X_{1:2})^2$
$C_p(2,3)$ exists
$S_{2,6} \supseteq \mathsf{bel}(X_{1:2})^*$
$C_p^d(1,2,6)$ exists
$S_{3,0} \supseteq \mathsf{bel}(X_0)^*$
Robot 3
$S_{3,3} \supseteq \mathsf{bel}(X_{1:2})^3$
$C_p(2,3)$ exists
$S_{3,7} \supseteq \mathsf{bel}(X_{1:2})^*$
$C_p^d(1,2,7)$ exists
$k=0$
$k=1$
$k=2$
$k=3$
$k=4$
$k=5$
$k=6$
$k=7$

Figure 4.3: An information flow graph showing the existence of partial checkpoints and distributed partial checkpoints. Assume that all robots initially have $\mathsf{bel}\left(X_0\right)^*$ in their knowledge sets. At $k=3$, robots 2 and 3 obtain distributed contributions for timesteps 1 to 2. Hence, the checkpoint $C_p(2,3)$ exists for both of them. Robot 1 is able to obtain the distributed contribution for timesteps 1 to 3 at $k=4$, hence, $C_p(3,4)$ exists for robot 1. We must remember that distributed contributions are obtained after communication. At $k=6$, robots 1 and 2 obtain the distributed contributions from all robots. Both robots apply the consensus rule to truncate $\mathsf{bel}\left(X_{1:3}\right)^1$ to $\mathsf{bel}\left(X_{1:2}\right)^1$, and calculate the consensus estimate by using the arbitration function. Therefore, $C^d(1,2,6)$ exists. At the next timestep, robot 3 also obtains the consensus estimate.

When a robot has obtained distributed contributions that have the same time interval from all robots (i.e., when a distributed partial checkpoint exists for the second synchronization), it can use the arbitration function to generate the consensus estimate and apply the Markov property. As we have seen previously in Chapter 2, the partial checkpoints may come into existence at different timesteps for different robots. Similarly, distributed partial checkpoints may come into existence at different timesteps for different robots. Hence, not all robots will apply the Markov property at the same time. We will show in the following theorem that all robots can still obtain the consensus estimate if a robot discards information as it applies the Markov property. This theorem is very similar to Theorem 3.1, previously introduced for the decentralized cooperative localization problem. The intuition here is that if a robot applies the Markov property, it means that it has already obtained the consensus estimate. Therefore, instead of communicating the data that were used to generate that consensus estimate, it can just pass the consensus estimate to another robot.

Theorem 7.3: *Suppose $C^d(k_s, k_c, k_d)$ exists. If robot i applies the Markov property at $k_{d,i}$ when $C^d_p(k_s, k_c, k_{d,i})$ exists, then $C^d(k_s, k_c, k_d)$ continues to exist and all robots can still obtain the consensus estimate $\mathsf{bel}\left(X_{k_s:k_c}\right)^*$.*

Proof. We will start with the condition that a distributed partial checkpoint exists for robot i:

$$\begin{aligned} &C^d_p(k_s, k_c, k_{d,i}) \text{ exists} \\ \Rightarrow &S_{i,k_{d,i}} \supseteq \mathsf{bel}\left(X_{k_s:k_c}\right)^* \end{aligned}$$

After applying the Markov property, robot i will no longer have the past information required to calculate $\mathsf{bel}\left(X_{k_s:k_c}\right)^j$. Let Q represent the robots $j \neq i$, for which $C^d_p(k_s, k_c, k_{d,j})$ exists at $k_{d,j} > k_{d,i}$ and $S_{j,k_{d,j}} \supseteq S_{i,k_{d,i}}$ (i.e., robots for which distributed partial checkpoints exist after the existence of robot i's distributed partial checkpoint, and after communicating with robot i). Also, let $\overline{Q} = N - Q$. At the distributed partial checkpoint existence time for any robot j:

$$\begin{aligned} &S_{j,k_{d,j}} \supseteq \begin{cases} \left\{\mathsf{bel}\left(X_{k_s:k_c}\right)^m | m \in N\right\}, \forall j \in \overline{Q} \\ \mathsf{bel}\left(X_{k_s:k_c}\right)^*, \forall j \in Q \end{cases} \\ \Rightarrow &S_{j,k_{d,j}} \supseteq \mathsf{bel}\left(X_{k_s:k_c}\right)^*, \forall j \in N \end{aligned}$$

Therefore, $C^d(k_s, k_c, k_d)$ continues to exist and all robots can still obtain the consensus estimate $\mathsf{bel}\left(X_{k_s:k_c}\right)^*$. □

Once again, Theorem 7.3 indicates that robots affected by robot i's action of applying the Markov property can directly receive the consensus estimate from robot i (instead of the distributed contributions from all robots). Robots that are not affected by robot i will use the arbitration function to obtain the consensus estimate. Therefore, robots can apply the Markov property without considering each other's knowledge.

Recall that due to the network topology, the timestep at which we can obtain the centralized-equivalent estimate in decentralized cooperative SLAM may not be the current timestep. Likewise, the timestep at which we can obtain the consensus estimate in distributed and decentralized cooperative SLAM may not be the current timestep. We will call this delay the *centralized-equivalent estimate delay.* In decentralized cooperative SLAM, where a robot only needs to detect the existence of a partial checkpoint, $C_p(k_{c,i}, k_{e,i})$ (i.e., where we only synchronize once), this delay is $k_{e,i} - k_{c,i}$. For distributed and decentralized cooperative SLAM (i.e., where we synchronize twice), where a robot first needs to detect the existence of partial checkpoint, and then the existence of a partial distributed checkpoint, $C_p^d(k_s, k_c, k_{d,i})$, this delay is $k_{d,i} - k_c$. Furthermore, we expect the centralized-equivalent estimate delay to be greater in distributed and decentralized cooperative SLAM (because we need to synchronize twice before obtaining the centralized-equivalent estimate).

Another type of delay what we can define is the time between which consecutive centralized-equivalent estimates are obtained. We call this the *existence delay.* For the non-distributed decentralized cooperative SLAM case, this is the time between two consecutive instances of partial checkpoint existence. In other words, if $C_p(k_{c_1,i}, k_{e_1,i})$ exists right before $C_p(k_{c_2,i}, k_{e_2,i})$ exists, the existence delay is $k_{e_2,i} - k_{e_1,i}$. In distributed and decentralized cooperative SLAM, this delay is the time between two consecutive instances of distributed partial checkpoint existence. Hence, if $C_p^d(k_{s_1,i}, k_{c_1,i}, k_{d_1,i})$ exists right before $C_p^d(k_{s_1,i}, k_{c_1,i}, k_{d_1,i})$ exists, then the existence delay is $k_{d_2,i} - k_{d_1,i}$. In general, we can expect the existence delay to be greater in distributed and decentralized cooperative SLAM (again due to the need to synchronize twice).

To summarize, to examine how a single robot can obtain the (centralized-equivalent) consensus estimate, we reused the concept of a partial checkpoint from previous chapters. However, in distributed and decentralized cooperative SLAM, the existence of a partial checkpoint is used to indicate when a robot can calculate its distributed contribution (an estimate based on a particular set of data association hypotheses). A robot requires the distributed contributions from all robots to form the (centralized-equivalent) consensus estimate by using an arbitration function. This is possible when a distributed partial checkpoint exists. This also allows a robot to apply the Markov property to discard information that it has already used. However, it is possible that the distributed contributions from other robots do not span the same time interval. Therefore, before applying the arbitration function, a robot must apply the consensus rule according to Definition 10. In applying the Markov property, we showed that it is possible for a robot to do so without considering the knowledge of other robots, similarly to what we have shown in Chapter 2 for the decentralized cooperative localization problem. To show the connections between the theoretical development of this chapter and that of the previous chapters, Fig. 4.4 is an illustrative summary of the relationships between the theorems that we have introduced up to this point.

4.3 Applying the Theory

From the theoretical development in the previous section, we now have the fundamental components of our distributed and decentralized cooperative SLAM algorithm, which allows us to distribute the computation of multiple estimates with different data association hypotheses. Furthermore, it allows robots to obtain (centralized-equivalent) consensus estimates while operating in a network that is never guaranteed to be fully connected.

In terms of the initial knowledge of robots, we still need robots to initially know the total number of robots in the team. As explained in Chapter 3, this is because the presence of landmarks does not allow us to divide the team into independent subsystems while maintaining the guarantee that the centralized-equivalent estimate is obtainable.

4.3.1 The Distributed and Decentralized Cooperative SLAM Algorithm

According to our theory, each robot must try to detect the existence of a partial checkpoint before calculating a distributed contribution, and detect the existence of a distributed partial checkpoint before calculating the consensus estimate (which is equivalent to the centralized estimate). Fig. 4.5 is an illustration of the distributed and decentralized cooperative SLAM algorithm (Algorithm 3). A copy of this algorithm runs on every robot, and iterates once per timestep.

