Title
CoLo: A Performance Evaluation System for Multi-robot Cooperative Localization Algorithms

Author
Chen, Shengkang

Publication Date
2019

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

CoLo: A Performance Evaluation System for Multi-robot Cooperative Localization Algorithms

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Electrical and Computer Engineering

by

Shengkang Chen

2019

ABSTRACT OF THE THESIS

CoLo: A Performance Evaluation System
for Multi-robot Cooperative Localization Algorithms

by

Shengkang Chen
Master of Science in Electrical and Computer Engineering
University of California, Los Angeles, 2019
Professor Ankur M. Mehta, Chair

This thesis presents CoLo - a performance evaluation system for two-dimensional cooperative localization algorithms. Multi-robot system have been used in a wide range of applications and cooperative localization is one of the fundamental tasks for mobile multi-robot systems. However, developing cooperative localization algorithm is complex and time-consuming. CoLo is created to reduce cooperative localization algorithm development cycle time. The system consists of two main parts: a physical experiment (CoLo-PE) for data collection and a software analysis tool (CoLo-AT) using real-world datasets to evaluate the performances of users' cooperative localization algorithms. CoLo uses an intuitive algorithm framework to allow researchers to conveniently add their cooperative localization algorithms to it. Instead of creating simulations or designing a new robotic testbed from the ground up, researchers only needed to load their algorithms in CoLo-AT and analyze them using data collected from CoLo-PE. Also, CoLo is aimed to create a standard so that effective comparisons can be made across research on localization algorithm.

This paper details the design and operation of the physical experiment (CoLo-PE) which provides users guidelines to create their own robotic testbed with a ROS-based, scalable and affordable robotic team. And the paper explains how the software analysis tool (CoLo-AT) tests algorithms by running simulation processes to recreate the experiment trials using compatible real-world datasets of odometry data, measurement data, and the related groundtruth

data. Researchers can test their algorithms and compare them with other state-of-the-art algorithms in various settings. CoLo-AT provides insightful metrics, and graphs include location error for localization accuracy and trace of state covariance to detect over-confident estimation results. It also has an animated plot to show the estimated trajectory and actual trajectory of each robot, which presents an intuitive visualization of the algorithm.

CoLo has been used in the development of a published localization algorithm, where it was used to test the performance of the algorithm and compared to other existing algorithms objectively. CoLo provided the performance results in various setting and saved much time in experimental validation, which enabled a more rapid algorithm design process.

CoLo is available at *https://git.uclalemur.com/billyskc/CoLo*

The thesis of Shengkang Chen is approved.

Gregory J. Pottie

Christina Panagio Fragouli

Ankur M. Mehta, Committee Chair

University of California, Los Angeles

2019

This dissertation is dedicated to my beloved parents Zhongming Lyu and Yaobang Chen, for their endless support and invaluable education opportunities.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would first like to thank my thesis adviser Prof. Ankur Mehta of Electrical and Computer Engineering (ECE) department at University of California, Los Angeles (UCLA). Prof Mehta introduced me to the world of robotics and always encouraged me to find problems and then try to solve them.

I would like to thank Prof. Christina Fragouli of Electrical and Computer Engineering (ECE) department at University of California, Los Angeles (UCLA). Prof. Fragouli accepted me to her lab when I was an undergraduate student. She was always patient to me and gave me guidelines on how to be a researcher.

I would like to thank Prof. Gregory Pottie of Electrical and Computer Engineering (ECE) department at University of California, Los Angeles (UCLA). Prof. Pottie was my undergraduate captone project adviser who was always kind to me and gave me instructful advises on not only my project but also my research.

I am also grateful to everyone in Algorithmic Research in Network Information Flow (ARNI) and the Laboratory for Embedded Machines and Ubiquitous Robots (LEMUR) in the Electrical and Computer Engineering (ECE) Department of the University of California, Los Angeles (UCLA), especially Tsang-kai Chang, Gaurav Agarwal and Martina Cardone for their help and support.

Finally, I would like to mentions Ben Limpanukorn and Wenzhong Yan from the University of California, Los Angeles (UCLA) and Clara Chun from Notre Dame Academy for their help on the physical experiments part (CoLo-PE).

CHAPTER 1

Introduction

Robotic localization is the problem of estimating a robot's own pose. The development of multi-robot systems has attracted more and more attention for its wide range of potential applications from underwater exploration to disaster rescue missions [STS16]. Since these applications require accurate information on robots' poses, cooperative localization for multi-robot systems has developed from single-robot localization. Cooperative localization provides positioning data for a multi-robot system, which often relies on its robots' sensors and communication within the system.

With the growing need for cooperative localization, different cooperative localization algorithms have been developed [MR06, WMG14]. However, the development of cooperative localization algorithms from formalization to experimental validation for performance evaluation is complicated and time-consuming. As a result, CoLo has been developed to help researchers validate algorithms by providing the framework to set up a complete evaluation system, in which users only need to provide their algorithms. CoLo is a performance evaluation system using real-world datasets for two-dimensional multi-robot cooperative localization algorithms. CoLo has two modular parts: a physical environment setup (CoLo-PE) and software algorithm analysis tool (CoLo-AT). The structure of CoLo is captured in Figure 1.1, where CoLo-PE provides real-world datasets for CoLo-AT to evaluate the performance of users' cooperative localization algorithms.

