Simultaneous Localization and Mapping via Multi-Robot Task Allocation (MRTA)

Christopher Covert
cwcovert@stanford.edu
Department of Aeronautics and Astronautics
Stanford University

Abstract

This paper presents a simple Multi-SLAM algorithm that expands upon existing work in the field of Multi-Robot Task Allocation (MRTA). The primary purpose of MRTA is dedicated to the various methods of assigning a network of mobile sensing robots to a set of tasks in order to complete them as quickly and efficiently as possible. This paper considers the problem of added uncertainty in both robot and task state information, adding the complexity of the robot-to-task decision-making process. In this work, sequential particle filter localization and extended Kalman filter mapping algorithms are implemented in order to estimate both personal and task states within each iteration. To verify hypotheses posed in this work, both algorithmic development and simulated results are shown for a range of uncertainty parameters to map sensitivity to measurement and process noise. As a result, the algorithm presented in this work is shown to not only be both feasible within the realm of realistic range-based sensors, but the model is able to complete the same number of tasks in the same amount of time with minor drift in the final estimated task state.

Keywords: multi-robot task allocation, Voronoi coverage, distributed control, Kalman filtering, Particle filtering, state estimation, SLAM

1 Introduction

1.1 MRTA Decision-Making

In the field of Multi-Robot Task Allocation (MRTA), a set of multiple robots in a network are given the objective of completing a set of tasks as efficiently as possible with limited communication between agents. This problem, it can become clear, is the foundation for large-scale robotic automation where a fleet of robots autonomously completes a variety of tasks as quickly as possible. Be it for the scanning, sorting, and shipping of parcels in a warehouse or the planting and managing of crops on a farm, a team of robotic vehicles completing a set of complex tasks is no easy challenge. This work is motivated by the exploration of the Martian surface for pre-colonization, where robots must survey land, gather resources, and construct habitats completely autonomously. Whereas most of the interplanetary vehicles sent to the Martian surface have taken the form of a rover, tasks such as these will require vastly different types of robots with unique abilities that can adequately handle the different tasks at hand.

The goal of MRTA within the scope of this research is therefore to simulate a fleet of decentralized robots on the surface of Mars, globally tasked with achieving some overall objective which requires the completion of heterogeneous tasks. To accomplish this, an algorithm was implemented that emphasized performance in the case of homogeneity and heterogeneity for both robots and tasks. Also, whereas many algorithms presented in the field of MRTA require centralized control, which leads to an optimal but fault-prone solution, this work utilizes a fully-decentralized approach under the assumption that communication on Mars is severely limited. To ensure distributed control, robots in this research are only capable of identifying resources or other robots within some range of themselves using an artificial radial range finding system.

Under the assumption of minimal communication, there are two main calculations that each robot must make independently: (1) their individual utility of pursuing tasks in their vicinity,

and (2) the utility of nearby robots completing nearby tasks. This information allows the robots to decide which task to ultimately pursue. To minimize distance traveled per robot, the problem frames the utility of pursuing a task as a function of the robot's proximity to the task, the attributes of the robots, the requirements of the task, and the utilities of the other agents with respect to the same task to be performed. In a fully homogeneous environment, the utility no longer becomes a function of the attributes of the robot nor the task to be performed since all feature sets are equal, and therefore is trivially based upon the proximity to the nearest objective.

1.2 MRTA Multi-SLAM

Currently, robots and tasks are initialized and decisions are made in a decentralized manner by means of a radial 'optical sensor' that delineates the radius of neighboring nodes. Throughout the entire process, robot states and task locations are well-defined and known to all vehicles within the radial distance with zero error. After tasks are completed, new tasks are introduced in the environment with known position to robots as well. While these assumptions make the decision-making process more precise, it remains detached from realistic sensor models and areas of uncertainty that would be found in deployment.

Therefore, it is necessary to introduce noise in the multi-robot system by relaxing the assumptions made in localization and environmental awareness. By injecting uncertainty into relative pose between robots and tasks, the MRTA problem will be scaled in complexity, but provide a more valuable insight into the worth/efficiency of the algorithm that a noise-free simulation suggests.

2 Related Work

2.1 MRTA Decision-Making

The field of MRTA has been a field of active research for over a decade. Many formulations of the problem have arisen and many solution methods have been examined. This section will examine a few of the approaches that helped define the work's problem statement.

As an overview of the field of research as a whole, [3] is a helpful introduction to the area of multi-robot task allocation (MRTA) problems. This work provides a taxonomy for MRTA problems and some applicable solution algorithms (with some analyses on performance), outlining problem variants where robots may be capable of performing single or multiple tasks, where tasks may require single or multiple robots, and where future task information may or may not be available. These problems are presented in the context of combinatorial optimization problems where the expected performance or utility is the quantity to be optimized. [13] also provides a survey of solutions relating to probabilistic MRTA environments within the scope of distributed task allocation that frame the problem as a (fully- or partially-observable) Markov Decision Process (MDP).

