Collision-free Optimal Trajectory Planning in a Multiple Robot System

Chahat Deep Singh
Robotics Graduate Student
University of Maryland
chahat@umd.edu

Harshvardhan Uppaluru
Robotics Graduate Student
University of Maryland
uhv93@terpmail.umd.edu

***Abstract*—We address the problem of motion planning and cooperative control in a multiple robot system. We study the problem of cooperative planning in order to simulate the algorithm for concurrent assignment and planning of trajectories (CAPT) for a team of robots. CAPT algorithm addresses two challenges: a. finding a fitting allotment of robots to a given goal locations, and b. generation of collision-free, time parametrized trajectories for every robot. There exists two variations of the CAPT problem. c-CAPT is a complete, centralized algorithm with collision-free optimal solution to a distributed problem in an obstacle-free environment. D-CAPT is a decentralized algorithm that provides suboptimal results in an obstacle-free environment. We extend this to an obstacle-course environment. For this, we propose a complete algorithm: Goal Assignment and Planning (GAP) that computes the optimal trajectories and assignments of robots to goals with complex vehicle dynamics and cost functions. The computational complexity of this algorithm is shown to be cubic, substantially better than the expected exponential complexity associated with planning in the joint state space and assignment of goals to robots. Finally, we illustrate the algorithms and resulting performance through simulation.**

Keywords*—*Multi-robot trajectory planning, Role assignment.

I. Introduction

Aerial robotics is an emerging field with opportunities in civil and navigation applications. We have seen it mature progressively over the past decade. Quadrotors are one of the most commonly used aerial vehicles used in exploration and navigation. Though, each quadrotor is still limited by the field of view, range and payload it can lift. Over the past decade, the range for a single quadrotor has increased from several meters to a few miles and the payload from a few grams to a few kilograms [1][3][4]. However, these robots can carry payloads beyond their capacity of individuals by collaborating in transportation tasks [5]. We have seen such development of the coordination in multiple quadrotor systems over the last five years. This project addresses the problem of simultaneously finding optimal paths and assignments of aerial robots to goals in a setting where robots are identical. We minimize the maximum cost over all robot trajectories. This problem can is relevant to missions with many tasks that have to be performed as quickly as possible and in parallel. Other representative examples of such applications include automated storage and product retrieval in warehouse applications (Enright and Wurman, 2011) and automated structure construction in which robots are commanded to fetch and place parts (Werfel et al., 2007; Lindsey et al., 2012) or when the robots themselves are used as structural elements of a larger construction (Oung and DAndrea, 2011). In all such applications, the deployment of a team of robots requires solving the task assignment or goal assignment problem in which each goal is assigned to a robot, thus generating a collision-free trajectories that guide robots to the assigned goal locations.

Since the aerial vehicles are assumed to be identical, the cost function may reflect the maximum effort over all robots or the maximum distance traveled by a robot. This is particularly important in first-response and search-and-rescue applications in which a number of robots must visit. This is known as the concurrent assignment and planning of trajectories (CAPT). A constraint on the fuel or energy that can be spent on the mission leads quite naturally to a setting where we want to minimize the maximum cost. Our goal in this project is to derive the solutions to the CAPT problem and show that the resulting computational complexity is manageable.

By definition, the CAPT problem seeks an assignment of the N (unlabeled) robots to M desired goal locations. Now, to explore the complexity of the problem, let us first consider $N = M$. Each possible assignment of robots to goals can be represented by an $N \times N$ permutation matrix and space of all possible assignments is isomorphic to $\mathbb{S}^N$ (group of all permutations). That means all possible assignments requires $N!$ evaluations which is clearly infeasible for any system with more than a few robots. There has been significant study into generating optimal solutions and useful suboptimal solutions. While there are multiple classes of the task assignment problem, we will restrict our focus to the linear assignment problem of minimizing the sum of individual costs for robotgoal pairs. We implemented the famous well-known Hungarian Algorithm (Kuhn 1955; Munkres 1957) which solves the linear assignment problem with computational complexity $\mathcal{O}(N^3)$, bounded polynomially in the number of robots. This is the most efficient known method to optimally solve the linear task assignment problem. There are few other methods as well that find near-optimal useful solutions. One of them was given by Rendl *et. al.* in 1988. Rendl uses a heuristic to minimize the sum of Euclidean distance traveled for a simplified Euclidean linear assignment problem with much lower complexity bound $\mathcal{O}(N^{\frac{5}{6}})$.

For solving the problem for trajectory planning for a multi-robot team with pre-assigned goal locations. We can use graph search techniques like A* and D* as they are extremely powerful in finding collision-free paths through an environment. Though, to consider multiple robots simultaneously and provide collision avoidance between robots requires enlarging the search space, growing exponentially with the number of robots. This was proved by Erdmann and Lazano-Perez in 1986.