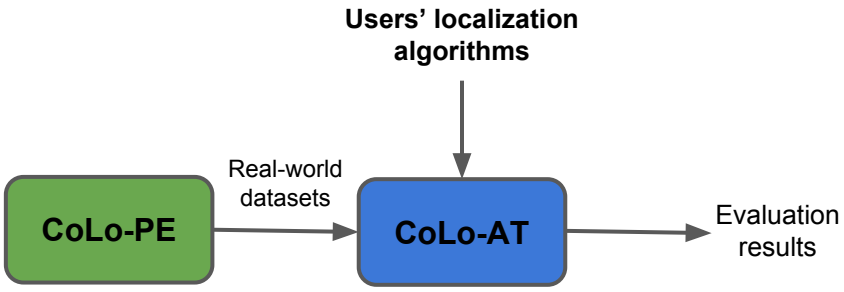


Figure 1.1: CoLo Structure

The main goal of CoLo is to provide researchers a toolbox to test their algorithms efficiently and conveniently. As a result, we built a physical environment setup (CoLo-PE) for real-world data collections so that researchers can evaluate their algorithm not only using simulated data but also real-world data for more realistic performance results. Also, we created an analysis tool (CoLo-AT) to enable users to test their algorithms using compatible real-world datasets without spending much time to build their own. Moreover, since CoLo-PE and CoLo-AT are two independet modules that can be used separately, researchers can use CoLo-AT to assess their algorithm using publicly available real-world datasets without building CoLo-PE. The objective of CoLo is to make research on cooperative localization more accessible.

CHAPTER 2

Background

Experiment validation is critical for multi-robot localization algorithm development but can be difficult to set up. CoLo aims to provide an accessible way for users to test their algorithms effectively using real-world data. Typically, researchers have built the whole experiment setup in their labs to evaluate the performance of their algorithms [RB02, MR06, HDJ10, CNG13]. These setups usually contain a team of multiple robots with external sensors to take measurements, internal sensors to collect odometry data as well as equipment for groundtruth collection [MR06, HDJ10]. However, thees experiment setups are expensive and complicated to develop since there are many components in the experiment that need to work in coordination. The cost and time investment associated with physical experiemnt setups make it less accessible to many researchers. As a result, some researchers tested their algorithm on simulations [LN12, LLS16, WMG14, KCA06]. Although simulation results can provide a general idea for the performance of cooperative localization algorithms, it is difficult to simulate all the issues related to the multi-robot system accurately such as the effects of asynchronous control.

To address this problem, there are several remotely accessible testbeds like Robotarium [PGW17] and the HoTDoc [SVF06] for multi-robot systems available for researchers, but using these remotely accessible testbeds may not be the optimal solutions. The general limitation of these testbeds is the limited selections of sensors and configurations of the robots. Since testbeds like the Robotarium [PGW17] or the HoTDoc [SVF06] focus on coordinated control, their robots do not have the appropriate sensors for testing localization algorithms. Researchers can also build or purchase robots for multi-robot localization from some existing multi-robot testbeds [RGU13, MFK08]. A detailed list of multi-robot testbeds

can be found in this survey paper [JDO13]. Different from these robots, CoLo's robots use off-the-shelf products instead of customized parts for greater availability and upgradability. CoLo's robots can have processors with higher processing rates that make them more suitable for compute-intensive tasks like image-processing when robots observe their surroundings. More importantly, CoLo's integrated software analysis tool can extract data from its robots and evaluate different localization algorithms directly.

Other than using testbeds, researchers can use real-world datasets collected from experiments to evaluate their cooperative localization algorithms. Using datasets is more convenient for researchers to test the performance of their algorithm compared with testbeds, but the main drawback of using dataset is that researchers have limited options on experiment settings. Although there are different datasets to study different localization problems like vision localization [PC09], there is a limited number of datasets for multi-robot systems like the UTIAS datastes [LHB11] available to researchers [CCC16]. To address these problems, CoLo provides both the physical experiment setup (CoLo-PE) to help researchers create the dataset they need to evaluate their algorithms. If they can't create their own setups, the dataset collected by the CoLo-PE in our lab to expand the selection of multi-robot localization dataset available. Moreover, CoLo contains the software analysis module which can directly use the datasets and save users time creating their own analysis tool.

What makes CoLo different from existing testbeds or datasets is that CoLo is a complete performance evaluation system for two-dimensional localization algorithms from physical experiment to algorithm analysis, which only requires users' algorithms formulation within the CoLo's algorithm framework.

CHAPTER 3

CoLo Overview

CoLo is designed to evaluate the performance of two-dimensional localization algorithms. CoLo is composed of 2 modular parts: CoLo-PE (the physical experiment setup) and CoLo-AT (the software analysis tool). CoLo-PE sets up the physical experiment to collect multi-robot localization data, which will be processed and organized as datasets. Then, CoLo-AT will load these datasets from CoLo-PE and test users' algorithms. CoLo-AT can be used independently by using compatible datasets including the UTIAS multi-robot cooperative localization and mapping dataset [LHB11].

In CoLo, a multi-robot system uses a cooperative localization algorithm to update its estimated states which includes its robots' locations: $s = [x_1, y_1, x_2, y_2, \dots]$ and their orientations: $\Theta = [\theta_1, \theta_2, \dots]$; as well as its state covariance Σ_s which indicates how confident the algorithm is in its estimation. The algorithm itself can be either centralized or distributed. The multi-robot system performs localization updates using data from robots' four different operations:

1. *Propagation:* The robots' internal sensors like odometry sensors provide odometry data continuously.

2. *Landmark observation:* The robots' external sensors like cameras provide observation data when they observe landmarks

3. *Relative observation:* The robots' external sensors like cameras provide observation data when they observe other robots from the multi-robot system

4. *Communication:* Robots from the multi-robot system send data to each other using a communication network.