One method of coordinating distributed tasks in situations where agents are unable to communicate is discussed in [6]. Here, the authors show that employing a probabilistic threshold response across agents (i.e. an agent will only participate in a task if its measurement of a stimulus exceeds some threshold) in a multi-robot systems results in a Bayesian Nash equilibrium. It should be noted that it does not seem as though the paper makes any indication of optimality (or sub-optimality) of this method of task allocation. However, it is a simple idea that could serve as a reasonable starting point for a system facing conditions in which communication may be limited, such as for the case of robots in an uncharted area on the Martian surface. MRTA problems have also implemented the Response Threshold Method (RTM) in [4] by adapting a probabilistic approach which utilizes fuzzy Markov chains for multi-robot swarms which subdivide into groups of collocated agents to complete a task. This use of fuzzy sets in this work is supported with evidence to the idea that noise-based uncertainty converges faster than traditional MDPs under certain conditions. [1] describes these conditions and provides the mathematical framework of fuzzy sets and their correlation to standard Markov decision processes. It also provides evidence to the claim that the application of fuzzy Markov chains have a finite convergence to a stationary solution with robustness to small perturbations of the transition matrix when the chain is ergodic, in support of the results of [4]. Additionally, [7] describes a probabilistic approach to decision-making under uncertainty via a centralized Monte Carlo Tree Search based algorithm which utilizes the branch

and bound method to solve the MRTA problem in a constrained environment, comparing task allocation efficiency to other distributed approaches.

As opposed to the [6] and [4], wherein agents simultaneously select an action (whether or not to participate in a task) as a one-shot decision process, [8] proposes a method of dynamically allocating tasks. In this case, the authors propose that agents observe their local environment with some frequency, keeping count of the number of available tasks of each type and the number of agents performing each task type. A given agent could then probabilistically determine whether or not to switch tasks and what task to switch to depending on these counts. The aim of this paper was not to determine the optimality of such a task allocation method, but rather to provide a mathematical model for this task allocation method leaving some of the specific details, such as observation rate and the exact task transition function, left open for tuning.

For systems with uncertain travel times and task duration, [12] describes an analytical framework for modeling resource contention which is used to accurately calculate the probability distributions for task durations, finding that it is agnostic to the specific objective function used. This framework, when applied to environments of complete uncertainty, rely heavily on resource-based health monitoring systems of the robots. For heterogeneous robot systems, [2] also abstracts MRTA to a set of design parameters which rely on individual robot traits rather than environment-based traits, including robot availability, individual skill sets, distance to task location, and remaining energy of the robots. The paper also analyses network performance in terms of throughput, task allocation delay, execution time, and residual energy as key metrics.

In swarm-based robotics, the challenge of partitioning groups of homogeneous robots into subgroups for MRTA was also studied in the application of distributed algorithms that compared stochastic decision-making to probabilistic methods in [10]. Similar approaches were adapted by [9] and [11] in the application of both grid world and real world task allocation experiments covering the emergency handling problem domain. The value in these papers lies in the coverage of the MRTA problem from introduction to application in both simulated and hardware-based results.

2.2 Multi-SLAM Applications

While the field of Multi-SLAM is already well-populated with literature, there exist a variety of applications that take advantage of the various different features of available filters.

2.2.1 Extended Kalman Filters

For instance, the Extended Kalman filter (EKF) is widely popular for state estimation of nonlinear systems. [17] outlines an algorithm where each robot detects landmarks and other robots, and estimates their positions by the EKF. To achieve good estimation accuracy, an optimal information fusion technique is adapted to the multi-robot SLAM problem. For feature-based environment learning, [18] uses a unified EKF-based SLAM framework for each robot which eventually builds a line feature based partial 3D model of the environment. Each moving robotic sensor node then shares its feature based map model to other moving nodes which are in communication range. [22] presents the multi-robot visual SLAM system based on the EKF by modifying the MonoSLAM system to allow for cooperation of several heterogeneous mobile robots. [20] analyzes three main methods of multi-SLAM: SLAM-based EKF, SLAM-based Particle Filter, and SLAM-based Graph Optimization (EKF-SLAM, PFSLAM and Graph-based SLAM). It also provides a comprehensive literature review on the key scientific and technical difficulties in the lunar context that Visual SLAM faces.

2.2.2 Particle Filters

An excellent source on Particle Filters for multi-robot systems, [21] describes an on-line algorithm for multi-robot SLAM by taking the starting point the single-robot Rao-Blackwellized particle filter and making three key generalizations. First, they extend the particle filter to handle multi-robot SLAM problems in which the initial pose of the robots is known (such as occurs when all robots start from the same location). Second, they introduce an approximation to solve the more general problem in which the initial pose of robots is not known a-priori (such as occurs when the robots start from widely separated locations). In this latter case, they assume that pairs of robots will eventually encounter one another, thereby determining their relative pose. They use this relative pose to initialize the filter, and combine the subsequent observations from both robots into a

common map. Third and finally, they introduce a method for integrating observations collected prior to the first robot encounter, using the notion of a virtual robot traveling backwards in time. This novel approach allows them to integrate all data from all robots into a single common map. [23] also successfully implements both single-robot SLAM and multi-robot SLAM using particle filters. By comparison between these two approaches, they show that the results from multi-robot SLAM outperform the single-robot SLAM in terms of output quality.