To explain Algorithm 3 in greater detail, on line 1, information exchange occurs with other robots. Remember that we are using our knowledge set update rules defined by (2.3) and (2.4). If a distributed contribution has not yet been calculated (line 2), then we attempt to look for a partial checkpoint and calculate $\mathsf{bel}\left(X_{k_s:k_{c,i}}\right)^i$ using a recursive state estimation method, if possible (lines 3-7). We then apply the consensus rule on

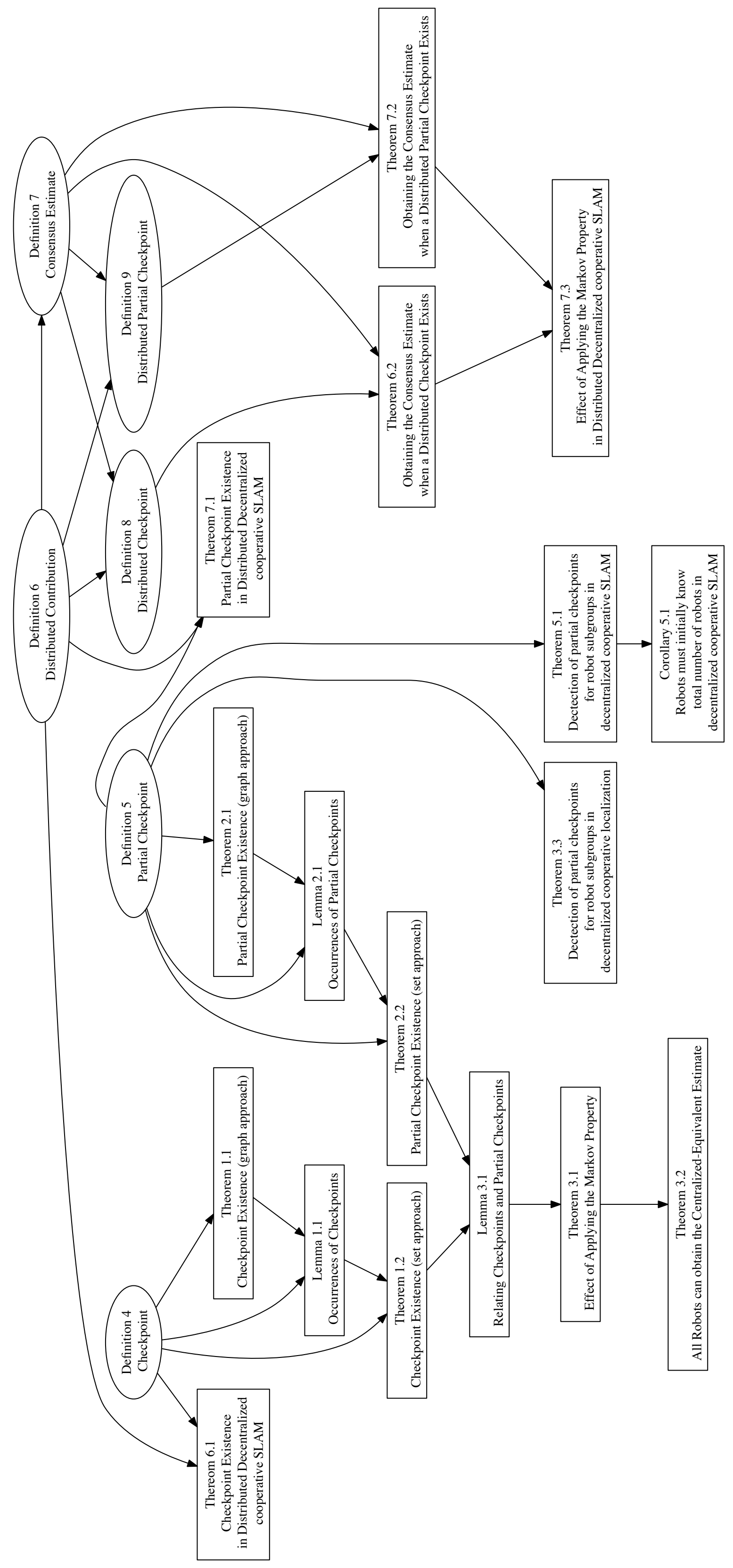

Definition 7
Consensus Estimate
Definition 9
Distributed Partial Checkpoint
Theorem 7.2
Obtaining the Consensus Estimate
when a Distributed Partial Checkpoint Exists
Theorem 7.3
Effect of Applying the Markov Property
in Distributed Decentralized cooperative SLAM
Theorem 6.2
Obtaining the Consensus Estimate
when a Distributed Checkpoint Exists
Definition 6
Distributed Contribution
Definition 8
Distributed Checkpoint
Thereom 7.1
Partial Checkpoint Existence
in Distributed Decentralized
cooperative SLAM
Theorem 5.1
Dectection of partial checkpoints
for robot subgroups in
decentralized cooperative SLAM
Corollary 5.1
Robots must initially know
total number of robots in
decentralized cooperative SLAM
Definition 5
Partial Checkpoint
Theorem 2.1
Partial Checkpoint Existence (graph approach)
Lemma 2.1
Occurrences of Partial Checkpoints
Theorem 3.3
Dectection of partial checkpoints
for robot subgroups in
decentralized cooperative localization
Theorem 2.2
Partial Checkpoint Existence (set approach)
Theorem 1.1
Checkpoint Existence (graph approach)
Lemma 1.1
Occurrences of Checkpoints
Lemma 3.1
Relating Checkpoints and Partial Checkpoints
Theorem 3.1
Effect of Applying the Markov Property
Theorem 3.2
All Robots can obtain the Centralized-Equivalent Estimate
Theorem 1.2
Checkpoint Existence (set approach)
Definition 4
Checkpoint
Thereom 6.1
Checkpoint Existence
in Distributed Decentralized
cooperative SLAM

Figure 4.4: A graphical summary of the relationship between theorems up to Theorem 7.3.

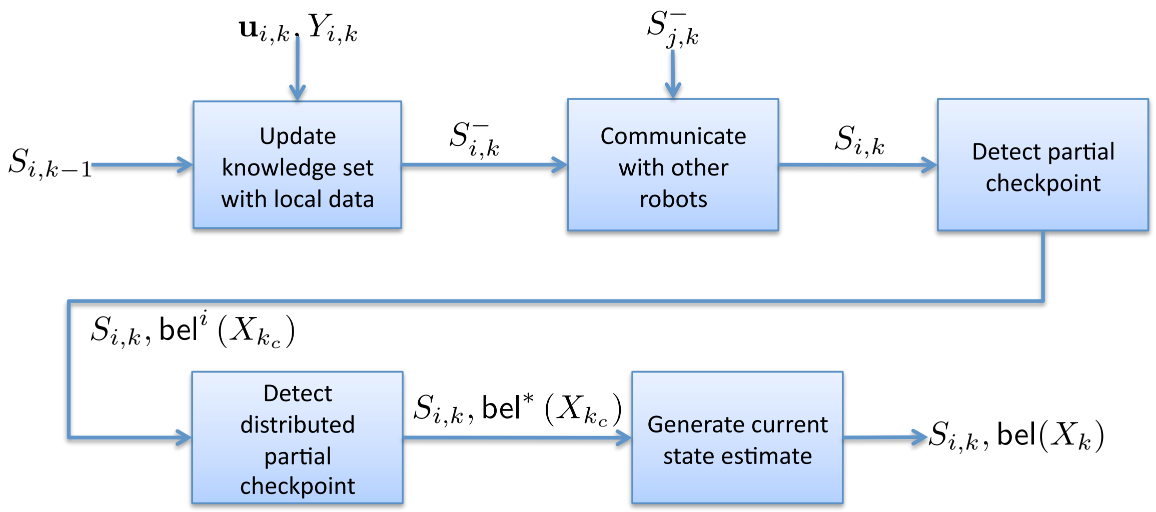

$\mathbf{u}_{i,k}, Y_{i,k}$
$S^-_{j,k}$
$S_{i,k-1}$
Update knowledge set with local data
$S^-_{i,k}$
Communicate with other robots
$S_{i,k}$
Detect partial checkpoint
$S_{i,k}, \mathsf{bel}^i(X_{k_c})$
Detect distributed partial checkpoint
$S_{i,k}, \mathsf{bel}^*(X_{k_c})$
Generate current state estimate
$S_{i,k}, \mathsf{bel}(X_k)$

Figure 4.5: The distributed and decentralized cooperative SLAM algorithm, which provides consensus estimates generated from the distributed contributions from all robots. The need to synchronize information twice is accomplished by the detection of a partial checkpoint, and the detection of a distributed partial checkpoint.

line 9 to ensure that all distributed contributions within the knowledge set have the same time interval. If a distributed partial checkpoint exists (line 10), we can apply the arbitration function (line 11) to obtain the consensus estimate in our knowledge set (line 12). On line 13, we apply the Markov property and discard information that is replaced by the consensus estimate. Finally, we calculate a temporary estimate for the current time on line 15 using information available in the knowledge set. In doing so, robots temporarily assume other robots maintain their last known velocity if odometry information is unavailable.

Note here that as with Algorithms 1 and 2, we can use a variety of recursive filters in calculating a distributed contribution (on line 5). Also, note that the estimate for the current timestep produced on line 15 is a temporary estimate that does not account for information from all robots up to the current time. This is because we are operating in a network that is never guaranteed to be fully connected, and hence information from other robots at the current timestep may not be received until a later time. Nevertheless, the (centralized-equivalent) consensus estimate can always be recovered at a later time.

4.3.2 Complexity Analysis

For the complexity analysis of our distributed and decentralized cooperative SLAM algorithm, we will refer back to the worst-case scenario illustrated in Fig. 3.4, where every robot measures every other robot and known landmarks at every timestep, and all but

Algorithm 3: Distributed and decentralized cooperative SLAM(k, $\mathbf{u}_{i,k}$, $Y_{i,k}$, $S_{i,k-1}$, $S_{j,k}$ $(i \in N)(\forall j)(d_k^{j,i} \leq r_{\mathsf{comm}})$)

```
1  $S_{i,k} \leftarrow S_{i,k-1} \cup \{\mathbf{u}_{i,k}\} \cup \{Y_{i,k}\} \bigcup_{j \in R_{i,k}} S_{j,k}^{-}$
2  if $\nexists i, S_{i,k} \supseteq \mathsf{bel}\left(X_{k_{c,i}}\right)^i$ then
3      $K \leftarrow \{k_r\}$, such that $\{\mathbf{u}_{j,k_r} | j \in N\} \in S_{i,k}$
4      if $K \neq \emptyset$ then
5          calculate $\mathsf{bel}\left(X_{k_{c,i}}\right)^i$ such that $k_{c,i} = \max(K)$
6          $S_{i,k} \leftarrow S_{i,k} \cup \mathsf{bel}\left(X_{k_s:k_{c,i}}\right)^i$
7      end
8  end
9  apply the Consensus Rule
10 if $S_{i,k} \supseteq \left\{\mathsf{bel}\left(X_{k_s:k_c}\right)^j | j \in N\right\}$ then
11     $\mathsf{bel}\left(X_{k_s:k_c}\right)^* \leftarrow f^a\left(\left\{\mathsf{bel}\left(X_{k_s:k_c}\right)^j | j \in N\right\}\right)$
12     $S_{i,k} \leftarrow S_{i,k} \cup \mathsf{bel}\left(X_{k_s:k_c}\right)^*$
13     $S_{i,k} \leftarrow S_{i,k} - \left\{\mathbf{u}_{j,k_r}, Y_{j,k_r}, \mathsf{bel}\left(X_{k_s-1}\right)^* | j \in N\right\}, \forall k_r \leq k_c$
14 end
15 $\mathsf{bel}\left(X_k\right) \leftarrow p(X_k | S_{i,k})$
16 return $\{\mathsf{bel}\left(X_k\right), S_{i,k}\}$
```

one robot communicates with every other robot at every timestep. As before, the number of measurements is on the order of $n^2 + nm$.