3.1 System Model

We consider a 2D multi-robot system with N robots denoted by $R = \{1, \ldots, N\}$, together with several landmarks with known known locations to the robots in advance. Landmarks are denoted as L, and $\Omega = R \cup L$. The position of robot i at time t is regarded as the state, denoted as $s_{i,t} = [x_{i,t}, y_{i,t}]^\intercal$, where $\intercal$ denotes matrix transpose. The orientation of robot i at time t is denoted by $\theta_{i,t}$, and we do not incorporate $\theta_{i,t}$ in the estimation state due to the linearization issue [BNG06]. The state of the whole system is denoted by $s_t = [s_{1,t}^\intercal, \ldots, s_{N,t}^\intercal]^\intercal$. In EKF, each robot i keeps an estimate of the whole system s_t, denoted by $\hat{s}_t^{(i)}$, together with its covariance $\Sigma_{s^{(i)},t}$.

3.1.1 Motion Model

The motion model describes the spatial displacement of robots due to odometry inputs. While the framework is not limited to any specific models, we use velocity input $v_{i,t}$ and orientation $\theta_{i,t}$ in this paper. Let T_p be the time interval between two consecutive odometry inputs, the state of robot i at the next time is given by

$$s_{i,t}^{(i)} = f_i(s_{i,t}^{(i)}, v_{i,t}) = \begin{bmatrix} x_{i,t} + v_{i,t} T_p \cos(\theta_{i,t}) \\ y_{i,t} + v_{i,t} T_p \sin(\theta_{i,t}) \end{bmatrix}. \tag{3.1}$$

3.1.2 Observation Model

If robot i observes an object j (either a robot or a landmark), the relative observation obtained by robot i is

$$o_{ij} = C^\intercal(\theta_{i,t}) \left(\begin{bmatrix} x_{j,t} \\ y_{j,t} \end{bmatrix} - \begin{bmatrix} x_{i,t} \\ y_{i,t} \end{bmatrix} \right) = h(s_t^{(i)}), \tag{3.2}$$

where $C(\theta)$ is the rotation matrix with argument θ.

Most of the time, the relative positions can not be obtained directly, but they are general enough to incorporate different kinds of sensing result. The observation is often accomplished by distance and bearing sensors. Consequently, the relative position can also be expressed

as

$$o_{ij} = d_{ij} \begin{bmatrix} \cos(\phi_{ij}) \\ \sin(\phi_{ij}) \end{bmatrix}, \tag{3.3}$$

based on the relative distance d_{ij} and relative bearing ϕ_{ij}.

3.2 algorithm framework

For users to add and evaluate different localization algorithms into CoLo conveniently, we have created an algorithm framework based on robots' four operations for users to define their localization functions.

n = number of robots in the team

$s^{(i)}$ = $robot_i$ state vector

$\Sigma_{s^{(i)}}$ = $robot_i$ state covariance matrix (m^2)

Θ_i = $robot_i$ estimated orientations of the team

δ_t = duration (s)

v = linear velocity (m/s)

w = angular velocity (rads/s)

θ_i = $robot_i$ own orientation (rad)

lm = landmark location [x (m), y (m)]

r = measurement range (m)

ϕ = measurement bearing (rad)

Algorithm 1 CoLo Algorithm Framework

Initialization

For each $robot_i$:

user-defined initialization function

Return : $s^{(i)}, \Sigma_{s^{(i)}}, , \Theta_i$

Propagation Update

For $robot_i$:

Given : $s^{(i)}, \Sigma_{s^{(i)}}, \Theta_i, i, \delta_t, v, \theta_i, w$

user-defined propagation function

Return : $s^{(i)}, \Sigma_{s^{(i)}}, \Theta_i$

Landmark Observation Update

For $robot_i$:

Given : $s^{(i)}, \Sigma_{s^{(i)}}, \Theta_i, i, lm, r, \phi$

user-defined landmark observation function

Return : $s^{(i)}, \Sigma_{s^{(i)}}, \Theta_i$

Relative Observation Update

For $robot_i$ *detects* $robot_j$

Given : $s^{(i)}, \Sigma_{s^{(i)}}, \Theta_i, i, j, r, \phi$

user-defined relative observation function

Return : $s^{(i)}, \Sigma_{s^{(i)}}, \Theta_i$

Communication Update

For $robot_i$ *recives from* $robot_j$

Given : $s^{(i)}, \Sigma_{s^{(i)}}, \Theta_i, i, s_j, \Sigma_{sj}, \Theta_j, j$

user-defined communication function

Return : $s^{(i)} i, \Sigma_{s^{(i)}}, \Theta_i$

CHAPTER 4

Physical Experiment Setup: COLO-PE

CoLo-PE is the physical experiment part of CoLo. It provides guidelines for setting up a physical robotic testbed for data collection with a scalable and affordable robot team. Each robot in CoLo is built with standardized components and shares the same software structure so that users can easily create a robot and add it to the team. CoLo-PE contains a team of robots, a group of landmarks and a motion tracking system to keep track of locations of each landmark and each robot during the experiment for groudtruth data. The Network Time Protocol (NTP) is used to synchronize the time between each robot and the groundtruth logging computer with its error on the scale of 1 ms, which is acceptable for the experiment. An experiment run is captured in Figure 4.1, where a team of three robots moved within the setup and used their onboard camera to take measurements by detecting the ArUco [GMM14] markers on other robots or landmarks, while these robots and landmarks are tracked by the motion tracking system for collecting groundtruth data.

4.1 Robot system

The robot system in CoLo-PE is designed to be configurable and economical. Each robot has four main components: a transportation platform, a computer, a battery and a camera. We chose the iRobot Create2 as the transportation platform that supports up to 2 kg payload, which is enough for a laptop or other smaller computers. This transportation platform gives users the freedom to build different performing robots. The components are commercially available products rather than customized parts for robots to be economical. These components are connected through standard interfaces like the universal serial bus (USB). This

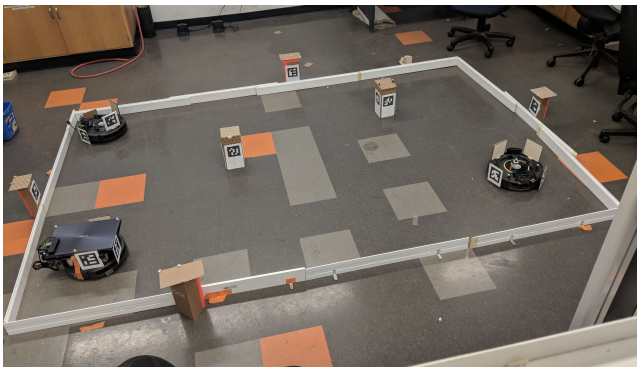


Figure 4.1: An experiment trial in CoLo-PE

enables users to create robots with different hardware configurations and upgrade them in the future easily.