2.2.3 Joint Maps

Another key feature of Multi-SLAM is the ability to map unknown environments. To this end, [6] presents an algorithm for the multi-robot SLAM problem with the robot initial locations completely unknown, where each robot builds its own local map using the traditional EKF-SLAM algorithm. They provide a new method to fuse the local maps into a jointly maintained global map by first transforming the local map state estimate into relative location information and then conducting the fusion using the decoupled SLAM (D-SLAM) framework. [14] also presents an algorithm that enables teams of robots to build joint maps, even if their relative starting locations are unknown and landmarks are ambiguous. It achieves this capability through a sparse information filter technique, which represents maps and robot poses by Gaussian Markov random fields. The alignment of local maps into a single global maps is achieved by a tree-based algorithm for searching similar-looking local landmark configurations, paired with a hill climbing algorithm that maximizes the overall likelihood by search in the space of correspondences. On the trend of creating joint maps, [15] presents a new approach to the multi-robot map-alignment problem that enables teams of robots to build joint maps without initial knowledge of their relative poses. The key contribution of this work is an optimal algorithm for merging (not necessarily overlapping) maps that are created by different robots independently.

2.2.4 Multi-Robot Belief Propagation

Establishing collaborative behavior is an important factor in coordinating teams of robots. Multi-robot task allocation is one aspect of group coordination that deals with the assignment of robots to subtasks. Toward this end, [16] proposes multi-robot belief propagation (MRBP), a synthesis of distributed probabilistic inference and notions of theory-of-mind for the purpose of multi-robot task allocation. MRBP does not rely upon a central planner or fixed decision hierarchy, but allows for the relaying of task assignments locally through Bayesian belief propagation.

A common theme among the referenced works is a mechanism for estimating state under uncertainty, both internally to the robot and externally to its neighbors and neighboring environment. Analysis of these methods is often framed as a SLAM problem, where uncertainty is compounded and filters are used to reduce errors in localization while simultaneously mapping the environment. For the scope of this report, an implementation of PFSLAM methods and rudimentary joint-maps will be applied to preliminary research conducted in MRTA on greedy, feature-mapping task allocation.

3 Methodology

3.1 MRTA Decision-Making

As a reminder of how the control law and decision-making processes are determined for this research, a brief discourse is provided in this section to serve as a highlight of the key features of the foundational dynamics.

To implement a control law which utilizes these heterogeneous feature sets to achieve task allocation, a utility function is used to determine compatibility between a robot and the nearby tasks. In this formulation, the utility of a robot completing a task is directly related to the robot's compatibility with the task and inversely related to its distance from the task. In this application, the utility of robot i completing task j is defined as:

$$U_{ij} = \frac{(\boldsymbol{\theta}_i \cdot \boldsymbol{\psi}_j)^\alpha}{(d_{ij})^\beta} - \max_k \frac{(\boldsymbol{\theta}_k \cdot \boldsymbol{\psi}_j)^\alpha}{(d_{kj})^\beta} \tag{1}$$

where i is the index of a robot, j is the index of a task, $k \in \mathcal{N}_i$, and α and β are power-law scaling parameters used to exponentially weight compatibility and proximity, respectively. It can be easily recognized that by tuning α and β, the utility function can place more weight on compatibility ($\alpha > \beta$) or proximity ($\alpha < \beta$), which can be shown to have significant and predictable changes to the network's behavior with respect to heterogeneity.

By taking the difference of the utility between an individual robot-task pair and the maximum value of utility within the neighbor set of robot i to the same task, a robot can determine whether or not it is the best agent to complete the task at hand within the whole of its network. This not only allows the system to perform a simple decision making process, but also decentralizes the algorithm by limiting the required communication between nodes to those within a specified communication range.

In regards to the dynamics, a robot first determines the optimal task by finding the task j that maximizes U_{ij}. It is then commanded to drive directly to the task under the following control law:

$$\dot{x}_i = \{(x_j - x_i) \quad | \quad x_j = \underset{j}{\text{argmax}} \quad U_{ij}\} \tag{2}$$

This control law holds only if $\max_j U_{ij} > 0$. Intuitively, this means that the robot will only pursue a task if it is the most compatible robot for the task. If $\max_j U_{ij} < 0$, the robot is not the most compatible robot for any task within its scope. In this event that the robot does not have an optimal task to approach, the robot uses Lloyd's algorithm to move toward the centroid of its Voronoi cell so as to maintain coverage throughout the environment and prepare for any new tasks that may appear. These centroidal Voronoi dynamics are given as such:

$$\dot{x} = k(Cv_i - x_i) \tag{3}$$

where Cv_i is the centroid of the robot's Voronoi cell and k is a scalar gain. As stated in the justification for the algorithm, this use of Voronoi coverage will naturally induce a coverage controller that will maintain spacing and minimize the sensing cost over the convex set as new tasks stochastically repopulate the environment.