Assume that the EKF is used within our framework, and that we have d data association hypotheses, which can be divided evenly amongst the robots. When all robots reestablish communication with each other, the computational complexity of our distributed decentralized approach is $O((k - k_c)(n^4 + n^3 m)d/n)$ while that of the (non-distributed) decentralized approach is $O((k - k_c)(n^4 + n^3 m)d)$. For both the distributed and non-distributed approaches, the complexities for communication bandwidth and memory use are $O((k - k_c)(n^3 + n^2 m))$ and $O\left((k - k_c)(n + m)^2\right)$, respectively. However, note that the $(k - k_c)$ centralized-estimate delay factor is generally larger for the distributed case (which we can support with our experimental results). This again is due to the double synchronization required with the distributed approach. From the experimental results in Chapter 3, we already know that a lower communication range limit leads to less frequent communication between robots. This in turn increases both the centralized-equivalent estimate delay, and the existence delay, and potentially produces greater difference between the current estimates produced by the decentralized estimator and the centralized estimator.

4.4 Performance Analysis

4.4.1 Experiment Setup

To validate our distributed and decentralized cooperative SLAM algorithm, we will conduct a proof-of-concept experiment utilizing our multi-robot cooperative localization and mapping dataset that was previously used for testing our decentralized cooperative SLAM algorithm in Chapter 3. We will use sub-dataset 1 in particular, which runs for a duration of 30 minutes. Furthermore, we will limit the communication range limit to 2m, causing the robot network to be fully connected only 21% of the time.

In testing our distributed approach, we will assume that data association is unknown (i.e., the barcode patterns on the landmarks are not used for correspondences). This means that robots can only use the position estimate of robots and landmarks to determine the actual robot or landmark that corresponds with a range-bearing measurement.

Since our focus is on demonstrating our distributed framework, and not on a particular data association method, or a particular filtering method, we will use a simple maximum likelihood [108] approach for data associations and the extended Kalman filter (EKF) [53] within our algorithm. However, note that our algorithm allows for other data association methods (such as a simple nearest neighbour [108] approach, or a more sophisticated joint-compatibility branch and bound [112] approach for examples) and other recursive filtering methods to be used.

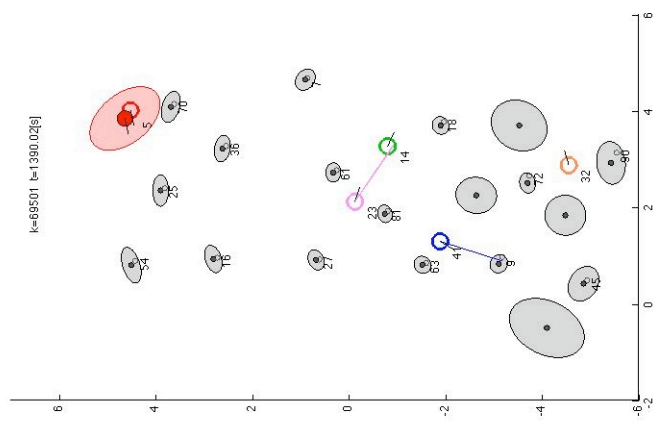
In this proof-of-concept experiment, when a robot detects a partial checkpoint, it will produce one distributed contribution. However, note that it is possible for each robot to maintain numerous estimates, each based on a different set of data association hypotheses. Nevertheless, we will have distributed the calculations of estimates based on various data association hypotheses amongst the team. When a robot obtains a measurement, $\mathbf{y}_{i,k}^{j,i}$, the identity of i is known since it is the robot that initiated the measurement. However, the robot must make a hypothesis to what j corresponds. In our experiment, a robot will randomly select a possible value for j. It will then calculate the likelihood of the measurement based on its current estimates of robot poses and landmark positions. That is, it will take its current knowledge of robot and landmark positions (and their uncertainty) to determine the likelihood distribution for the expected measurement, and calculate the *Mahalanobis distance* [108] using the actual measurement. We can expect that as the actual measurement gets closer to the expected measurement, the Mahalanobis distance will also decrease (i.e., the actual measurement appears more likely). Since each robot is only selecting one (out of many possible) data association hypothesis for a measurement, we apply a threshold to ensure that the correspondence selected is at least somewhat likely. If the Mahalanobis distance for a measurement is below the threshold value, the robot must select an alternate correspondence for the measurement. We also have a distance threshold to determine when a new landmark should be initiated. From the Mahalanobis distance of a measurement, we can also obtain the likelihood of the measurement. That is, the value of the PDF $p(\mathbf{y}|X_k)$ evaluated at $\mathbf{y} = \mathbf{y}_{i,k}^{j,i}$. For each distributed contribution, its metric (used in the arbitration function) is the product of all the measurement likelihoods. When a distributed partial checkpoint exists for a robot, the arbitration function (which is the same for all robots) selects the distributed contribution with the highest metric as the consensus estimate.

4.4.2 Performance Results

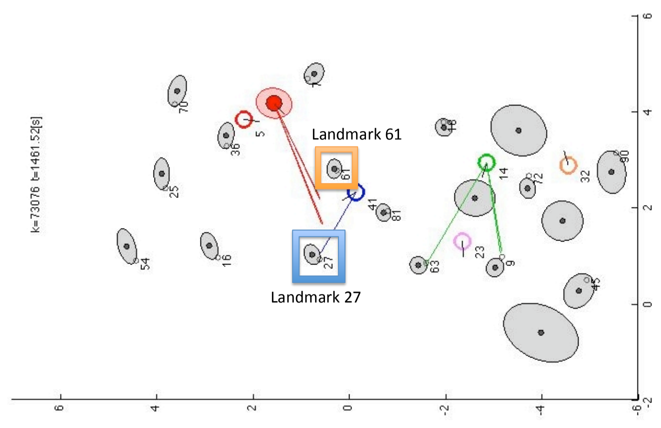
Recall that our motivation for maintaining multiple estimate based on different data association hypotheses is that in case we make a bad choice in data association (which we cannot correct since we are performing recursive filtering and applying the Markov property to discard used information), we can throw out the estimate that used the bad data association and continue with the other estimates that we have (by using the arbitration function). To provide a real example of this from our experiment, consider the sequence of events shown in Fig. 4.6. In each sub-figure, the estimates (drawn as ellipses) of robot and landmark positions from robot 1 (the red robot) are shown. In Fig. 4.6a, we can see that robot 1's estimates of landmark positions are fairly consistent (i.e., the true positions of the landmarks are close to the estimated mean positions, and within the 95% uncertainty ellipses). Shortly after, in Fig. 4.6b, robot 1's self estimate appears to have diverged (partly due to a lack of measurements and unexpected motion). At this point, robot 1 makes a measurement of landmark 61 (which is drawn from its estimated pose in the plot). Visually this measurement falls somewhere between landmark 61 and landmark 27 (i.e., both are likely to have been measured). In this case, robot 1 incorrectly corresponds the measurement with landmark 27, and runs its estimator based on this data association hypothesis. Due to this mistake, it later causes the entire map to begin to diverge, as shown in Fig. 4.6c. If we examine the estimate of landmark 54, it can be clearly seen that the actual position of the landmark is well outside of the uncertainty ellipse of its estimate. If we only maintained one estimate, based on the data association hypothesis made by robot 1, it is plausible that the map will continue to diverge and we will not be able to go back to correct the mistake in data association. Luckily, we have other robots with estimates calculated with the correct association for the previous measurement of landmark 61. In Fig. 4.6d, when the robots are within communication range, they are able to exchange their distributed contributions and select one of them as the consensus estimate. In this case, robot 1's distributed contribution was not chosen as the consensus estimate. As a result, we can see that the landmark estimates are once again consistent. Note that it is a possibility that a consensus estimate has been selected from a distributed contribution that uses incorrect correspondences. In Fig. 4.6, the relatively larger ellipses to the right of each plot are results of incorrect correspondences in the consensus estimate.

We will now examine the error (compared against groundtruth data) for several components of robot 1's estimate. The results from the distributed and decentralized cooperative SLAM algorithm are similar to the results we observed from the decentralized cooperative SLAM algorithm. Robot 1's x-position estimate (output from line 15 of Algorithm 3) errors of itself, robot 3 (representative of errors for other robots), and landmark 54 from Fig. 4.6c (representative of errors for other landmarks) are shown in Fig. 4.7. Instances of distributed partial checkpoint existence are also shown. Even though the network is only fully connected 21% of the time, robot 1 still maintains an estimate of its own pose with low errors because its own odometry data is always available. Landmark estimates also have low errors because they are static. However, estimates of other robots can have high errors when they are out of communication range. Nevertheless, estimate errors for other robots reduce once updated odometry data are available, and when a robot obtains a new consensus estimate. Notice that in the landmark estimate error plot, the error temporarily increased at about 1500s. This error is the result of the mistake in data association shown in Fig. 4.6. When robot 1 obtains a consensus estimate, the estimate error immediately reduces.