There are four main components in general robotic setup in CoLo-PE: Transportation platform, a computer running Ubuntu, a camera and a battery for the computer. Users can configure different robots in CoLo-PE for their own needs. There are two types of robot designs available in our lab with different computers as two possible instantiations of CoLo on physical robots: type I with a full-size laptop and type II with a much smaller compute stick and an external battery. Compared with type II robots, type I robots have better performance, but they need additional structure to support their laptops, which makes it bulkier and harder to maneuver between landmarks. Instead of using full-size laptops, we selected compute sticks as the type II robots' computers. The compute stick and the batteries are stored in the dustbin in the back of each iRobot Create2 to avoid any additional support structures. Table 4.1 and Table 4.2 shows the details of components used in both types of robots with estimated costs.

Since Robot Operating System (ROS) [QCG09] is a widely used robotic middleware which has a large selection of packages available, developers can easily find the packages they need to accelerate their development process. ROS Kinetic is chosen to be the software framework for the robots. In CoLo-PE, robots can perform two kinds of actions and record propagation and measurement data. Table 4.3 shows the list of ROS packages used in CoLo-PE's robots. We developed several ROS packages can combine them with some public available ROS

Table 4.1: Hardware components in type I robot

Component	**Product name**	**Price**
Transportation platform	iRobot Create2	$200
Computer	Asus F510UA Laptop	$510
Camera	Logitech Webcam C920	$50
Accessories	USB hub	$10
Total Cost		$760

Table 4.2: Hardware components in type II robot

Component	**Product name**	**Price**
Transportation platform	iRobot Create2	$200
Computer	Azulle Access Plus PC Stick	$200
Camera	Logitech Webcam C920	$50
Battery	Anker PowerCore 13000	$30
Accessories	USB hub	$10
Total Cost		$490

packages for the software structure of each robots. Users can easily modified our packages for different tasks like different robotic paths instead of pseudo random movements.

For propagation, robots are controlled through a ROS driver to perform pseudo-random movements with barrier avoidance. Command inputs on linear velocity, v, and angular velocity, w, are logged as odometry data around 10 Hz in the following format: [time, velocity, angular velocity].

For observation measurement, CoLo-PE uses ArUco [GMM14] markers which are widely used for camera pose estimation for both landmark observations and relative observations. The camera (resolution: 1280 by 720, the field of view: 78 degrees) on each robot will process images to detect these markers at around 10 Hz. When an ArUco maker is detected, robots

Table 4.3: ROS Packages used CoLo-PE

ROS package	**Function**
create_autonomy [Per16]	iRobot Create drivers
robot_control	motion control and odometry data recording
usb_cam [PT14]	webcam driver
aruco_detect [Vau18]	ArUco markers detection
meas_record	measurement data recording
bt_network (optional)	Bluetooth communication network setup

will log the measurement data in the following format: [time, marker ID, range, bearing].

For communication, robots in CoLo-PE can communicate with others using Bluetooth or radio module. Although the robots could set up a reliable peer-to-peer communication network in a small experiment via Bluetooth, they failed to provide accurate link quality metrics (LQ) for users to study the topology of the communication networks. Currently, we are working on using RF modules for inter-robot communication and measurement link quality metrics (LQ) and receive signal strength indicator (RSS)I for each message sending.

4.2 Groundtruth

In order to evaluate the performance of localization algorithms accurately, a high accuracy tracking system is needed to track the locations of the robots and the landmarks. An Optitrack motion capture system is deployed to capture the 2-dimensional grouthtruth data for the robots [time, x, y, orientation] and the landmarks [time, x, y] at around 120 Hz.

4.3 Dataset

CoLo-PE can create datasets with arbitrary numbers of robots and landmarks for arbitrary duration depending on the size of the experiment setup. We used CoLo-PE to provide a

multi-robot cooperative localization dataset collected by in a 3 m by 4 m indoor rectangular space with three robots and eight landmarks. There are four subdataset in the CoLo dataset with different landmark positions. The duration of each subdataset is approximately 5 minutes long. Each subdataset contains 18 files, which include:

- One CSV file from Optitrack.
- One label file for robots' markers.
- One groundtruth files for the landmarks.
- Three groundtruth files for the robots.
- Three odometry files for the robots.
- Three raw measurement files for the robots.
- Three processed measurement files for the robots.

Landmarks and robots are considered subjects in the dataset. Each landmark or robot is assigned a unique subject ID, where robots' subject IDs are different from their marker IDs. The three raw measurement files for the robots have the measurement data format: [time, marker ID, range, bearing] with respect to their camera; whereas processed measurement files contain measurement data format [time, subjects ID, range, bearing] with respect to the centers of the robots using the label file to transform some of the marker IDs to robot IDs.

CHAPTER 5

Analysis Tool: COLO-AT

CoLo-AT as the core of CoLo is a Pyhton-based localization algorithm analysis tool using real-world datasets to evaluate the performance of different localization algorithms. CoLo-AT is aimed at helping researchers develop cooperative localization algorithms by providing insightful algorithm evaluation data. It can test the performance of cooperative localization algorithms on various settings by running a simulation process to recreate the experiment trials using compatible real-world datasets including the UTIAS dataset [LHB11]. If there is no communication data in the dataset, users can define their communication scheme for the simulation.