In the homogeneous case, where all robots and tasks are identical, $\boldsymbol{\theta}_i \cdot \boldsymbol{\psi}_j = \text{constant}\ \forall i, j$ in the network since all robots and all tasks will have identical feature sets. This furthermore implies that the assignment of a task for each node is only a function of proximity, as the only variable in the utility function becomes d_{ij}. Therefore, the homogeneous case can be thought of as driving the robots toward the nearest task in their Voronoi cell.

Lemma 3.1 *In the homogeneous case, robots navigate toward the nearest task in their Voronoi cell.*

Proof. The Voronoi cell of a robot is defined as the subspace of points in the environment that are closer to robot i than any other robot in the environment. Mathematically, the Voronoi cell can be described as:

$$V_i = \{x \in X \mid d(x, x_i) \leq d(x, x_k),\ \forall k \neq i\}, \tag{4}$$

Since $\boldsymbol{\theta}_i \cdot \boldsymbol{\psi}_j = c\ \ \forall i, j$ in presence of pure homogeneity,

$$U_{ij} = \frac{c}{(d_{ij})^\beta} - \max_k \frac{c}{(d_{kj})^\beta} \tag{5}$$

Therefore, if a task exists within the Voronoi cell of robot i, then $d_{ij} \leq d_{kj}$ and $U_{ij} \geq 0$. From here it is intuitive that the task in the Voronoi cell of robot i that maximizes the utility will be the one that minimizes d_{ij}.

Although this simplification breaks down in the heterogeneous case, the control law remains the same. Future work may consider a way to redistribute unassigned robots in a way that takes account of their individual attributes, however in this formulation, unassigned robots will always move toward the centroid of their Voronoi cell.

Algorithm 1: Robot Dynamics

```
while tasks left to be completed do
    calculate utilities;
    if max_j U_ij > 0 then
        move towards task;
        if arrived at task then
            task completed + = 1;
        else
            continue moving towards task;
        end
    else
        move towards centroid of Voronoi cell;
    end
end
```

3.2 MRTA Multi-SLAM

Although the proposed MRTA algorithm may be a simple means of solving the task allocation problem, it runs on the assumption that each robot has complete certainty of its own location, its exact proximity to a set of tasks in the environment. While the distance between each robot is not shared, the utility function is directly correlated to range-based measurements of each agent to a task at hand, which is shared using a distributed network amongst robots within a preallocated range from each other. Obviously, by adding uncertainty into the environment, as would be present in hardware demonstrations of such an algorithm, proximity uncertainty would skew the results of the utility function by a scale of how noisy the measurements are as compared to their true state.

Therefore, a Multi-SLAM algorithm that achieves takes advantage of a simplistic joint-map propagation is used to quickly minimize the global uncertainty of the position of each robot. To implement this approach, a particle filter is used to localize the robots in the environment, and an EKF is used to map the locations of the tasks. These two algorithms are run sequentially, where the input into the mapping algorithm is the estimated localized state, and the input to the next iteration of localization takes in the mapped task estimates.

These algorithms take advantage of a multi-robot/multi-task environment by individually estimating all locations of robots and tasks in the frame of each bot and then combine estimates to get a mean-estimate of the state.

For the implementation of the particle filter, each robot's position is estimated as a set of particles populated within the error ellipse of the covariance at each time step. For the purpose of this research, ten particles has been selected as the standard count, and has been implemented for all subsequent simulations. A particle filter with this number of particles was chosen in particular because of it's behavior when approaching mapped feature sets. When implementing other filters that do not propagate more than one state, navigation through a dynamic environment causes drift in state estimation as a result of sampling from a changing set of known locations. Particle filters reduce this drift behavior by allowing for a finer resolution of estimates to be made for each state, allowing less cumulated error when the map changes. For obvious reasons, as a task is completed in the MRTA problem, a new task is stochastically generated and the completed task falls off of the radar of the agents.

These rapid, stochastic environment dynamics require a filter that will provide the best estimate as tasks relocate, and therefore the following algorithm has been generated:

Algorithm 2: Multi-SLAM Particle Filter Localization

INPUT: μ_{rob}, Σ_{rob}, μ_{task}
INIT: n = number of robots
m = number of tasks
M = number of particles
for *1:m* **do**
  for *1:n* **do**
    Measure range of each robot to each task;
  end
end
for *1:M* **do**
  for *1:n* **do**
    Add covariance of each measurement to state for each particle;
    Propagate each new particle with noisy state dynamics;
    for *1:m* **do**
      Measure new range of each particle to each task;
    end
    Calculate particle weight vectors for each robot;
  end
end
Normalize the weight vectors for each robot;
for *1:n* **do**
  Calculate weighted positions for particle for each robot;
end
$\bar{\mu}$ = average robot particle position;
for *1:M* **do**
  for *1:n* **do**
    Calculate weighted covariance for each particle for each robot;
  end
end
$\bar{\Sigma}$ = average robot covariance;
OUTPUT: $\bar{\mu}$, $\bar{\Sigma}$