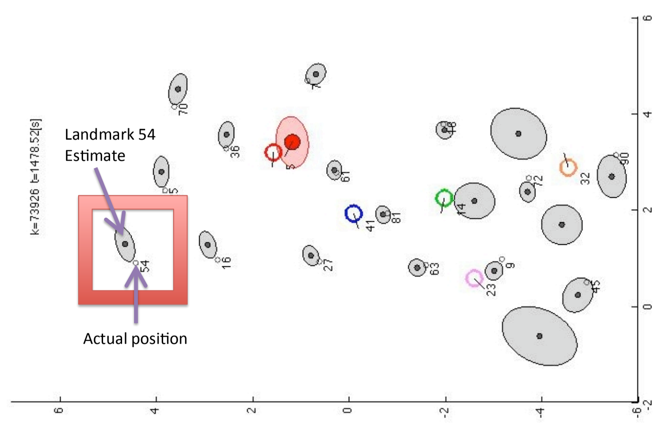
Note that the results shown in Fig. 4.7 are temporary current estimates, which are updated later with consensus estimates. As we have discussed earlier, one of the trade-offs between our distributed approach and a non-distributed approach is the centralized-equivalent estimate delay, and the existence delay. We expect that the delays will be greater in the distributed case because robots have to look for partial checkpoints and distributed partial checkpoints to obtain the (centralized-equivalent) consensus estimate (i.e., double synchronization is required). In contrast, robots only need to look for partial checkpoints in the non-distributed case to obtain the centralized-equivalent estimate. In our experiment, for the decentralized cooperative SLAM algorithm, the average centralized-equivalent estimate delay is 144 timesteps (2.9s), while the average existence delay is 490 timesteps (9.8s). In comparison, for the distributed and decentralized cooperative SLAM algorithm, the average centralized-equivalent delay is 518 timesteps (10.4s), while the average existence delay is 745 timesteps (14.9s). Although we experi-



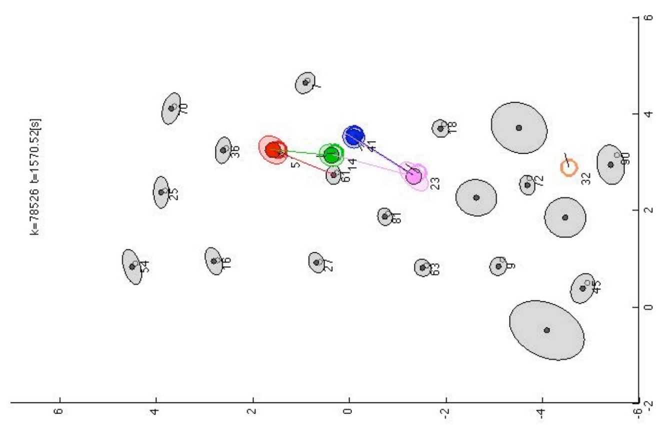
(a) This plot shows (the red) robot 1's position estimates of robot and landmark positions with (95%) uncertainty ellipses. At the moment, robot 1's estimates appear consistent.



(b) A mistake in data association is made, as robot 1 associates a measurement (red line) of landmark 61 (orange box) with landmark 27 (blue box).

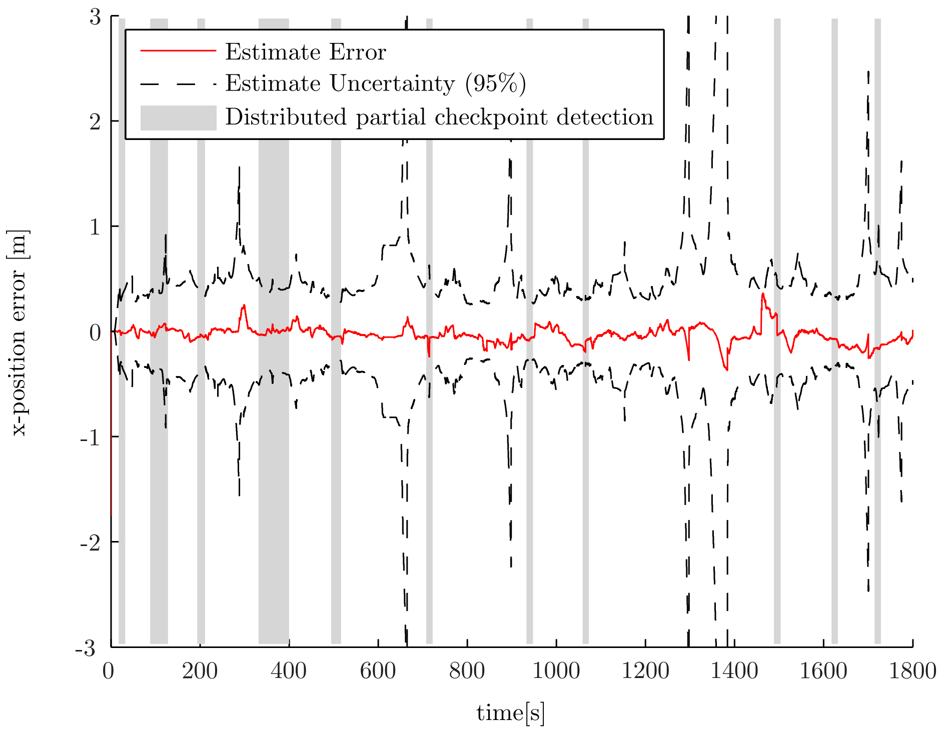


(c) The mistake in data association causes robot 1's estimates to begin diverging. In this case all the estimates appears to have rotated. (Look ahead to Fig. 4.7c to see the x-estimate error plot of the landmark in the red box.

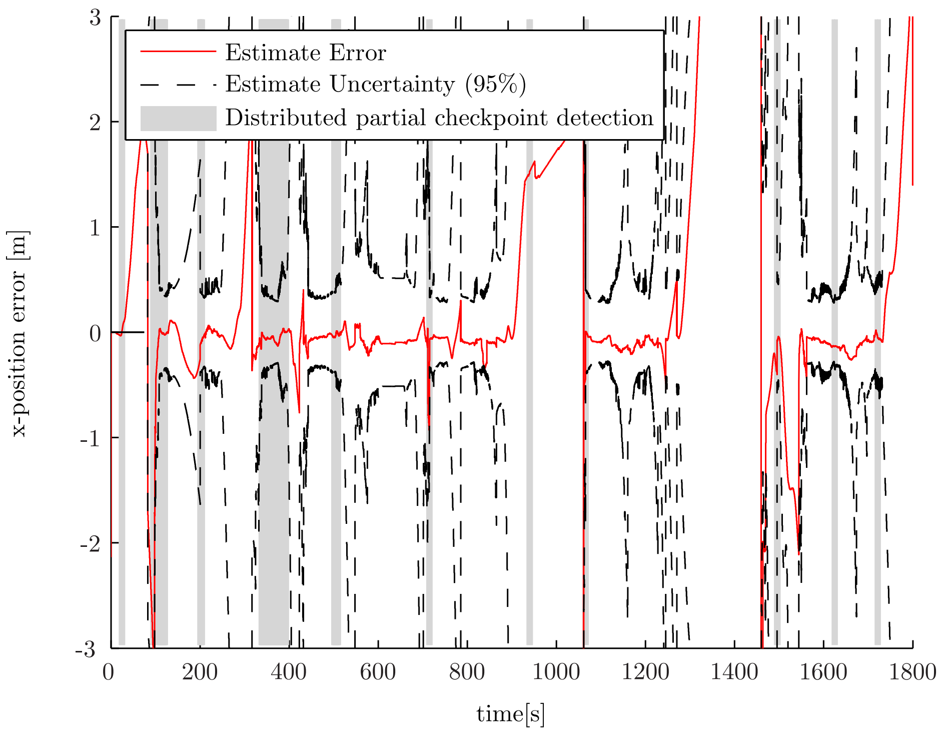


(d) A distributed partial checkpoint occurs for robot 1. Its diverged estimate is replaced by the consensus estimate, which does not make the mistake in data association. The position estimates for robots and landmarks appear consistent again.

Figure 4.6: A sequence of events from our experiments showing the benefit of maintaining multiple data association hypotheses.

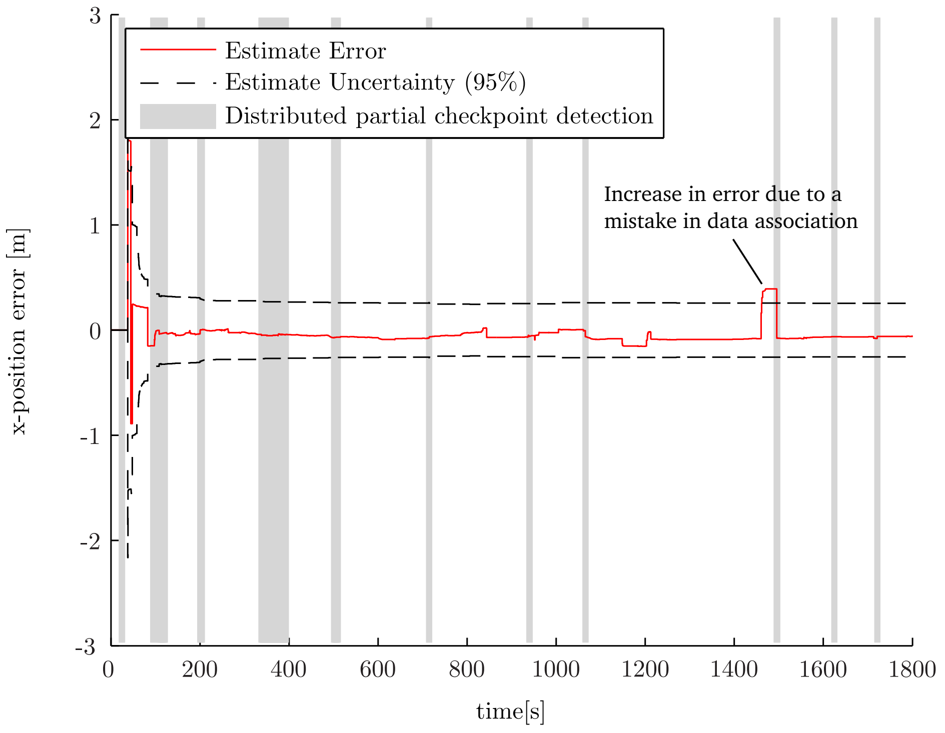


(a) Robot 1's estimate error of its own x-position.