We begin with the related work to Multi-Robot goal assembly problem in section 2, introducing preliminaries in Section 3. Section 4 introduces c-CAPT, a solution to the CAPT problem in obstacle-free environments. Section 5 demonstrates D-CAPT, the distributed sub-optimal version of C-CAPT. Section 6 introduces the obstacles in the given environment and implements A* and D* like algorithms. Section 7 presents the simulations and analysis of C-CAPT and D-CAPT.

II. Related Work

Multi-robot system came into application in early 1980s [7], when researchers and scientists wished to improve the efficiency of robots and achieve some complex tasks that cannot be finished by single robot. An individual robot is found to be weak at reactions to dynamic surroundings and intricate assignments. As a consequence, people paid attention to multi-robot systems. The noticeable advantage is that they are low-cost to produce and able to improve the stability and robustness by their parallel character and redundancy. Though a single robot may not be powerful enough, by cooperating in a team, multiple robots can concentrate on details respectively and compensate disadvantages of other members.

The difficulty of such problem mainly resides with the coupling between the robots paths which leads to an enormous state space and branching factor. As such, algorithms that are both complete and (distance) optimal, such as the A* algorithm and its variants, do not perform well on tightly coupled problems beyond very small ones. On the other hand, faster algorithms for finding the paths generally do not provide optimality guaranteed: Sifting through all feasible path sets for optimal ones greatly increases the search space, which often makes these problems intractable.

In order to achieve optimality, the sum of distance travelled by all robots should be minimized. To solve such problems, traditionally there were two kinds of approaches: a) Market-Based Approaches and b) Optimization-Based Approaches. Market-based approach gained a considerable attention within the robotics research community because of several desirable features, such as the efficiency in satisfying the objective function, robustness and scalability. The market-based approach is an economically inspired approach that provides a way to coordinate the activities between robots/agents. It is mainly based on the concept of auctions. In economic theory, an auction is defined by any mechanism of trading rules for exchange. An auction is a process of assigning a set of goods or services to a set of bidders according to their bids and the auction criteria. Auctions are common and simple ways of performing resource allocation in a multi-agent system. Market-based approaches for MRTA problem involve explicit communications between robots about the required tasks. Robots bid for tasks based on their capabilities. The negotiation process is based on market theory, in which the team seeks to optimize an objective function based upon robots utilities for performing particular tasks. Generally, any protocol that allows agents to indicate their interest in one or more resources or tasks is considered an auction. This makes auctions very important to consider when tackling many applications. Moreover, auctions provide a general theoretical structure for understanding resource allocation among self-interested agents.

The traditional auction method has several designs which can be used to solve multi-robot task allocation problem. This can be explained in different stages. First, the Contract Net Protocol (CNP) stage which is nothing but a task-sharing protocol in a multi-agent system. It specifies the interaction between agents for autonomous competitive negotiation through the use of contracts. Thus, CNP allows tasks to be distributed among multi-agents. The contract net protocol enables dynamic distribution of information via three methods. i) Nodes can transmit a request directly to another node for the transfer of the required information. ii) Nodes can broadcast a task announcement in which the task is a transfer of information. iii) Nodes can note, in its bid on a task, that it requires particular information in order to execute the task. Second is the announcement stage. It is the stage where an agent takes up the role of the coordinator and announces the tasks or a set of tasks to be available for bidding. This is followed by the submission stage where the individual agents/bidders communicate this value to the coordinator agent. Subsequently, in the next so-called selection stage, the job of the auctioneer is to evaluate the received bids based on an optimization strategy to determine the winning agent. Thereafter, the winning agent get assigned by a contract to execute the task and the process loops all over again. The main contribution of the contract net protocol is that it offers structuring high-level interactions between nodes for cooperative task execution. Wherever, the main drawback is that each agent is a self-interested agent; meaning that the final solution may be the best for the agents involved, but not for the group as whole.
Pros and Cons of Market-Based Approaches:
i) Efficiency: One of the greatest strengths of market-based approaches is their ability to utilize the local information and preferences of their participants to arrive at an efficient solution given limited resources. Market-based approaches have elements that are centralized and other elements that are distributed.
ii) Robustness: As mentioned previously, fully centralized approaches employ a single agent to coordinate the entire team in a multi-agent system. They may suffer from a single point of failure, and have high communication demands. Market-based approaches implemented based on decentralized paradigm do not require a permanent central coordinator agent and therefore there is no common-mode failure point or vulnerability in the system. These approaches can be made robust to several types of malfunctions, including complete or partial failures of agents.
iii) Uncertainty: Market-based systems are able to operate in unknown and dynamic environments by allowing team members to adapt cost estimates over time, and reallocate tasks when appropriate. Although market-based approaches have many advantages, they are not without their disadvantages. Perhaps the biggest drawback of market-based approaches is the lack of formalization in designing appropriate cost and revenue functions to capture design requirements. Also, negotiation protocols, developing appropriate cost functions,

and introducing relevant penalty schemes can complicate the design of the market approach. In domains where fully centralized approaches are feasible, market-based approaches can be more complex to implement,and can produce poorer solutions. Also, when fully distributed approaches suffice, market-approaches can be unnecessarily complex in design and can require excessive communication and computation.