As for task mapping, the algorithm takes in the estimates robot state as output from the particle filter localization function and uses the estimated state of each robot to determine the location of the feature using the following EKF mapping algorithm:

Algorithm 3: Multi-SLAM EKF Mapping

INPUT: μ_{task}, Σ_{task}, μ_{rob}
INIT: n = number of robots
m = number of tasks
for *1:m* **do**
  for *1:n* **do**
    Measure range of each robot to each task;
  end
end
for *1:m* **do**
  for *1:n* **do**
    Calculate range-based measurement Jacobian;
  end
end
Predict μ_{task} and Σ_{task} for each task;
for *1:n* **do**
  Calculate Kalman gain for each task;
end
for *1:m* **do**
  for *1:n* **do**
    Measure range of each robot to each predicted task;
  end
end
for *1:n* **do**
  Calculate estimated μ_{task} and Σ_{task} for each robot;
end
$\bar{\mu}$ = average task estimate from each robot;
$\bar{\Sigma}$ = average task covariance from each robot;
OUTPUT: $\bar{\mu}$, $\bar{\Sigma}$

4 Simulations and Results

4.1 Simulation Methodology

In order to qualitatively demonstrate the success of the algorithm, simulations modeling the robot dynamics and task dynamics were implemented using MATLAB. Each simulation begins by generating a predefined number of robots and initializing them to random initial locations. Furthermore, each robot is randomly assigned an attribute vector from a set of predefined robot types. Similarly, tasks are also stochastically generated in the environment and initialized with a task requirements vector from a set of matching task types. Once the initialization of robots and tasks is complete, at each time step, the robot dynamics follow Algorithm 1. In the simulation, a task is considered to be completed if a robot comes within some small ϵ distance of said task, and once a task is completed, it is removed from the environment and a new task of random type is generated at another random location. A block diagram representing the simulation logic can be seen in Figure 1. The simulation then ends when the predetermined timespan is reached.

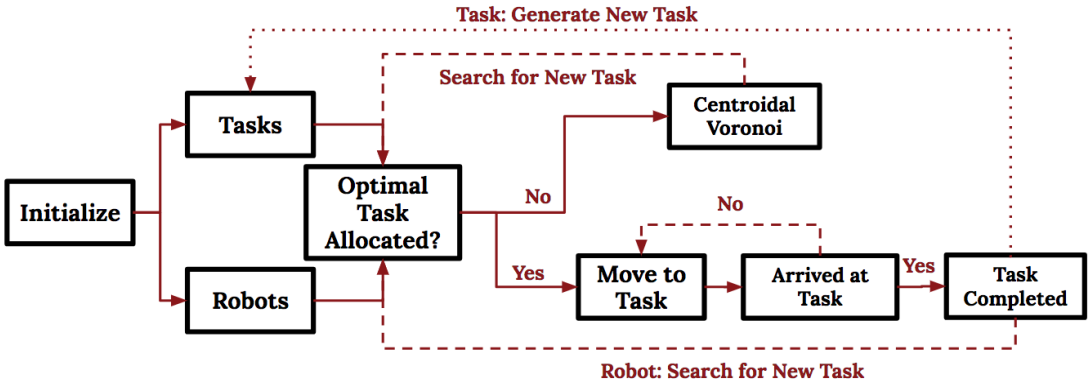

Figure 1: Simulation logic block diagram.

4.2 Simulated MRTA Multi-SLAM

For the purpose of testing the robustness of the multi-SLAM filters, several additional simulations are presented with variations of noise and estimate covariance. Also, as the feature sets of the robot and task are still known with absolute certainty, the addition of state uncertainty will provide the same effects to both homogeneous and heterogeneous datasets. Therefore, for the ease of visualization, only homogeneous simulations have been shown.

All simulations also carry an initial identity estimate covariance, while the parameters for Q and R are tweaked to represent the variety of scalability of sensor models (high variance will most likely come from inexpensive range-detection systems and lowing the variance of the measurement and process noise will increase the cost of the required hardware). All simulations also follow the same series of time-histories has been plotted for decision-making visualization to represent initial, mid-duration, and final trajectories of the simulation.

4.2.1 Low Covariance Case

To begin the simulation series of covariance-based runs, process and measurement noise for both localization and mapping was set to be equally low with Q and R being 0.001. This represents near certainty in both where each robot is located and where each task is located in relation to it. Therefore, it should be expected that this set of data represents the closest to the noise-free simulation. These parameters are are also scaled to a 10x10 convex environment, so if the grid is to be represented in meters, the standard deviation will be on the order of the centimeter level.

The following figure represents this low covariance run of five robots and five tasks running the Multi-SLAM algorithm. The black lines represent true robot trajectories, while the blue dots represent state estimates. Additionally plotted within each graph is now a plot of task location, represented by red dots to show drift over time.