5.1 CoLo-AT structure

CoLo-AT itself is highly modularized, and users can edit or add new modules to fit their needs. The structure of CoLo is captured in Figure 5.1. There are six main modules in CoLo-AT:

- Dataset manager (DM): reads a compatible dataset and puts it in a proper data structure for convenient lookup so that it can communicate with the simulation manager efficiently. If there are datasets with different formats, users only need to modify the dataset manager and other parts of CoLo-AT will remain the same.
- Simulation manager (SM): manages simulation process by communicating between different modules. It has two modes: native and schedule. In native mode, robots will perform actions based on the chronological order of the dataset which will mimic

the operations taken in the dataset exactly. In schedule mode, users can control the frequency for each operation type, which provides more flexibility on robot control.

- State recorder (SR): records the robots's data after each localization update, including their estimated locations, state covariances, groundtruth (actual locations), operation type.

- Robot system (RS): a multi-robot system uses data provided by the dataset manager for each robotic operation to update its estimated states based on its loaded localization algorithm (LA). There are two different types of robot system: it can be a team of distributed robots with a distributed cooperative localization algorithm or a centralized system using a centralized cooperative localization algorithm.

- Algorithm library (AL): an extendable localization library provides the robot system with different localization algorithms.

- Result analyzer (RA): compares estimated results with the groundtruth to generate graphs and numerical values of the performance of the localization algorithms after a simulation process is finished.

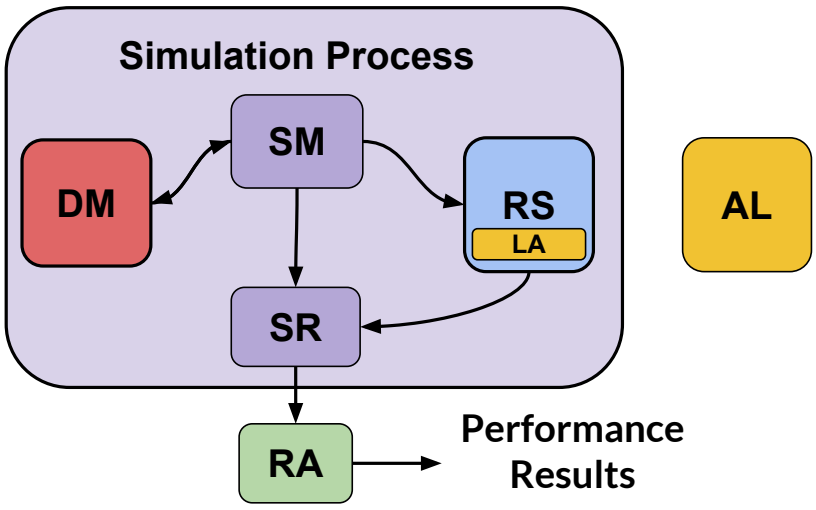


Figure 5.1: CoLo-AT's modularized structure

5.2 Evaluation Process

In order to evaluate localization realistically, CoLo-AT uses real-world datasets to recreate the experimental trails for multi-robot systems. As a result, users don't need to run physical experiments and still obtain realistic algorithm performance results.

The simulation manager and the dataset manager communicate based on a request-response mechanism, where the dataset manager interacts with different datasets and the simulation manager is in charge of the simluation process. During each update in a simulation process, the simulation manager will request a set of data for this update from dataset manager. Then, the dataset manager will provide the data needed by responding to the request. The robot system will update its estimated states based on the data provided by the simulation manager using its localization algorithm. After that, the state recorder will record the robots' states. When the simulation process is completed, the state recorder will send all of its data to the analyzer which provide performance evaluation results.

There are two modes of simulation process: naive (passive) mode and scheduling modes. In naive (passive) mode, the simulation manager will receive all the data from the dataset manager and let the robot system perform different operation update depends on the data provided by the dataset manager. This is suitable for localization algorithm that have no requirement on the order of the operations. In scheduling mode, the simulation manager will receive a subset the data from the dataset manager based on the frequency of each operations and let the robot system perform different operation update coordinately. Scheduling mode is designed for localization algorithm that have no requirement on the order of and frequency of each operation type. Users can also use scheduling mode to study the optimal scheduling problems in cooperative localization.

5.3 Algorithm Library

The library already includes six popular or state-of-art algorithms for algorithm evaluation and comparison. These can be categorized into two types: local state algorithms which are

implemented by a centralized robot system where each robot only need to have estimated localization results for itself; and global state algorithms which are implemented by distributed robots where each robot needs to have estimated localization results for itself and other robots. Here is the list of algorithms in CoLo-AT library:

- Extended Kalman Filter (EKF) [WB06]: a standard EKF-based localization algorithm for each robot itself.
- Local State Centralized extended Kalman Filter (LS-Cen) [KRM14]: a centralized EKF-based localization algorithm for the centralized robot system.
- Local State covariance Intersection (LS-CI) [CNG13]: a covariance Intersection (CI)-based algorithm for the centralized robot system.
- Local State Block Diagonal Approximation (LS-BDA) [LSR16]: an EKF-based localization algorithm with block diagonal approximation on covariance matrix for the centralized robot system.
- Global State Split Covariance Intersection (GS-SCI) [LN12]: a cooperative localization algorithm base on split covariance intersection filter for distributed robots.
- Global State Covariance Intersection (GS-CI) [CCM17]: a distributed cooperative EKF-based localization algorithm using covariance intersection.

CHAPTER 6

Using CoLo

Using CoLo is straight-forward and users will get insightful results of their algorithms. After users have created a cooperative localization algorithm within CoLo's algorithm framework, they can easily load it into the robot system and run CoLo-AT using different datasets with various setting to test the algorithm for different scenarios.