The other method one can apply for multi-robot goal assignment is Optimization-based approach. Optimization is the branch of applied mathematics focusing on solving a certain problem in the aim of finding the optimum solution for this problem out of a set of available solutions. This set of available solutions is restricted by a set of constraints, and the optimum solution is chosen within these constrained solutions according to a certain criteria. This criteria defines the objective function of the problem that quantitatively describes the goal of the system. Deterministic techniques follow a rigorous procedure and its path and values of both design variables and the functions are repeatable. For the same starting point, they will follow the same path whether you run the program today or tomorrow. Deterministic techniques include numerical and classical methods such as graphical methods, gradient and hessian based methods, derivative-free approaches, quadratic programming, sequential quadratic programming, penalty methods, etc. They also include graph-based methods such as blind/uninformed search and informed search methods.

Stochastic techniques always have some randomness. These techniques can be classified into trajectory-based and population-based algorithms. A trajectory-based meta heuristic algorithm such as simulated annealing uses a single agent or solution which moves through the design space or search space in a piece-wise style. A better move or solution is always accepted, while a not-so-good move can be accepted with certain probability. The steps or moves trace a trajectory in the search space, with a non-zero probability so that this trajectory can reach the global optimum. On the other hand, population-based algorithms such as genetic algorithms, ant colony optimization and particle swarm optimization use multiple agents to search for an optimal or near-optimal solution. By reviewing the literature, it was found that different optimization approaches have been used in order to solve the general task allocation problems. In, a mixed integer linear programming optimization approach was used in order to allocate heterogeneous robots for maximizing the coverage area of the regions of interest. Also in, a mixed integer linear programming approach was used for solving the task allocation problem in the context of UAV cooperation.

By C-CAPT, we seek to find optimal trajectories (not necessary always) and safe trajectories for multiple robots concurrently in a known environment. This is done by minimizing a cost functional, which in this case will be the sum of squared distances between every start to every goal. Given all the start locations, a goal is assigned to it such that the path from the robot to goal is optimal.

In D-CAPT, the algorithm is used to plan an optimal, safe trajectory in an environment where the interaction between the robots are local. A sensing range h is defined for communication such that $h > 2\sqrt{2}R$, where R is the radius of the robot. When robots come within communicating range of each other, they learn new information about the neighbors. D-CAPT is definitely a sub-optimal version of c-CAPT. It is important to note that both c-CAPT and D-CAPT algorithms are not complete in nature. Since, the CAPT algorithm is an optimization technique, there can be situations (such as: goal of robot A is the starting position of robot B and *vice versa* in a two robot goal assignment problem) where the optimization function might not converge, hence the algorithm cannot be said as complete. Moreover, we will see in the next sections, why the D-CAPT is more computationally expensive, harder to implement and sub-optimal solution to goal-assignment problem as compared to c-CAPT.

III. Problem Formulation

The problem statement was to generate optimal and safe trajectories for multiple robots in a known obstacle-free two or three dimensional environment. Optimality is defined as the minimization of the total distance traveled by all the robots to reach their respective assigned goal locations. In order to solve this problem, we will decouple the goal assignment and the trajectory planning problems. This is done using Concurrent Assignment and Planning of Trajectories (CAPT).
We consider N identical robots with radius R navigating from initial locations to M desired goal locations in an n-dimensional Euclidean space. Let the integer set between 1 and positive integer Z be defined as $\mathcal{I}_Z \equiv \{1, 2, ..., Z\}$. For example, if there are $M = 4$ goal locations, then $\mathcal{I}_M = \{1, 2, 3, 4\}$. The location of the i^{th} robot is specified by $\mathbf{x}_i \in \mathbb{R}^n$, $i \in \mathcal{I}_N$ and similarly the j^{th} goal locations is specified by $\mathbf{g}_j \in \mathbb{R}^n$, $j \in \mathcal{I}_M$.

The state vector can be defined as $\mathbf{X} \in \mathbb{R}^N$:

$$X(t) = [x_1(t)^T, x_2(t)^T, ..., x_N(t)^T]^T$$

We define the *assignment matrix* $\phi \in \mathbf{R}^{N\times M}$ which assigns agents to goals:
$\phi_{i,j} = 1$, if robot i is assigned to goal j and $\phi_{i,j} = 0$, otherwise.