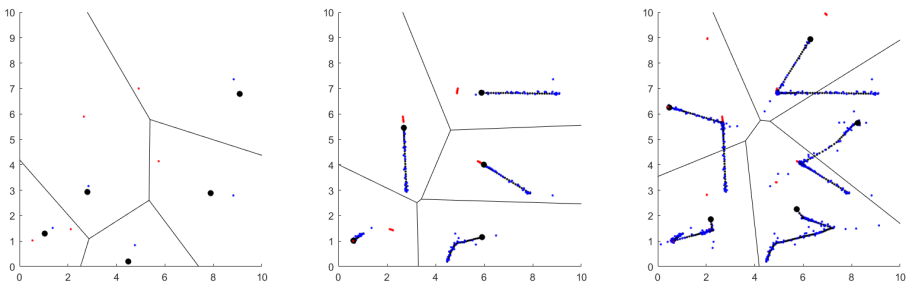

Figure 2: Low Covariance Case Time History with Q = 0.001, R = 0.001.

Of note in this simulation is that the initial estimate of the robot position carries approximately one meter of error, but converges to the true trajectory within a few timesteps. This is because the system is using five estimates to converge on position instead of just one. In the final trajectory plot,

it is possible to see state estimates that diverge from the true trajectory, which is also expected. This effect is analyzed further in the following section.

4.2.2 Mid-Range Covariance Case

By increasing the values of Q and R to 0.1, the standard deviation is on the decimeter level, which has a much stronger influence on the trajectory of the estimates.

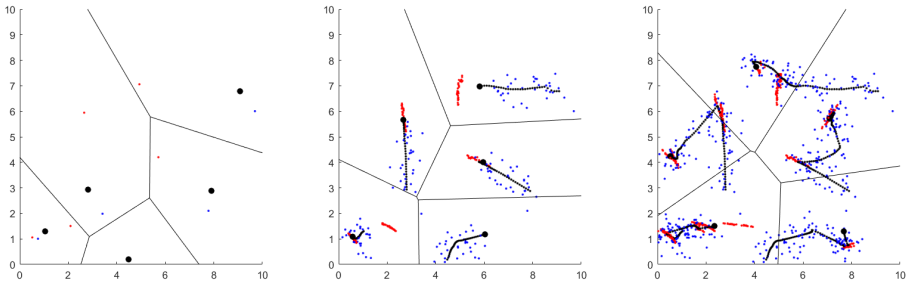

Figure 3: Mid-Range Covariance Case Time History with Q = 0.1, R = 0.1.

While the mid-range covariance results still achieve their goal of tracking the tasks and successfully navigating toward them, the noise present in the robot's state estimate causes the estimated task locations to drift over time. While the amount of drift in this approximately one meter in magnitude from end to end in the worst case scenario, high-precision tasks (such as intelligence, surveillance, and reconnaissance missions that require high quality imagery) will be completely useless if they start to deviate from the target.

4.2.3 High Covariance Case

In the worst case scenario, meter-level standard deviations in a 10x10 meter grid have been simulated as well, mainly to test the limit of the simulation and its robustness to complete uncertainty.

With such a large amount of uncertainty present in the system, the robots only have a general idea of where they are, and make an incredibly poor estimate of the target's location. As they approach that target, their collective new estimates drift significantly from their original estimates, and the robots play a follow-the-leader-type mission, never converging on their own state nor the state of the task.

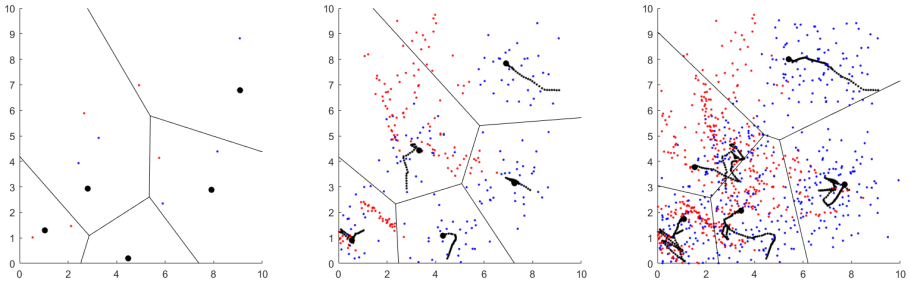

Figure 4: High Covariance Case Time History with Q = 1, R = 1.

4.2.4 Realistic Covariance Case

With those parameters defined, a more realistic simulation was run with process noise covariance of a magnitude of 0.01 and measurement noise covariance of a magnitude of 1. This represents low process noise (to smooth the estimates) and high sensor noise (to represent a feasible range sensor).

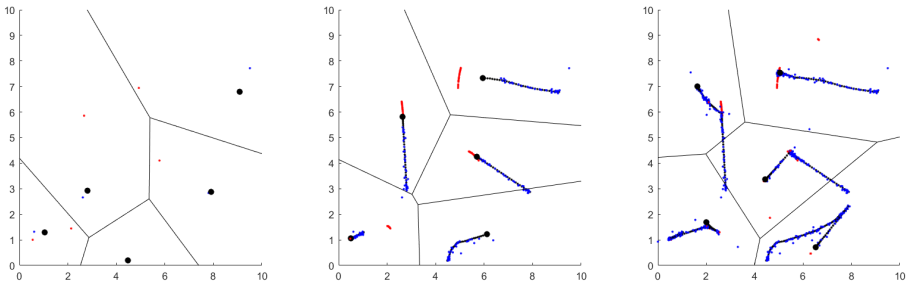

Figure 5: High Covariance Case Time History with Q = 0.01, R = 1.