There are two main indicators for localization algorithms: location error and trace of state covariance. Location error is defined as $\frac{1}{N}\sqrt{\sum_{i=1}^{N} ||\hat{s}_i^{(i)} - s_i^{(i)}||^2}$ (how much estimated locations deviate from actual locations) and trace of state covariance is defined as $tr(\Sigma_{s^{(i)}})$ (how uncertain robots are for their estimated locations).

CoLo offers a various plots and metrics based on these two indicators to show the performance of the cooperative localization algorithm. Here are some of the algorithm evaluation options from CoLo-AT.

- An animated plot to recreate robots' estimated tracks with their groundtruth. Figure 6.1 shows the estimation deviation error (how much estimated locations deviate from actual locations), trace of state covariance and the track including both estimated locations and actual locations for each robot in the multi-robot system
- A plot of location error and the trace of state covariance vs. time for each robot.
- A plot of location error and the trace of state covariance vs. time for the multi-robot system.
- A histogram of the number of robotic operations vs. time for the multi-robot system.

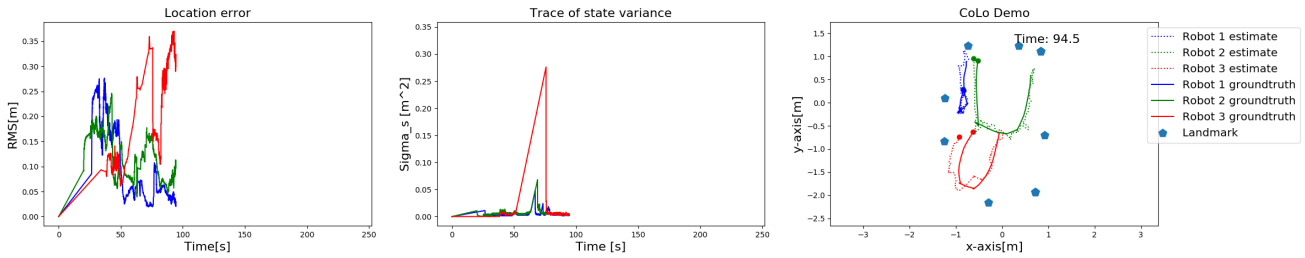


Figure 6.1: A snapshot of the animated plots for a multi-robot system with three robots with centralized extended Kalman filter cooperative localization algorithm (LS-Cen) [KRM14] after 94.5 s from simulation start using CoLo Dataset 4

6.1 Algorithm Development and Evaluation

The main goal of CoLo is to provide localization algorithm developers a toolbox for convenient and effective algorithm evaluation. Running CoLo is simple, users only need to load the algorithm into the robot sysem, specify the which dataset to use and the duation they want to test their algorithms. Also, there are various setting users can choose from from robot types to different performance plots. Instead of letting developers to design and building a simulation environment for their algorithms, developer can use CoLo effortlessly and collect more realistic performance results.

CoLo has been used in a published paper for algorithm developemnt [CCM17]. We used CoLo to help develop our extended Kalman Filter (EKF) based global state covariance-intersection (GS-CI) cooperative localization algorithm with explicit communication updates. It decouples communication update and relative observation. It achieves great performance under limited communication scenario. We have used CoLo to test the performance of GS-CI cooperative localization algorithm in different real-world datasets. In Figure 6.2, a team of 3 distributed robots achieves less than 0.12 m location errors in a sparse communication scenario using CoLo dataset 1 collected by CoLo-PE.

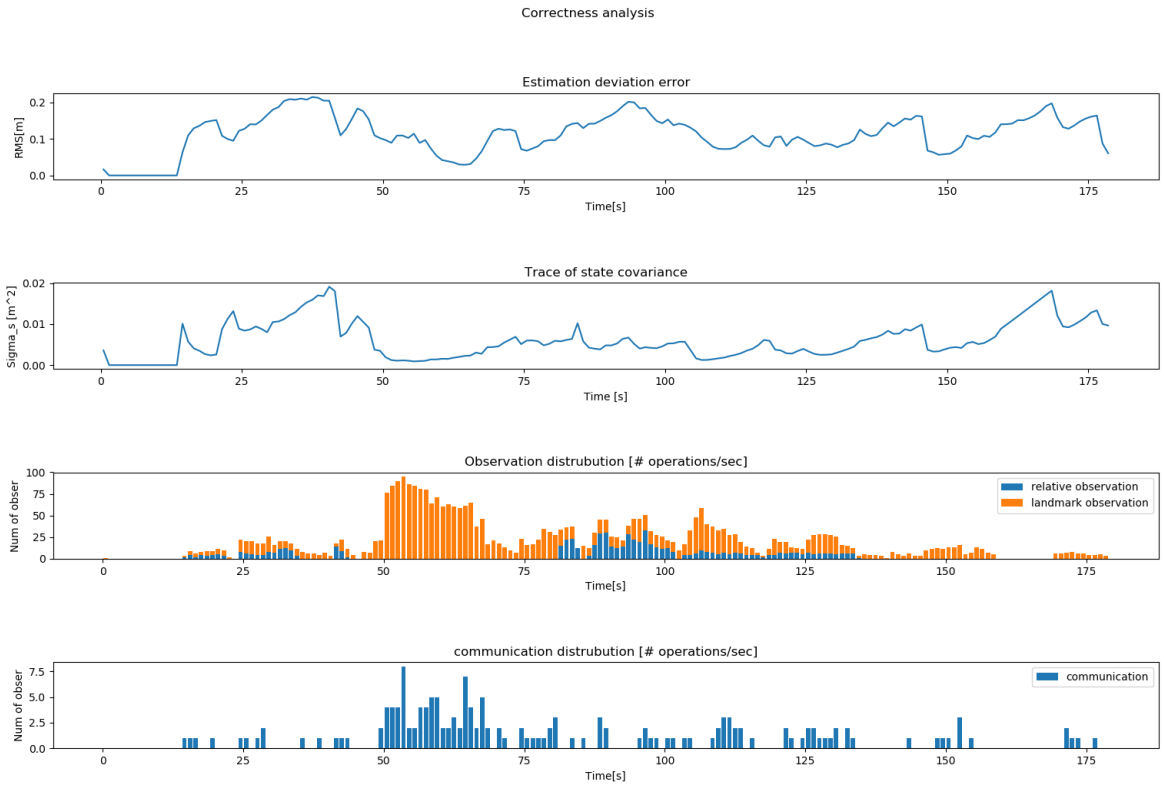
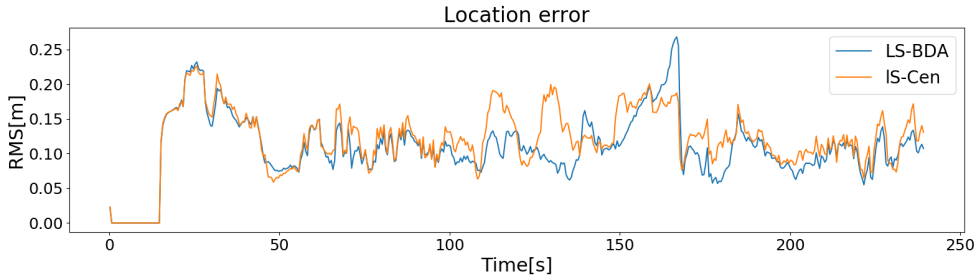