Also, we define clearance δ as the minimum space between any pair of robots at any time during the trajectory:

$$\delta(t) = inf||x_i(t) - x_j(t)|| - 2R$$

Now, to ensure the collision avoidance for all robots, we require the clearance to always be greater than zero:

$$\delta(t) > 0, \quad \forall\ t \in [t_0, t_f]$$

We define $\mathcal{K}$ as the convex hull of initial locations and goal locations with the *Minkowski* sum of a ball of radius R:

$\mathcal{K}$ =conv $(\{\mathrm{x}_i(t_0)|\ i \in \mathcal{I}_N\} \cup \{g_j|j \in \mathcal{I}_M\})$

IV. Centralized-CAPT

C-CAPT seeks to find optimal trajectories $\gamma^*(t)$ which minimize a specified cost functional:

$$\gamma^*(t) = argmin \int_{t_o}^{t_f} L(\gamma(t))\ dt$$

The assumptions required for C-CAPT are as follows:

1) All robots are homogeneous and interchangeable with no preference of goal locations.
2) Each robot is a set of points confined to a ball with radius *R*.
3) The region κ defined above is obstacle-free.
4) Robots are fully actuated differentially flat systems, do not have any actuation error, and always have perfect state knowledge.

Given M robots (completely centralized system) and N destinations, we wish to find the most optimal and a collision-free solution for the system. There can be three cases:(a) $M < N$, (b) $M = N$ and (c) $M > N$

Fig 1 shows **C**-CAPT simulation for M=N case for time $t = 0$ and $t = t_f$. The algorithm[1] for **C**-CAPT is given in the **Algorithm 1**.

```
while $M > N$ do
  C-CAPT
  remove goals assigned by $\psi^{*\ 1}$
C-CAPT
```

Algorithm 1: C-CAPT Algorithm

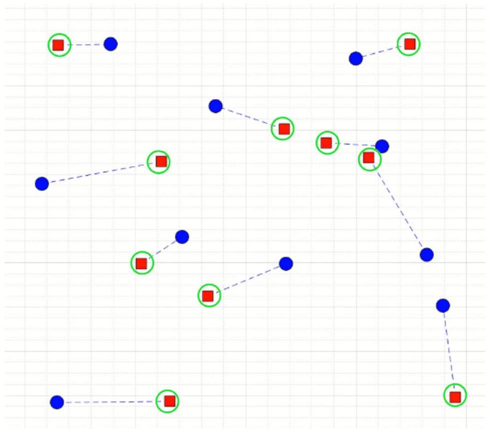

Fig. 1. Centralized-CAPT simulation: (a) $t = 0$

A. The Hungarian Algorithm

The Hungarian Algorithm is a combinatorial optimization algorithm that solves the assignment problem in polynomial time.

The assignment problem is the problem of choosing an optimal assignment of n men to n goal locations, assuming

[1]Standard notations are from the paper [2]

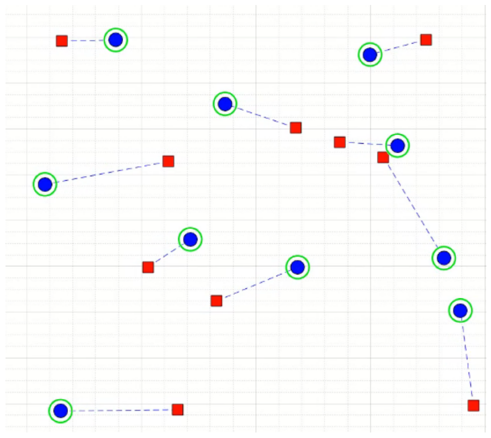

Fig. 2. Centralized-CAPT simulation: $t = t_f$

that the distances are given of each robot to goal locations. An optimal assignment is considered the one which makes the sum of the distances between robot and assigned goal locations is minimum.

Let d_{ij} be the distance between robot R_i to goal location G_j. Elements of a matrix are *independent* if no two of them lie in the same line. We wish to choose a set of n independent elements of the matrix d_{ij} so that the sum of these elements is maximum. Let d = max (d_{ij}) and let $x_{ij} = d - d_{ij}$.We are interested in choosing a set of n independent elements of the matrix A = $[x_{ij}]$ such that the sum of these elements is minimum.

The following algorithm applies the above theorem to a given $n \times n$ cost matrix to find an optimal assignment.

1) Subtract the smallest entry in each row from all the entries of its row.
2) Subtract the smallest entry in each column from all the entries of its column.
3) Draw lines through appropriate rows and columns so that all the zero entries of the cost matrix are covered and the minimum number of such lines is used.
4) *Test for Optimality:* (i) If the minimum number of covering lines is *n*, an optimal assignment of zeros is possible and we are finished. (ii) If the minimum number of covering lines is less than *n*, an optimal assignment of zeros is not yet possible. In this case, proceed to the next step.
5) Determine the smallest entry not covered by any line. Subtract this entry from each uncovered row, and then add it to each covered column. Return to 3.

V. Decentralized-CAPT

D-CAPT is used to plan an optimal and safe trajectory in an environment and reassign goal locations. Since the interactions between the robots are local, a sensing range, h is suitably defined so that robots can ignore neighbors outside this sensing range. It should also be small enough so that the robot can learn new information about its neighbors. When a robot comes within h of a neighbor, it communicates information about their current state and their assigned goals. We also require the following assumptions apart from the assumptions stated in Section III:

1) All robots can exchange information about their current state and their assigned goals to other robots closer than sensing range, h.
2) Given M robots and N destinations, $N = M$, where each goal location is assigned to exactly one robot.