As it can be discerned from the plot, even with high sensor noise, the system does a magnificent job of tracking robot positions with only minor drifts in sensor noise. Despite low robot state estimation error, the estimate still diverges with the completion of tasks.

5 Discussion

To understand why the divergence of the state occurs as the tasks are completed, it is easy to visualize a time history of the estimation error over time overlaid with the time of task completion.

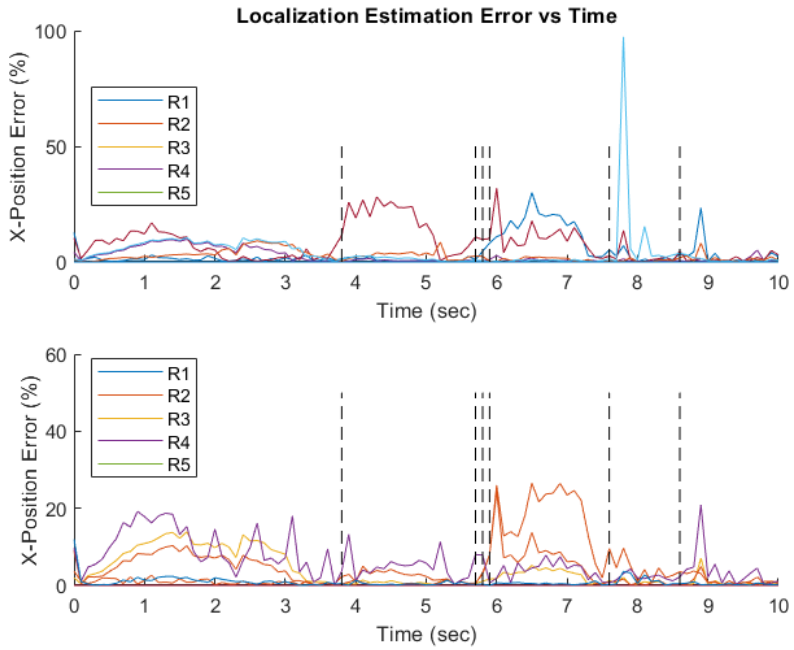

Figure 6: Plot of Robot Localization Error vs Time

As seen in the plot above, it is clear that the completion of a task is correlated to an increase in localization estimation error. This is a direct result of a task being completed, removed, and regenerated within a single time step. The temporary divergence of the estimate causes the known map of the tasks to have extreme differences between predicted measurement to the completed task's original location and the task's estimated new location. Even with these divergences, however, the trajectory is able to realign with the true state. Since the Multi-SLAM algorithm couples the robotic estimates with each other estimate in the network, this compounds the error of the estimation across all robots, but primarily affects the robot that completed the task, as the difference between predicted task location and new task location is the highest. Therefore, the plot above not only suggests correlation, but causation in return.

6 Conclusion

The objective of this research was to implement a Multi-SLAM filter into the MRTA problem to eliminate certainty from the simulation of the decision-making process and test the simplistic utility-based algorithm and its robustness to the sensor models and state estimation of a potential hardware demonstration. As seen in the results, although it is not completely robust to high-variance process noise, the simulation works as expected with as few as five robots under realistic sensor measurements. The ultimate trade-off reflected in these results is the selection of sensor noise that permits an acceptable amount of task drift for the mission profile at hand.

7 Future Work

There are numerous avenues upon which the work in the report could be expanded. To begin, the implementation of the range based sensor model only takes into consideration proximity of each task to each target. When a target is moved, the state of the collocated task diverges with non-trivial error. One particular solution to this would be to re-normalize the estimates with a inter-robot range-based measurement, even limited to its Delaunay neighbors. This would allow the divergent state estimates to be re-centered on the nominal trajectory without taking into consideration the locations of the tasks at all.

In addition to adjusting the localization algorithm, an additional Multi-Hypothesis Kalman Filter could be introduced to determine a noisy estimate of the feature sets of the tasks as they are generated. This would obviously scale the simulation in complexity, and would be initially based upon an assumed estimate of the feature set that would need to be derived from a form of object recognition algorithm such as that from a neural network.

Also, in order to further prepare for prototype-readiness, this algorithm can also be expanded by applying unicycle/tricycle models into the dynamics of the models. From these realistic robot dynamics, valuable information may be gathered about the practicality of centroidal coverage in the presence of nonholonomic constraints.

References

[1] Avrachenkov, K. and Sanchez, E. (2002). Fuzzy Markov Chains and Decision-Making. *Fuzzy Optimization and Decision Making*, 1(2), pp.143-159.