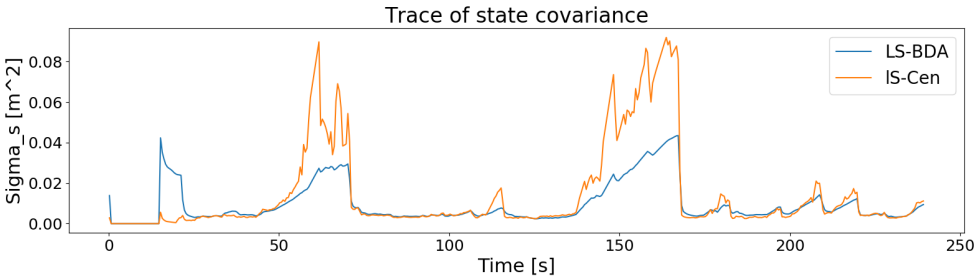
Figure 6.2: Performance of based global state covariance-intersection (GS-CI) cooperative localization algorithm in CoLo Dataset 1

6.2 Algorithm Comparison

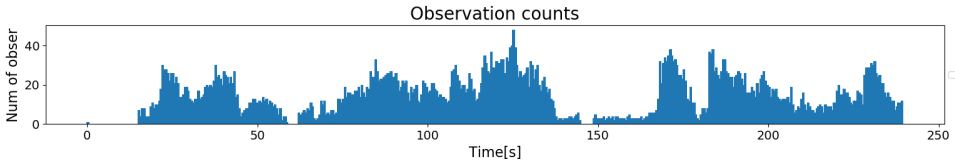
Other than showing the performance results of a single cooperative localization algorithm, users can have CoLo-AT compare the performance between any algorithms that are included in CoLo's algorithm framework. Here is an example of using CoLo-AT to compare the performance of Local state block diagonal approximation cooperative localization algorithm (LS-BDA) [LSR16] with centralized extended Kalman filter cooperative localization algorithm (LS-Cen) [KRM14]:



(a) How far away are the multi-robot systems' estimated locations from its actual locations with respect to time.



(b) Trace of the multi-robot systems' state covariance matrices.



(c) Number of observations including(landmark observation and relative observation) taken for the multi-robot during the simulation process.

Figure 6.3: Some of the plots that can be used for comparing the performances of two cooperative localization algorithms: Local state block diagonal approximation (LS-BDA) [LSR16] and centralized extended Kalman filter (LS-Cen) [KRM14].

These results show that both local state block diagonal approximation (LS-BDA) [LSR16] and centralized extended Kalman filter (LS-Cen) [KRM14] have similar performances but LS-Cen performs slightly better when there is a lack of observation data using CoLo Dataset 4. Users can use other datasets with different settings to compare these two algorithms more thoroughly.

Table 6.1: Algorithm comparison performance metrics

Settings		
Dataset		CoLo Dataset 4
Duration (s)		240
Robots		[1, 2, 3]
Results	**LS-BDA**	**LS-Cen**
Robot 1 location errors (m)	0.1034	0.1117
Robot 1 $trace(\Sigma_s)$ (m^2):	0.0061	0.0090
Robot 2 location errors (m):	0.0946	0.1080
Robot 2 $trace(\Sigma_s)$ (m^2):	0.0098	0.0067
Robot 3 location errors (m):	0.1614	0.1922
Robot 3 $trace(\Sigma_s)$ (m^2):	0.0075	0.0153
Avg. location errors (m):	0.1092	0.1207
Avg. $trace(\Sigma_s)$ (m^2):	0.0095	0.0144

CHAPTER 7

Conclusion

In this thesis, I presented a performance evaluation system CoLo for two-dimensional cooperative localization algorithms from physical experiment (CoLo-PE) to software analysis tool (CoLo-AT). Researchers can easily test their cooperative localization algorithms using different real-world datasets with various settings on CoLo to evaluate the performance of their algorithms more thoroughly instead of creating their own simulation environments. We introduced CoLo's physical experiments setup (CoLo-PE) for real-world data collection with a scalable team of low-cost and configurable robots, as well as CoLo's software analysis tool (CoLo-AT) which effectively tests the performance of cooperative localization algorithms using the datasets from CoLo-PE or other compatible datasets. With the creation of CoLo, it will be much more convenient for researchers to evaluate their cooperative localization algorithms, which can reduce the algorithm development cycle time. As better performing cooperative localization algorithms emerge, these algorithms can provide more accurate data on robots' poses to support the development of multi-robot systems.