(b) Robot 1's estimate error of robot 3's x-position.

Figure 4.7: Components of robot 1's distributed decentralized estimates with $r_{\mathsf{comm}} = 2$.



(c) Robot 1's estimate error of landmark 54's x-position.

Figure 4.7: Components of robot 1's distributed decentralized estimates with $r_{\mathsf{comm}} = 2$.

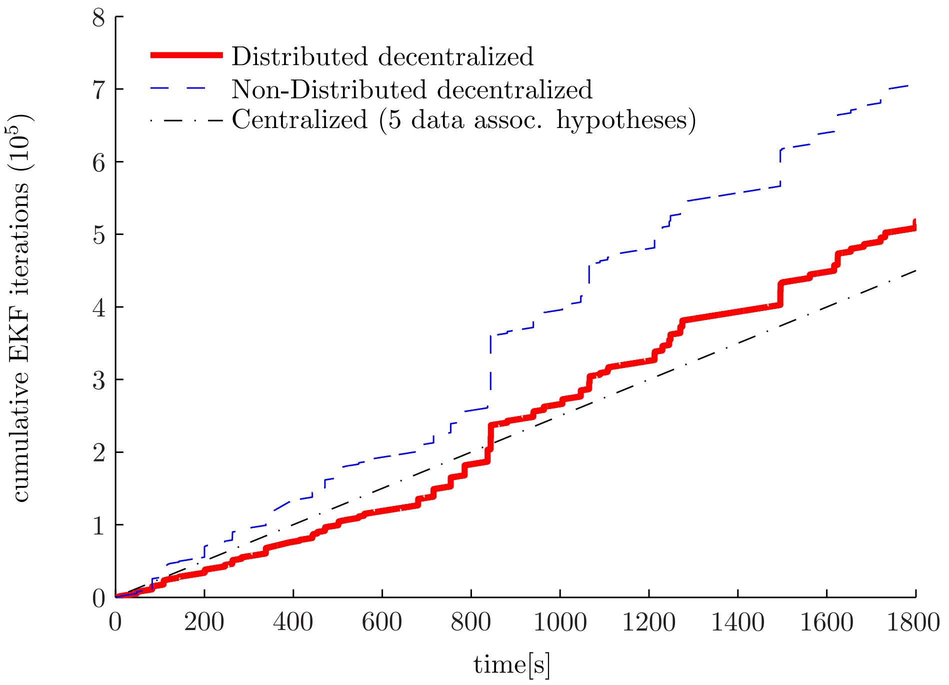


Figure 4.8: A comparison of the amount of computation performed on robot 1.

ence greater delays with a distributed approach, the amount of computations performed by each robot is reduced compared to the non-distributed approach if we consider the same number of data association hypotheses in generating the centralized-equivalent estimate. As a comparison of the amount of computation performed on a robot, Fig. 4.8 plots the cumulative number of EKF iterations processed on robot 1 over time with our distributed decentralized approach versus the non-distributed approach. As we can see, the amount of computation performed by a robot using the distributed approach is lower than that of the non-distributed approach.

4.5 Summary

In this chapter, we examined the cooperative SLAM problem with the assumption that data association for measurements is unknown, while keeping the assumption that the communication network is never guaranteed to be fully connected. To reduce the chance of divergence and catastrophic failure in our estimator caused by incorrect correspondences, we chose to consider multiple data association hypotheses and perform state estimation on using each of those hypotheses. In order to reduce computational load, we examined how computation can be distributed amongst a team of robots and developed the distributed and decentralized cooperative SLAM problem.

Comparing with the non-distributed approach to cooperative SLAM, the distributed approach requires two synchronization events instead of one. The first is to ensure that robots have the necessary information in knowledge sets to calculate distributed contributions, which are essentially estimates based on the same measurement and odometry data, but on different sets of data association hypotheses selected through, for example, a random process. For this, we reused the notions of a checkpoint and a partial checkpoint. The second synchronization event ensures that robots have the distributed contributions from all robots, so that the (centralized-equivalent) consensus estimate can be produced using the arbitration function.

Since the local information on each robot may be different due to the dynamic network, partial checkpoints may occur at different times, leading to distributed contributions to be calculated for different times on different robots. To ensure that we can apply the arbitration function to obtain the consensus estimate, we require robots to maintain their distributed contributions over a time interval, so that it gives us the opportunity to take the distributed contributions from all robots and truncate them to the same time interval. When a robot obtains a consensus estimate, it is able to apply the Markov property to discard used information. Similarly to Chapter 2, we proved that a robot can apply the Markov property without consideration of other robots' knowledge sets, while ensuring that all other robots can still obtain the consensus estimate.

In our proof-of-concept validation, we showed that the performance of the distributed and decentralized cooperative SLAM algorithm is comparable to that of the decentralized cooperative SLAM algorithm in terms of estimator error. However there is a trade-off with using a distributed approach as it causes greater centralized-equivalent estimate delay and existence delay. The benefit with a distributed approach is the decrease in the amount of computation performed on each robot. In conclusion, we are able to reduce the amount of computation on each robot by allowing longer delays in obtaining centralized-equivalent estimates because we need to perform an additional synchronization of information with the distributed approach.

Chapter 5

Conclusions

How strange is the lot of us mortals! Each of us is here for a brief sojourn; for what purpose he knows not, though he senses it. But without deeper reflection one knows from daily life that one exists for other people.

Albert Einstein (1879–1955),
scientist.

In this thesis, we examined the problem of cooperative SLAM with a multi-robot system operating in a network that is never guaranteed to be fully connected. In this problem, we required each robot to estimate the poses of all robots, as well as the position of all known landmarks. We call our novel solution to the problem the "centralized-equivalent approach", because it enables all robots to recover the centralized-equivalent estimates whenever possible. That is, when possible, the estimates that are exactly the same as ones produced by a centralized-estimator, even when the network is never fully connected. This enabled us to avoid the problems of out-of-sequence-measurements, as well as cyclic updates. Our solution is widely applicable, and can be used in situations where robots are operating over a large distance and cannot always maintain channels for communication, or in environments where communication range is limited by occlusions. Even if robots are in close proximity, our solution can be used simply to improve the robustness of the system by handling temporary drop outs and delays in network connections.

In a sense, the development of our solution has shown that in the best interest of the team, robots should act in the best interest of themselves provided that they share information with the rest of the team. That is, assuming that the knowledge set exchange rules are followed, if a robot applies the Markov property whenever it detects the existence of a partial checkpoint, we can guarantee that all other robots are able to also obtain the same centralized-equivalent estimate and also apply the Markov property. This allows a robot to be greedy and apply the Markov property whenever possible based only on its local knowledge. Furthermore, a robot does not need to know what other robots know when applying the Markov property, nor does it need to care about this.

In developing a solution, we decided to first look at a simplified version of the decentralized SLAM problem by neglecting the presence of landmarks and the need to estimate a map. This transformed the decentralized SLAM problem into the decentralized localization problem, where all robots need to obtain the centralized-equivalent estimate of all robot poses in a decentralized manner. In our mathematical formulation of the problem, we defined knowledge sets and their update rules based on the exchange of information between robots. We defined the key concept of a checkpoint, which allows all robots to obtain a centralized-equivalent estimate and apply the Markov property, and developed theorems regarding its existence. However, checkpoints are impractical in the sense that in order to detect their existence, an outside observer is required. This led us to examine information flow from the perspective of a robot. We introduced the concept of a partial checkpoint, which indicates when a single robot can obtain a centralized-equivalent estimate of all robot poses. We also developed several theorems for detecting the existence of a partial checkpoint. In relating partial checkpoints to checkpoints, we were able to show that a robot's decision to invoke the Markov property does not affect another robot's ability to do the same. Furthermore, if all robots invoke the Markov property whenever possible (when a partial checkpoint exists) based on their own knowledge, we can be sure that all robots can still obtain an centralized-equivalent estimate.

Using the theorems we presented, an algorithm was developed that allows each robot to obtain the centralized-equivalent estimate whenever possible. At every timestep, the algorithm produces a current estimate (which may be temporary), and a centralized-equivalent estimate, if possible. The timestep of the centralized equivalent estimate may not be the current time (i.e., there may be a delay) due to network connectivity. This is the reason why the algorithm also generates a temporary estimate for the current timestep. The decentralized cooperative localization algorithm is scalable in the sense that the number of robots in the network does not need to be known initially, but only when communication is bi-directional, and when the communication range limit is greater than the measurement range limit.

We used simulations to validate our decentralized cooperative localization algorithm and showed that memory usage (although large compared to the centralized estimator) is limited by exploiting the Markov property at partial checkpoints. In our simulations, we also looked at how various factors (captured by dimensionless variables) affect performance of our decentralized state estimator in comparison to the centralized estimator, and how this relates to network connectivity. Simulation results show that the performance of our decentralized state estimator begins to approach that of the centralized state estimator when a phase transition occurs in the frequency of network connectivity. It must be stressed again that a centralized state estimator will not be able to produce an estimate unless we assume full network connectivity at all times.