[2] Chowdhury, M. and Maier, M. (2017). Local and Nonlocal Human-to-Robot Task Allocation in Fiber-Wireless Multi-Robot Networks. In *IEEE Systems Journal*, (PP)99, pp.1-11.

[3] Gerkey, B. and Matarić, M. (2004). A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. *The International Journal of Robotics Research*, 23(9), pp.939-954.

[4] Guerrero, J., Velero, O. and Oliver, G. (2017). Toward a Possibilistic Swarm Multi-robot Task Allocation: Theoretical and Experimental Results. *Neural Processing Letters*, 46(3), pp.881-897.

[5] Hanna, H. (2005). Decentralized approach for multi-robot task allocation problem with uncertain task execution. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 535-540.

[6] Kanakia, A., Touri, B. & Correll, N. Swarm Intell (2016). Modeling multi-robot task allocation with limited information as global game. *Swarm Intelligence*, 10(2), pp.147-160.

[7] Kartal, B., Nunes, E., Godoy, J. and Gini M. (2016). Monte Carlo Tree Search for Multi-Robot Task Allocation. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pp.4222-4223.

[8] Lerman, K., Jones, C., Galstyan, A. and Matarić, M. (2006). Analysis of Dynamic Task Allocation in Multi-Robot Systems. *The International Journal of Robotics Research*, 25(3), pp.225-241.

[9] Mataric, M., Sukhatme, G., and Ostergaard, E. (2003). Multi-Robot Task Allocation in Uncertain Environments. *Autonomous Robots*, 14(2-3), pp.255-263.

[10] McLurkin, J. and Yamins, D. (2005). McLurkin, J., & Yamins, D. (2005). Dynamic Task Assignment in Robot Swarms. *Robotics: Science and Systems.*

[11] Ostergaard, E., Mataric, M. and Sukhatme, G. (2002). Multi-robot task allocation in the light of uncertainty. In *Proceedings 2002 IEEE International Conference on Robotics and Automation*, pp.3002-3007.

[12] Palmer, A., Hill, A. and Schedling, S. (2017). Modelling resource contention in multi-robot task allocation problems with uncertain timing.

[13] Riccio, F., Lazaro, M. T., Gemignani, G. and Nardi, D. (2015). Multi-Robot Perception and Action: World Modeling and Task Allocation. In *RSS Workshop on Principle of multi-robot systems.*

[14] S. Thrun, and Y. Liu, "Multi-robot SLAM with Sparse Extended Information Filters," In: Dario P., Chatila R. (eds) Robotics Research. The Eleventh International Symposium, 2005, Springer Tracts in Advanced Robotics, vol 15. Springer, Berlin, Heidelberg

[15] X. S. Zhou and S. I. Roumeliotis, "Multi-robot SLAM with Unknown Initial Correspondence: The Robot Rendezvous Case," 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, 2006, pp. 1785-1792. doi: 10.1109/IROS.2006.282219

[16] J. N. Schwertfeger and O. C. Jenkins, "Multi-robot belief propagation for distributed robot allocation," 2007 IEEE 6th International Conference on Development and Learning, London, 2007, pp. 193-198. doi: 10.1109/DEVLRN.2007.4354060

[17] T. Sasaoka, I. Kimoto, Y. Kishimoto, K. Takaba and H. Nakashima, "Multi-robot SLAM via Information Fusion Extended Kalman Filters," IFAC-PapersOnLine, Volume 49, Issue 22, 2016, pp 303-308, ISSN 2405-8963, https://doi.org/10.1016/j.ifacol.2016.10.414.

[18] S. R. u. N. Jafri and R. Chellali, "A distributed multi robot SLAM system for environment learning," 2013 IEEE Workshop on Robotic Intelligence in Informationally Structured Space (RiiSS), Singapore, 2013, pp. 82-88. doi: 10.1109/RiiSS.2013.6607933

[19] Z. Wang, S. Huang and G. Dissanayake, "Multi-robot simultaneous localization and mapping using D-SLAM framework," 2007 3rd International Conference on Intelligent Sensors, Sensor Networks and Information, Melbourne, Qld., 2007, pp. 317-322. doi: 10.1109/ISSNIP.2007.4496863

[20] Y. Wang, W. Zhang and P. An, "A survey of simultaneous localization and mapping on unstructured lunar complex environment," 2017 AIP Conference Proceedings, American Institute of Physics, 2017. doi: 10.1063/1.5005198

[21] A. Howard, "Multi-robot Simultaneous Localization and Mapping using Particle Filters," I. J. Robotic Res., 2006, 25. 1243-1256. 10.1177/0278364906072250.

[22] A. Schmidt, "Multi-robot, EKF-Based Visual SLAM System," In: Chmielewski L.J., Kozera R., Shin BS., Wojciechowski K. (eds) Computer Vision and Graphics. ICCVG 2014. Lecture Notes in Computer Science, vol 8671. Springer, Cham. doi: 10.1007/978331911331967

[23] K. Ma and Z. Ma, "Multi-Robot Simultaneous Localization and Mapping (Multi-SLAM)."