The equation used to compute the trajectory is

$$\mathbf{x}_i(t) = \left(1 - \frac{t - t_c}{t_f - t_c}\right)\mathbf{x}_i(t_c) + \left(\frac{t - t_c}{t_f - t_c}\right)\mathbf{f}_i \qquad (1)$$

where $t \in [t_c, t_f]$, $t_0 \leq t_c < t_f$ (t_c is the current time)

A. *D-CAPT Algorithm*

```
**Data:** M: Number of Robots in space
**Data:** N: Number of desired destinations in space
**Result:** Getting the collision-free optimal solution
Compute trajectory using equation (1)
$\mathcal{U}_i = \mathcal{C}_i(t_0)$
$t_{prev} \leftarrow t_0$

while $t < t_f$ do
  for $j \in \mathcal{C}_i(t_c)$ do
    if $j \notin \mathcal{C}_i(t_{prev})$ then
      $\mathcal{U}_i = \mathcal{U}_i \bigcup j$    $\mathcal{U}_i = \mathcal{U}_i \bigcap \mathcal{C}_i(t_c)$
    request $\mathbf{x}_j(t_c)$ and $\mathbf{f}_j$ from agent $j$
    if $\mathbf{u}_{i,j}^T \mathbf{w}_{i,j} < 0$ then
      send to robot $j$ to change goal to $\mathbf{f}_i$
      $\mathbf{f}_i \leftarrow \mathbf{f}_j$
      $\mathcal{U}_i = \mathcal{C}_i(t_c)$
      Recompute trajectory using equation (1)
    $\mathcal{U}_i = \mathcal{U}_i \; \forall j$

if robot $j$
sends $\mathbf{f}_j$ as the new robot $i$ goal then
  $\mathbf{f}_i \leftarrow \mathbf{f}_j$
  $\mathcal{U}_i = \mathcal{C}_i(t_c)$
  Again, recompute trajectory using equation (1)
  $\mathcal{U}_i = \mathcal{U}_i \; \forall j$
$t_{prev} \leftarrow t_c$
```

Algorithm 2: **D**-CAPT algorithm [2]

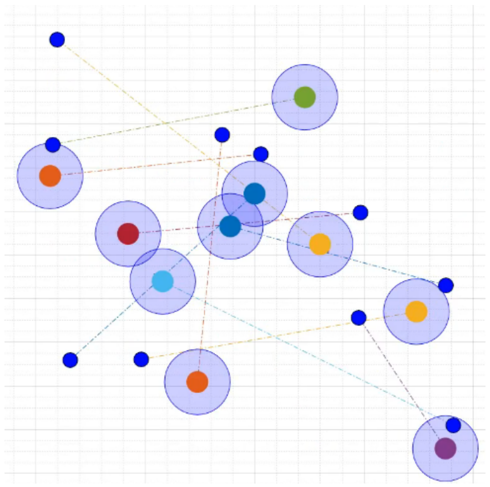

Fig. 3. Decentralized-CAPT simulation: $t = 0$

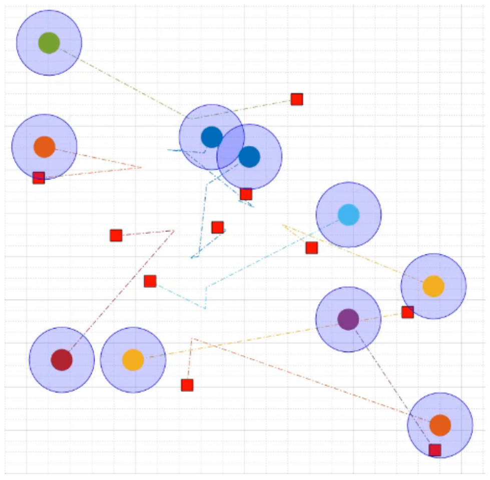

Fig. 4. Decentralized-CAPT simulation: $t = t_f$

B. *Effect of communication delay ($\mathcal{T}_d$) on h in D-CAPT*

If $\mathcal{T}_d << \frac{\delta d}{V}$, where δd is the incremental distance traveled with a velocity V, assuming that V stays constant over δd, there is negligible effect on h, *i.e.*, we do not have to increase h to ensure safety. In contrast, if $\mathcal{T}_d \geq \frac{\delta d}{V}$, which is highly realistic if we have a noisy medium and acknowledgement-reacknowledgement protocol is being used for data exchange, the safety of the robots is compromised, increasing the probability of collision. In order to restore the safety to zero-delay case, we will have to grow h to

$$h` = h + V_{max}\mathcal{T}_d$$

where V_{max} is the maximum velocity of the robot.

VI. Results

In this section, we demonstrate the implementation and results of C-CAPT and D-CAPT algorithms. Multiple number of experiments have been performed by varying sensing range, h, number of goal locations, M and number of robots, N.