REFERENCES

[BNG06] T. Bailey, J. Nieto, J. Guivant, M. Stevens, and E. Nebot. "Consistency of the EKF-SLAM Algorithm." In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3562–3568, October 2006.

[CCC16] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age." *IEEE Transactions on Robotics*, **32**(6):1309–1332, Dec 2016.

[CCM17] Tsang-Kai Chang, Shengkang Chen, and Ankur Mehta. "Multirobot cooperative localization algorithm with explicit communication and its topology analysis." In *2017 International Symposium on Robotics Research*, 2017.

[CNG13] L. C. Carrillo-Arce, E. D. Nerurkar, J. L. Gordillo, and S. I. Roumeliotis. "Decentralized multi-robot cooperative localization using covariance intersection." In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1412–1417, Nov 2013.

[GMM14] S. Garrido-Jurado, R. Muñoz-Salinas, F.J. Madrid-Cuevas, and M.J. Marín-Jiménez. "Automatic generation and detection of highly reliable fiducial markers under occlusion." *Pattern Recognition*, **47**(6):2280 – 2292, 2014.

[HDJ10] Haoyao Chen, Dong Sun, Jie Yang, and Jian Chen. "Localization for Multirobot Formations in Indoor Environment." *IEEE/ASME Transactions on Mechatronics*, **15**(4):561–574, aug 2010.

[JDO13] Adrián Jiménez-González, Jose Ramiro Martinez de Dios, and Anibal Ollero. "Testbeds for ubiquitous robotics: A survey." *Robotics and Autonomous Systems*, **61**(12):1487 – 1501, 2013.

[KCA06] N. Karam, F. Chausse, R. Aufrere, and R. Chapuis. "Cooperative Multi-Vehicle Localization." In *2006 IEEE Intelligent Vehicles Symposium*, pp. 564–570, June 2006.

[KRM14] S. S. Kia, S. F. Rounds, and S. Mart nez. "A centralized-equivalent decentralized implementation of Extended Kalman Filters for cooperative localization." In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3761–3766, Sept 2014.

[LHB11] Keith YK Leung, Yoni Halpern, Timothy D Barfoot, and Hugh HT Liu. "The UTIAS multi-robot cooperative localization and mapping dataset." *The International Journal of Robotics Research*, **30**(8):969–974, 2011.

[LLS16] C. Li, J. Lu, and W. Su. "EKF based distributed cooperative localization for a multirobot team." In *2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pp. 1–6, Nov 2016.

[LN12] H. Li and F. Nashashibi. "Cooperative multi-vehicle localization using split covariance intersection filter." In *2012 IEEE Intelligent Vehicles Symposium*, pp. 211–216, June 2012.

[LSR16] Lukas Luft, Tobias Schubert, Stergios I. Roumeliotis, and Wolfram Burgard. "Recursive Decentralized Collaborative Localization for Sparsely Communicating Robots." In *Proceedings of Robotics: Science and Systems*, AnnArbor, Michigan, June 2016.

[MFK08] N. Michael, J. Fink, and V. Kumar. "Experimental Testbed for Large Multirobot Teams." *IEEE Robotics Automation Magazine*, **15**(1):53–61, March 2008.

[MR06] A.I. Mourikis and S.I. Roumeliotis. "Performance analysis of multirobot Cooperative localization." *IEEE Transactions on Robotics*, **22**(4):666–681, aug 2006.

[PC09] Andrzej Pronobis and Barbara Caputo. "COLD: COsy Localization Database." *International Journal of Robotics Research (IJRR)*, **28**(5):588–594, May 2009.

[Per16] Jacob M. Perron. "create_autonomy.", 2016.

[PGW17] D. Pickem, P. Glotfelter, L. Wang, M. Mote, A. Ames, E. Feron, and M. Egerstedt. "The Robotarium: A remotely accessible swarm robotics research testbed." In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1699–1706, May 2017.

[PT14] Benjamin Pitzer and Russell Toris. "usb_cam.", 2014.

[QCG09] Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. "ROS: an open-source Robot Operating System." In *ICRA Workshop on Open Source Software*, 2009.

[RB02] S.I. Roumeliotis and G.A. Bekey. "Distributed multirobot localization." *IEEE Transactions on Robotics and Automation*, **18**(5):781–795, oct 2002.

[RGU13] M. Di Rocco, F. La Gala, and G. Ulivi. "Testing Multirobot Algorithms: SAETTA: A Small and Cheap Mobile Unit." *IEEE Robotics Automation Magazine*, **20**(2):52–62, June 2013.

[STS16] Sajad Saeedi, Michael Trentini, Mae Seto, and Howard Li. "Multiple-Robot Simultaneous Localization and Mapping: A Review." *Journal of Field Robotics*, **33**(1):3–46, 2016.

[SVF06] A. Stubbs, V. Vladimerou, A. T. Fulford, D. King, J. Strick, and G. E. Dullerud. "Multivehicle systems control over networks: a hovercraft testbed for networked and decentralized control." *IEEE Control Systems Magazine*, **26**(3):56–69, June 2006.

[Vau18] Jim Vaughan. "aruco_detect.", 2018.

[WB06] Greg Welch and Gary Bishop. "An Introduction to the Kalman Filter." Technical report, University of North Carolina at Chapel Hill, 2006.

[WMG14] T. R. Wanasinghe, G. K. I. Mann, and R. G. Gosine. "Decentralized Cooperative Localization for Heterogeneous Multi-robot System Using Split Covariance Intersection Filter." In *2014 Canadian Conference on Computer and Robot Vision*, pp. 167–174, May 2014.