Having established a solution for decentralized cooperative localization in a dynamic network that is never guaranteed to by fully connected, we re-considered the presence of landmarks and returned to the the decentralized cooperative SLAM problem, where it is necessary for all the robots in a team to obtain centralized-equivalent estimates of known landmark positions and all robot poses.

In developing our decentralized cooperative SLAM algorithm, we generalized the concepts of a checkpoint and a partial checkpoint, making them applicable to both the decentralized cooperative localization problem, and the decentralized cooperative SLAM problem. The generalization is important as it allows us to use theorems previously developed for the decentralized cooperative localization problem. However, we proved that it is necessary in decentralized cooperative SLAM for each robot to initially know the total number of robots in the network. This initial condition is not necessary for the cooperative localization case but is required in decentralized cooperative SLAM. Although this may seem somewhat restrictive, it is necessary for ensuring that the centralized-equivalent estimate is obtainable.

To validate our decentralized cooperative SLAM algorithm, we created an extensive hardware experiment dataset. Using this, we were able to compare the results from our algorithm against that from a centralized estimator (which we allowed to cheat by ignoring communication constraints), and evaluated the performance of our algorithm under different communication range limits. Our results again show that memory usage is limited in our algorithm due to use of the Markov property. The results also show that the centralized-equivalent estimate can always be recovered after a period of poor network connectivity. In terms of estimation accuracy, our overall results show that a robot's estimate of its own pose and the position of landmarks are close to the centralized estimate even with a low percentage of time in which the network is fully connected. Furthermore, the accuracy of a robot's estimate of another robot depends on the percentage of time in which the network is fully connected.

In testing our decentralized cooperative SLAM algorithm, we assumed that data association (correspondences of measurements) is known. However, the algorithm should be able to handle the unknown data association case, as it is possible to incorporate a data association routine in our algorithm before an estimate (for both the centralized-equivalent estimate, and the temporary current estimate). The quality of the resulting estimate depends largely on the data association method used, and how well it does in assigning proper correspondences.

To reduce the chance of a catastrophic failure in our estimator caused by incorrect correspondences, we need to maintain multiple data association hypotheses and run the state estimator on each of those hypotheses. However, maintaining multiple hypotheses also increases computational complexity (as we are essentially running a separate filter for each hypothesis), and memory use. In order to reduce computational load, we examined how computation can be distributed amongst a team of robots to make better use of the computational resources available. This led to the distributed and decentralized cooperative SLAM solution. In formulating the solution, we used the concepts of checkpoint and partial checkpoint to indicate when robots can calculate a distributed contribution for the (centralized-equivalent) consensus estimate. Each distributed contribution is essentially an estimate made using a particular data association hypothesis. The contributions from all robots are used to generate the consensus estimate through the use of the consensus rule and an arbitration function when robots detect the existence of a distributed partial checkpoint. Having generated a consensus estimate, we proved that a robot can apply the Markov property without consideration of other robot's knowledge, while ensuring that all other robots can still obtain the same consensus estimate. The performance of the distributed and decentralized cooperative SLAM algorithm is comparable to that of the decentralized cooperative SLAM algorithm in terms of estimator error. However there is a trade-off with using a distributed approach as it causes greater centralized-equivalent estimate delay and existence delay. The benefit with a distributed approach is the decrease in computational load on each robot.

We will summarize the contributions of this thesis in the following list:

1. We introduced the notions of a checkpoint and a partial checkpoint, which indicate when all robots or a single robot can obtain the centralized-equivalent estimate in decentralized cooperative localization and decentralized cooperative SLAM, while operating in a network that is never guaranteed to be fully connected.

2. We developed the decentralized cooperative localization algorithm, where each robot uses only local knowledge to determine when it can obtain the centralized-equivalent estimate and apply the Markov property. While this is a greedy action, we are also guaranteed that other robots can also obtain the centralized-equivalent

estimate for the entire team of robots.

3. We developed the decentralized cooperative SLAM algorithm, which extends the decentralized cooperative localization algorithm and applies to systems with landmarks.

4. We showed that for decentralized cooperative localization, robots do not need to initially know the total number of robots in the team, and that partial checkpoints can be detected based on a subgroup of known robots.

5. We showed that for decentralized cooperative SLAM, it is necessary for robots to initially know the total number of robots in the team to guarantee that the centralized-equivalent estimate is obtainable.

6. We created a multi-robot SLAM dataset, which not only helped validate the algorithms presented in this thesis, but is also freely available for the robotics community to use.

7. We performed detailed analysis on how communication range affects localization performance, memory use, and communication bandwidth requirements in both decentralized cooperative localization and decentralized cooperative SLAM

8. We introduced the notions of a distributed checkpoint and a distributed partial checkpoint, which indicate when all robots or a single robot can obtain a consensus estimate in distributed and decentralized cooperative SLAM. The consensus estimate is considered the best estimate chosen from estimates generated with different data association hypotheses.

9. We presented the distributed and decentralized cooperative SLAM algorithm, where robots produced estimates (distributed contributions) based on specific data association hypotheses when a partial checkpoint exists, and combines them to form a consensus estimate when a partial distributed checkpoint exists.

There are a number of potential future developments based on the work presented in this thesis. Some of these topics aim to further improve the estimates generated with our "centralized-equivalent" approach, while others examine more practical issues in implementing a system for real world applications.

The runaway robot

With the current decentralized cooperative localization, decentralized cooperative SLAM, and distributed and decentralized cooperative SLAM algorithms, if one robot unexpectedly fails to ever communicate with the rest of the team again (whether due to hardware failures, or poor planning), then all robots will never detect the existence of a partial checkpoint or a distributed partial checkpoint again. As such, every robot will continue to collect measurement and odometry information without every applying the Markov property again. Eventually, all robots will run out of memory and cause a system-wide failure. One possible research topic is to examine how to deal with such a scenario. The solution may not be as simple as dropping the estimate of the lost robot, as we also need to consider what needs to be done if the presumably lost robot returns to join the team.

Improve temporary estimates with planning information

One of the consequences of operating in a dynamic network that is never guaranteed to be fully connected is that a robot's estimate of another robot may have diverged so much that the estimate of the other robot is almost useless if actions need to be planned based on the current system state. One possibility of mitigating this divergence is for robots to inform other robots of their intentions of where they are moving to (i.e., their motion plans). Currently in our algorithms, if odometry information is not available from a certain robot in calculating an estimate, we assume that the robot maintained its last known forward and angular velocities. If planning information is available, we can use that to predict the motion of another robot, and possibly reduce the divergence of the estimate on another robot when it goes out of communication range for an extended period of time.

Planning to checkpoint

Along with the motivation from the last point, where we want to prevent estimate divergence, we can plan for checkpoints to occur at regular intervals for the system of robots. There are many ways in which this can be achieved such as having robots return to a certain location for rendezvous with another robot at a certain time. Another approach may be to have robots dedicated as messengers, whose sole purpose is to rendezvous with other robots and relay information so that checkpoints can occur.

The sliding window approach

Throughout this thesis, we have emphasized the application of the Markov property to limit memory use on each robot. However, the fact that the Markov property is applicable (i.e., that future estimates only depends on the current estimate so that past information can be discarded) is only an assumption. If we assume that the Markov property does not apply, then we are forced to perform batch estimation over all timesteps and our decentralized approach because inapplicable. However, that is an extreme case. We can also assume that future estimates only depend on the estimates from a limited number of timesteps in the past (i.e., over a window). This sliding window approach may lead to better estimates, but at a cost of higher computational requirements.

Unknown initial correspondence

An assumption that we have made in this thesis is that the initial relative poses between robots are known (although a robot may not initially have the initial estimate of another robot in its knowledge set, each robot begins with a belief of its own pose, and all the beliefs are expressed in the same reference frame). In other words, we have avoided the initial correspondence problem. How this initial correspondence can be determined is still an open problem in cooperative localization and cooperative SLAM. One approach that falls in line with the notion of a checkpoint is to define the notion of a initial correspondence checkpoint, the existence of which could indicate when enough measurements have been made between robots so that the initial correspondence is observable, and can be solved by batch estimation. Extending this, we can impose that we require adequate measurements so that we can solve for the initial correspondence with a certain level of confidence.

Currently, the number of autonomous multi-robot systems in use is still limited. One of the reasons for this is that we are still at the stage of making single robot systems robust enough for real deployments. Although research and experiments have shown the potential of multi-robot systems in improving localization and mapping results, the effort required in implementing a multi-robot system and making it robust is significantly higher than that for single-robot systems. In many cases, the required effort in operating a multi-robot system may not be worthwhile considering the improvement that is gained. We hope that the work presented in this thesis will help make future multi-robot systems more robust and thereby increase the deployment of multi-robot systems.

Bibliography

[1] R. Madhavan, K. Fregene, and L.E. Parker. Distributed cooperative outdoor multirobot localization and mapping. *Autonomous Robots*, 17(1):23–39, 2004.

[2] G. Sukhatme, J. Montgomery, and R. Vaughan. *Robot Teams: From Diversity to Polymorphism*, chapter Experiments with Cooperative Aerial-Ground Robots. A K Peters, Ltd., 2001.

[3] R. Vaughan, G. Sukhatme, F. Mesa-Martinez, and J. Montgomery. Fly spy: Lightweight localization and target tracking for cooperating air and ground robots. In *Symposium on Distributed Autonomous Robotic Systems*, pages 315–324, 2000.

[4] A. Kelly, A. Stentz, O. Amidi, M. Bode, D. Bradley, A. Diaz-Calderon, M. Happold, H. Herman, R. Mandelbaum, T. Pilarski, et al. Toward reliable off road autonomous vehicles operating in challenging environments. *International Journal of Robotics Research*, 25(5-6):449–483, 2006.