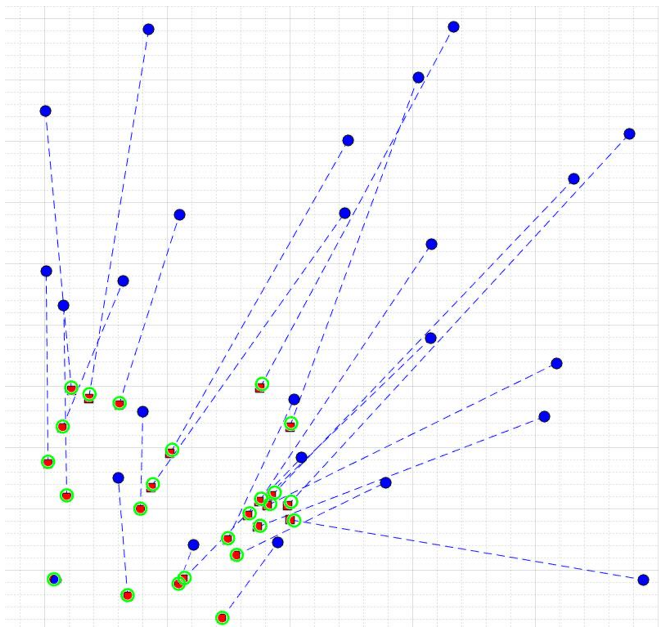

Fig. 5. Centralized-CAPT simulation: $t = 0$

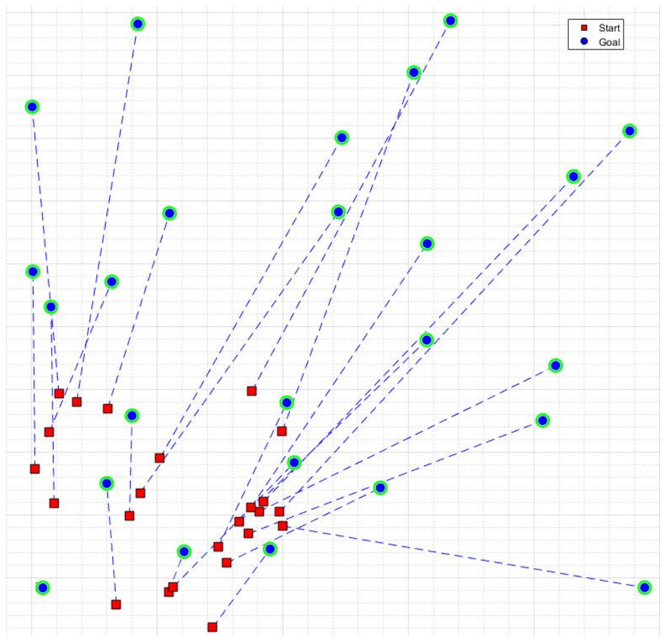

Fig. 7. Centralized-CAPT simulation: $t = t_f$

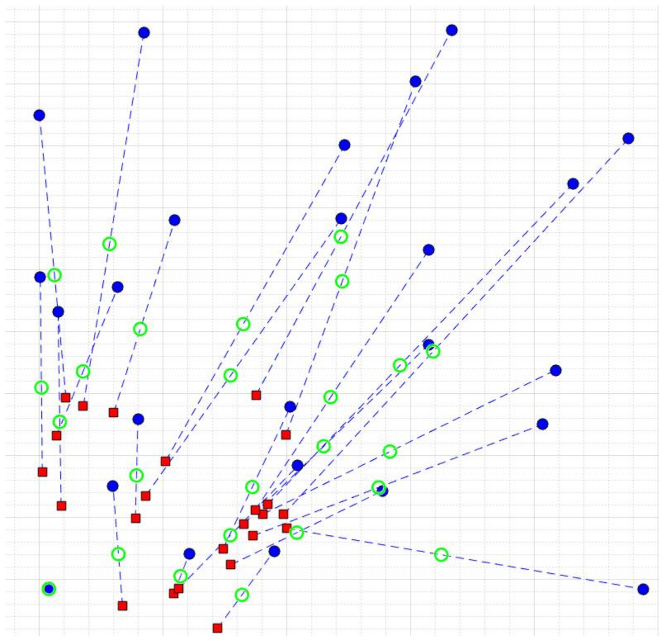

Fig. 6. Centralized-CAPT simulation: $t = t_i$

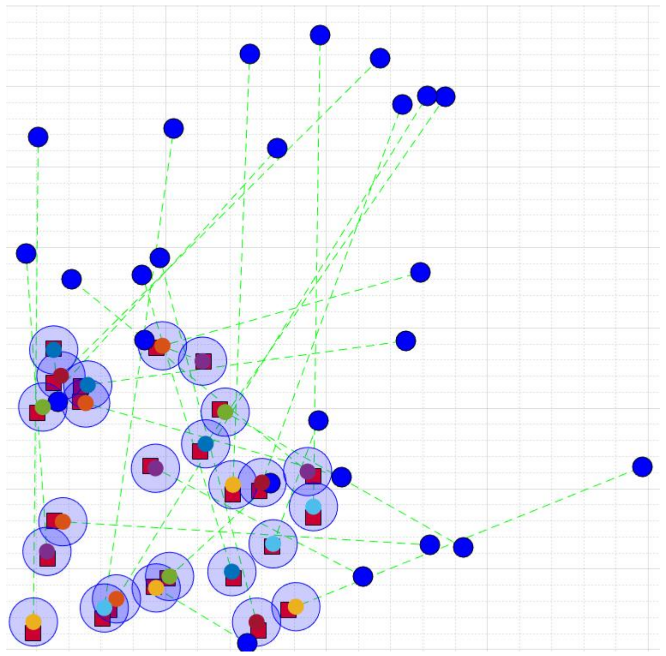

Fig. 8. Decentralized-CAPT simulation: $t = 0$

VII. Discussion

We simulated C-CAPT, a centralized solution to the goal assignment problem in multi-robot systems. We first developed C-CAPT that considers least-square-velocity (along the trajectory) as the cost function and find the *argmin* of that function. This was an optimal collision-free solution for CAPT problem. We then studied and simulated D-CAPT that relies on the exchange of information between robots within the communication range. The algorithm requires local reassignment and re-planning across a pair of robots when maximum distance traveled can be reduced while simultaneously increasing minimum clearance. It was a sub-optimal solution but a collision-free trajectory set. This project is based on Ref. [2] research that was published in 2014 in IJRR. It is the state-of-the-art algorithm for the planning of trajectories in multiple robots which will boost our path planning concepts in research perspective. We have later added obstacles