[5] A. Stentz, A. Kelly, H. Herman, P. Rander, O. Amidi, and R. Mandelbaum. Integrated air/ground vehicle system for semi-autonomous off-road navigation. In *Proceedings of AUVSIs Symposium on Unmanned Systems*, 2002.

[6] A. Stentz, A. Kelly, P. Rander, H. Herman, O. Amidi, R. Mandelbaum, G. Salgian, and J. Pedersen. Real-time, multi-perspective perception for unmanned ground vehicles. In *Proceedings of AUVSIs Symposium on Unmanned Systems*, 2003.

[7] L. Chaimowicz, B. Grocholsky, J. F. Keller, V. Kumar, and C. J. Taylor. Experiments in multirobot air-ground coordination. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, pages 4053–4058, 2004.

[8] L. Chaimowicz, A. Cowley, D. Gomez-Ibanez, B. Grocholsky, M.A. Hsieh, H. Hsu, J.F. Keller, V. Kumar, R. Swaminathan, and C.J. Taylor. Deploying air-ground multirobot teams in urban environments. In *Proceedings from the 2005 International Workshop on Multi-Robot Systems*, 2005.

[9] M.-Y.A. Hsieh, V. Kumar, and C.J. Taylor. Constructing radio signal strength maps with multiple robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 4, pages 4184 – 4189, 2004.

[10] B. Grocholsky, S. Bayraktar, V. Kumar, C.J. Taylor, and G. Pappas. Synergies in feature localization by air-ground robot teams. In *Proceedings of the 9th International Symposium on Experimental Robotics*, 2004.

[11] B.P. Grocholsky, R. Swaminathan, V. Kumar, C.J. Taylor, and G.J. Pappas. Coordinated perception by teams of aerial and ground robots. In *Proceedings of SPIE*, volume 5609, page 2004, 2005.

[12] B. Grocholsky, R. Swaminathan, J. Keller, V. Kumar, and G. Pappas. Information driven coordinated air-ground proactive sensing. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2211–2216, 2005.

[13] B. Grocholsky, J. Keller, V. Kumar, and G. Pappas. Cooperative air and ground surveillance. *IEEE Robotics and Automation Magazine*, 13(3):16–26, 2006.

[14] M. Long, A. Gage, R. Murphy, and K. Valavanis. Application of the distributed field robot architecture to a simulated demining task. In *IEEE International Conference on Robotics and Automation*, pages 3193–3200, 2005.

[15] E.Z. MacArthur, D. MacArthur, and C. Crane. Use of cooperative unmanned air and ground vehicles for detection and disposal of mines. In *Proceedings of the SPIE conference on Intelligent Robots and Computer Vision XXIII: Algorithms, Techniques, and Active Vision*, 2005.

[16] A. Gage and R.R. Murphy. Affective recruitment of distributed heterogeneous agents. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence*, pages 14–19, 2004.

[17] A. Ollero, J. Alcázar, F. Cuesta, F. López-Pichaco, and C. Nogales. Helicopter teleoperation for aerial monitoring in the comets multi-uav system. In *Proc. 3rd IARP Workshop on Service, Assistive and Personal Robots*, pages 137–142, 2003.

[18] A. Ollero, S. Lacroix, L. Merino, J. Gancet, J. Wiklund, V. Remuss, IV Perez, LG Gutierrez, DX Viegas, MAG Benitez, et al. Multiple eyes in the skies: architecture and perception issues in the comets unmanned air vehicles project. *Robotics & Automation Magazine, IEEE*, 12(2):46–57, 2005.

[19] L. Jun, G. Zhenbang, and S. Hong. Tri-dimensional system for firefighting and rescuing based on the aerial vehicles and ground mobile robots,. In *Proceedings of the 3rd IARP International Workshop on Service, Assistive and Personal Robots*, 2003.

[20] I. Maza, A. Viguria, and A. Ollero. Aerial and ground robots networked with the environment. In *Proceedings of the Workshop on Network Robot Systems in the IEEE International Conference on Robotics and Automation*, 2004.

[21] I. Maza, A. Viguria, and A. Ollero. Networked aerial-ground robot system with distributed task allocation for disaster management. In *Proceedings of IEEE International Workshop on Safety, Security and Rescue Robotics*, 2006.

[22] R.R. Murphy. Marsupial and shape-shifting robots for urban search and rescue. *IEEE Intelligent Systems and Their Applications*, 15(2):14 – 19, 2000.

[23] D. Spears, D. Zarzhitsky, and D. Thayer. Multi-robot chemical plume tracing. In *Proceedings of the Third International Workshop on Multi-Robot Systems*. Springer, 2005.

[24] D. Zarzhitsky, D.F. Spears, and W.M. Spears. Distributed robotics approach to chemical plume tracing. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4034–4039, 2005.

[25] K.W. Sevcik, W.E. Green, and P.Y. Oh. Exploring search-and-rescue in near-earth environments for aerial robots. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 699–704, 2005.

[26] R.A. Cassanova. Planetary exploration using biomimetics - an entomopter for flight on mars.

Technical report, NASA Institute for Advanced Concepts, 2002.

[27] T.M. Berg and H.F. Durrant-Whyte. Distributed and decentralized estimation. In *Proc. of the Singapore Int'l Conference on Intelligent Control and Instrumentation*, volume 2, pages 1118–1123, 1992.

[28] T.M. Berg and H.F. Durrant-Whyte. General decentralized kalman filters. In *Proc. of the ACC*, volume 2, pages 2273–2274, 1994.

[29] S. Grime and H.F. Durrant-Whyte. Data fusion in decentralized sensor networks. *Control Engineering Practice*, 2(5):849–863, 1994.

[30] S. Utete and H.F. Durrant-Whyte. Reliability in decentralised data fusion networks. In *Proc. of IEEE MFI*, pages 215–221, 1994.

[31] P. Corke, S. Hrabar, R. Peterson, D. Rus, S. Saripalli, and G. Sukhatme. Deployment and connectivity repair of a sensor net with a flying robot. In *Proceedings of the 9th International Symposium on Experimental Robotics*, 2004.

[32] Y. Bar-Shalom. Update with out-of-sequence measurements in tracking: exact solution. *IEEE Trans. on Aerospace and Electronic Systems*, 38(3):769–777, 2002.

[33] Y. Bar-Shalom, H. Chen, and M. Mallick. One-step solution for the multistep out-of-sequence-measurement problem in tracking. *IEEE Trans. on Aerospace and Electronic Systems*, 40(1):27–37, 2004.

[34] P. Ferguson and J. How. Decentralized estimation algorithms for formation flying spacecraft. In *Proc. of AIAA Conference on Guidance Navigation and Control*, 2003.

[35] A. Howard, M.J. Mataric, and G.S. Sukhatme. Localization for mobile robot teams using maximum likelihood estimation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 434–439, 2002.

[36] A. Howard, M.J. Mataric, and G.S. Sukhatme. Putting the'i'in'team': an ego-centric approach to cooperative localization. In *IEEE International Conference on Robotics and Automation*, 2003.

[37] J. Capitán, L. Merino, F. Caballero, and A. Ollero. Delayed-state information filter for cooperative decentralized tracking. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3865–3870, 2009.

[38] C. Stachniss. *Exploration and mapping with mobile robots.* PhD thesis, Universitt Freiburg, 2006.

[39] K. Y. K. Leung, T. D. Barfoot, and H. H. T. Liu. Decentralized localization for dynamic and sparse robot networks. In *IEEE International Conference on Robotics and Automation*, 2009.

[40] K. Y. K. Leung, T. D. Barfoot, and H. H. T. Liu. Decentralized localization of sparsely-communicating robot networks: A centralized-equivalent approach. *IEEE Transactions on Robotics*, 26(1):62–77, 2010.

[41] E. Nerurkar and S. Roumeliotis. Asynchronous multi-centralized cooperative localization. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010.

[42] R. Kurazume, S. Nagata, and S. Hirose. Cooperative positioning with multiple robots. In *IEEE International Conference on Robotics and Automation*, pages 1250–1257, 1994.

[43] I. Rekleitis, G. Dudek, and E. Milios. Probabilistic cooperative localization and mapping in practice. In *IEEE International Conference on Robotics and Automation*, pages 1907–1912, 2003.

[44] I. Rekleitis, G. Dudek, and E. Milios. Experiments in free-space triangulation using cooperative localization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1777–1782, 2003.

[45] R. Kurazume, S. Hirose, S. Nagata, and N. Sashida. Study on cooperative positioning system. In *IEEE International Conference in Robotics and Automation*, pages 1421–1426, 1996.

[46] R. Kurazume and S. Hirose. An experimental study of a cooperative positioning system. *Autonomous Robots*, 8(1):43–52, 2000.

[47] R. Kurazume and S. Hirose. Study on cooperative positioning system: optimum moving strategies for cps-iii. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, pages 2896–2903, 1998.

[48] N. Trawny and T.D. Barfoot. Optimized motion strategies for cooperative localization of mobile robots. In *IEEE International Conference on Robotics and Automation*, pages 1027–1032, 2004.

[49] S. Tully, G. Kantor, and H. Choset. Leap-frog path design for multi-robot cooperative localization. In *International Conference on Field and Service Robotics*, pages 307–317, 2010.

[50] A.C. Sanderson. A distributed algorithm for cooperative navigation among multiple mobile robots. *Advanced Robotics*, 12(4):335–349, 1997.

[51] R. Madhavan, K. Fregene, and L.E. Parker. Distributed heterogeneous outdoor multi-robot localization. In *IEEE International Conference on Robotics and Automation*, pages 374–381, 2002.

[52] D. Fox, W. Burgard, H. Kruppa, and S. Thrun. A probabilistic approach to collaborative multi-robot localization. *Autonomous Robots*, 8(3):325–344, 2000.

[53] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, 2005.

[54] S.I. Roumeliotis and G.A. Bekey. Collective localization: A distributed kalman filter approach to localization of groups of mobile robots. In *IEEE International Conference on Robotics and Automation*, pages 2958–2965, 2000.

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

[56] A. Martinelli, F. Pont, and R. Siegwart. Multi-robot localization using relative observations. In *IEEE International Conference on Robotics and Automation*, pages 2797–2802, 2005.

[57] T. Bailey, M. Bryson, H. Mu, J. Vial, L. McCalman, and H. Durrant-Whyte. Decentralised cooperative localisation for heterogeneous teams of mobile robots. In *IEEE International Conference on Robotics and Automation*, 2011.

[58] E.D. Nerurkar, S.I. Roumeliotis, and A. Martinelli. Distributed maximum a posteriori estimation for multi-robot cooperative localization. In *IEEE International Conference on Robotics and Automation*, 2009.

[59] N. Karam, F. Chausse, R. Aufrere, and R. Chapuis. Localization of a group of communicating vehicles by state exchange. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 519–524, 2006.

[60] S. Panzieri, F. Pascucci, and R. Setola. Multirobot localisation using interlaced extended kalman filter. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2816–2821, 2006.

[61] A. Martinelli. Improving the precision on multi robot localization by using a series of filters hierarchically distributed. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1053–1058, 2007.

[62] S.I. Roumeliotis and I.M. Rekleitis. Propagation of uncertainty in cooperative multirobot localization: Analysis and experimental results. *Autonomous Robots*, 17(1):41–54, 2004.

[63] A.I. Mourikis and S.I. Roumeliotis. Performance analysis of multirobot cooperative localization. *IEEE Transactions on Robotics*, 22(4):666–681, 2006.

[64] I.M. Rekleitis, G. Dudek, and E. Milios. Multi-robot cooperative localization: A study of trade-offs between efficiency and accuracy. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002.

[65] J. Spletzer, A.K. Das, R. Fierro, C.J. Taylor, V. Kumar, and J.P. Ostrowski. Cooperative localization and control for multi-robot manipulation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 631–636, 2001.

[66] L. Montesano, J. Gaspar, J. Santos-Victor, and L. Montano. Cooperative localization by fusing vision-based bearing measurements and motion. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2333–2338, 2005.

[67] A. Bahr, J.J. Leonard, and M.F. Fallon. Cooperative localization for autonomous underwater vehicles. *The International Journal of Robotics Research*, 28(6):714–728, 2009.

[68] A. Bahr, M.R. Walter, and J.J. Leonard. Consistent cooperative localization. In *IEEE International Conference on Robotics and Automation*, pages 3415–3422, 2009.

[69] J.R. Spletzer and C.J. Taylor. A bounded uncertainty approach to multi-robot localization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1258–1265, 2003.

[70] C.J. Taylor and J. Spletzer. A bounded uncertainty approach to cooperative localization using relative bearing constraints. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2500–2506, 2007.

[71] Y. Dieudonné, O. Labbani-Igbida, and F. Petit. On the solvability of the localization problem in robot networks. In *IEEE International Conference on Robotics and Automation*, pages 480–485, 2008.

[72] Y. Dieudonné, O. Labbani-Igbida, and F. Petit. Deterministic robot-network localization is hard. *IEEE Transactions on Robotics*, 26(2):331–339, 2010.

[73] H. Wymeersch, J. Lien, and M.Z. Win. Cooperative localization in wireless networks. *Proceedings of the IEEE*, 97(2):427–450, 2009.

[74] N. Patwari, J.N. Ash, S. Kyperountas, A.O. Hero III, R.L. Moses, and N.S. Correal. Locating the nodes: cooperative localization in wireless sensor networks. *Signal Processing Magazine, IEEE*, 22 (4):54–69, 2005.

[75] N. Trawny, S.I. Roumeliotis, and G.B. Giannakis. Cooperative multi-robot localization under communication constraints. In *IEEE International Confererence on Robotics and Automation*, 2009.

[76] A.H. Jazwinsky. *Stochastic Processes and Filtering Theory.* Academic Press, Inc., 1970.

[77] Z. Brzeźniak and T. Zastawniak. *Basic Stochastic Processes: A Course Through Exercises.*

Springer, 1999.

[78] Y. Bar-Shalom, X.R. Li, and T. Kirubarajan. *Estimation with applications to tracking and navigation.* Wiley New York, 2001.

[79] Dan Simon. *Optimal State Estimation.* John Wiley and Sons, 2006.

[80] S.O. Lee, Y.J. Cho, M. Hwang-Bo, B.J. You, and S.R. Oh. A stable target-tracking control for unicycle mobile robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 822–1827, 2000.

[81] R. Siegwart and Nourbakhsh. *Introduction to Autonomous Mobile Robots.* MIT Press, 2004.

[82] E. Buckingham. On physically similar systems; illustrations of the use of dimensional equations. *Physical Review*, 4(4):345–376, 1914.

[83] M. Sanchez, P. Manzoni, and Z.J. Haas. Determination of critical transmission range in ad-hoc networks. In *Multiaccess, Mobility and Teletraffic for Wireless Communications*, 1999.

[84] B. Bollobas. *Random Graphs.* Academic Press, 1985.

[85] B. Krishnamachari, SB Wicker, and R. Bejar. Phase transition phenomena in wireless ad hoc networks. In *IEEE Global Telecommunications Conference*, 2001.

[86] J. Frank and C.U. Martel. Phase transitions in the properties of random graphs. In *Principles and Practice of Constraint Programming*, pages 62–69, 1995.

[87] Zhimin Jiang, Sen Zhang, and Lihua Xie. Cramer-rao lower bound analysis for mobile robot navigation. In *Proceedings of the International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pages 229–234, 2005.

[88] K. Y. K. Leung, T. D. Barfoot, and H. H. T. Liu. Decentralized simultaneous localization and mapping for dynamic and sparse robot networks. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010.

[89] K. Y. K. Leung, T. D. Barfoot, and H. H. T. Liu. Decentralized cooperative slam for sparsely-communicating robot networks: A centralized-equivalent approach. *[Accepted in] Journal of Intelligent Robotic Systems*, 2011.

[90] J.W. Fenwick, P.M. Newman, and J.J. Leonard. Cooperative concurrent mapping and localization. In *IEEE International Conferernce on Robotics and Automation*, 2002.

[91] E.W. Nettleton, H.F. Durrant-Whyte, P.W. Gibbens, and A.H. Goktogan. Multiple platform localisation and map building. *Proc. SPIE*, 4196:337–347, 2000.

[92] M. Rosencrantz, G. Gordon, and S. Thrun. Decentralized sensor fusion with distributed particle filters. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2003.

[93] E. Nettleton, S. Thrun, H. Durrant-Whyte, and S. Sukkarieh. Decentralised slam with low-bandwidth communication for teams of vehicles. *Field & Service Robotic*, 24:179–188, 2006.

[94] S. Reece and S. Roberts. Robust, low-bandwidth, multi-vehicle mapping. In *IEEE FUSION*, 2005.

[95] S. Thrun, Y. Liu, D. Koller, A. Ng, Ghahramani Z., and H. Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *International Journal of Robotics Research*, 23:693, 2004.

[96] J. Ko, B. Stewart, D. Fox, K. Konolige, and B. Limketkai. A practical, decision-theoretic approach

to multi-robot mapping and exploration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.

[97] X.S. Zhou and S.I. Roumeliotis. Multi-robot slam with unknown initial correspondence: The robot rendezvous case. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[98] Z. Wang, S. Huang, and G. Dissanayake. Multi-robot simultaneous localization and mapping using d-slam framework. In *IEEE ISSNIP*, 2007.

[99] A. Howard, G.S. Sukhatme, and M.J. Matarić. Multirobot simultaneous localization and mapping using manifold representations. *IEEE Special Issue on Multi-robot Systems*, 94(9):1360–1369, 2006.

[100] A. Howard. Multi-robot simultaneous localization and mapping using particle filters. *International Journal of Robotics Research*, 25(12):1243–1256, 2006.

[101] Z. Wang and G. Dissanayake. Observability analysis of slam using fisher information matrix. In *IEEE International Conference on Control, Automation, Robotics and Vision*, pages 1242–1247, 2008.

[102] K. Y. K. Leung, Y. Halpern, T. D. Barfoot, and H. H. T. Liu. The utias multi-robot cooperative localization and mapping dataset. *The International Journal of Robotics Research*, 30(8):969–974, 2011.

[103] B. Gerkey, R.T. Vaughan, and A. Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. In *international conference on advanced robotics*, pages 317–323, 2003.

[104] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *International Conference on Computer Vision*, pages 666–673, 1999.

[105] M. Windolf, N. Götzen, and M. Morlock. Systematic accuracy and precision analysis of video motion capturing systems - exemplified on the vicon-460 system. *Journal of Biomechanics*, 41 (12):2776–2780, 2008.

[106] T. Bailey, J. Bieto, J. Guivant, M. Stevens, and E. Nebot. Consistency of the ekf-slam algorithm. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[107] T. Bailey and H. Durrant-Whyte. Simultaneous localization and mapping (slam): Part ii. *IEEE Robotics and Automation Magazine*, 13(3):108–117, 2006.

[108] Y. Bar-Shalom. *Tracking and Data Association.* Academic Press, Inc., 1988.

[109] K. Y. K. Leung, T. D. Barfoot, and H. H. T. Liu. Distributed and decentralized cooperative simultaneous localization and mapping for dynamic and sparse robot networks. In *IEEE International Conference on Robotics and Automation*, 2011.

[110] I.D. Schizas, G.B. Giannakis, and Z.Q. Luo. Distributed estimation using reduced-dimensionality sensor observations. *IEEE Transactions on Signal Processing*, 55(8):4284–4299, 2007.

[111] C.S. Chen, Y.P. Hung, and J.B. Cheng. Ransac-based darces: A new approach to fast automatic registration of partially overlapping range images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(11):1229–1234, 2002.

[112] J. Neira and J.D. Tardós. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17(6):890–897, 2001.