and simulated the obstacle-course environment to solve the assignment problem. Rather than going on a straight line path in the Euclidean space, each quadrotor will finds their path using A* algorithm. Moreover, to solve the CAPT algorithm, we reviewed the *Hungarian method* which could solve the assignment problem in polynomial time ($\mathcal{O}(n^3)$).

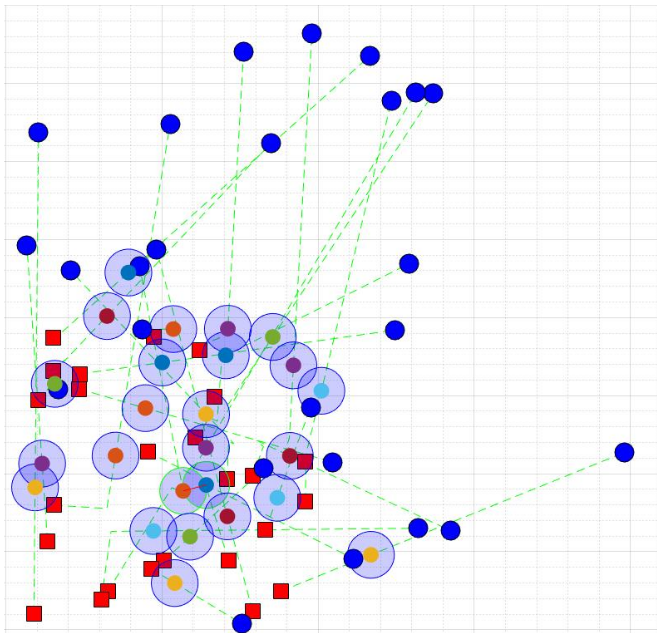

Fig. 9. Decentralized-CAPT simulation: $t = t_i$

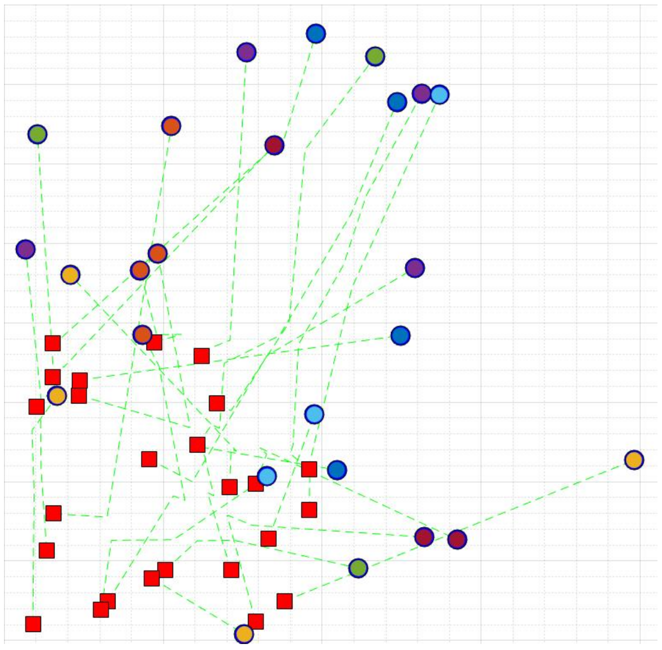

Fig. 10. Decentralized-CAPT simulation: $t = t_f$

As we've seen from the previous sections D-CAPT results in sub-optimal paths when compared with C-CAPT. In D-CAPT, it is assumed that the memory X of each robot is enough to store its respective goal location. However, in real life each robot has sufficient memory to store more than one goal location. However, in real life each robot has sufficient memory to store more than one goal location. When more than two robots cluster together. Robot 1 and 2 are in each-others sensing range and so are 2 and 3. If each robot can store all the significant (in terms of cost) locations it has been to or explored and the respective goal locations and this information can be shared among all the robots via an ad-hoc like network. We can solve the path planning problem using

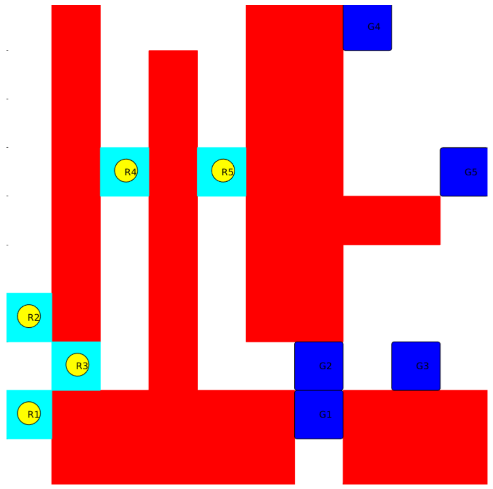

Fig. 11. Centralized-CAPT simulation with obstacles: $t = 0$

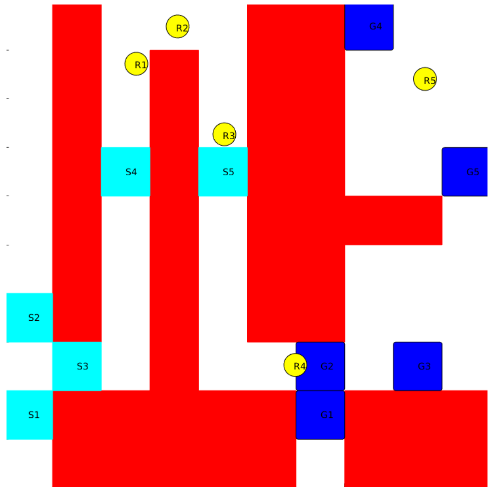

Fig. 12. Centralized-CAPT simulation with obstacles: $t = t_i$

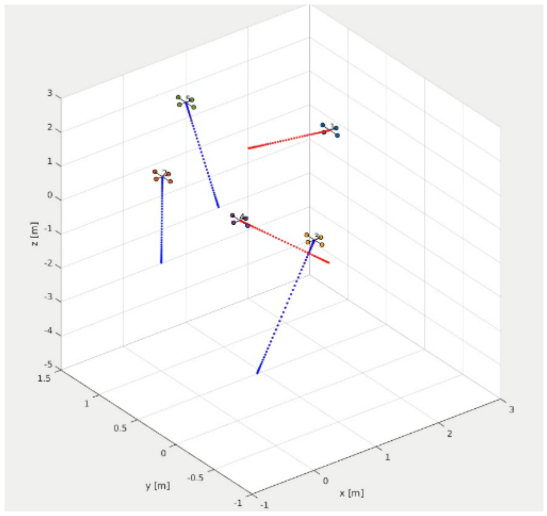

Fig. 13. Centralized-CAPT: Simulating multiple Quadrotors $N = M$

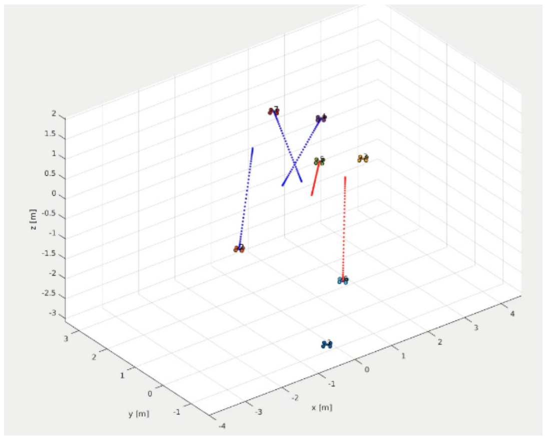

Fig. 14. Centralized-CAPT: Simulating multiple Quadrotors $N \neq M$

the Hungarian algorithm and Dijkstra/A* algorithm for path planning. However a simple swap is much faster than solving the Hungarian algorithm on the fly. We can model the time to perform Hungarian algorithm as a delay d and to ensure safety we will have to increase h according to the previous section. This methodology will use up more resources in terms of memory and processing power on each robot, however on the plus side we will be reducing the amount of sub-optimality.

A. Merits and Demerits of D-CAPT

In D-CAPT, there is no involvement of any centralization. There can be one leader robot which can be sensed by all other robots in order to calculate their respective trajectories. Hence we do not need the knowledge of the entire map. Computational and memory complexity of operations reduces because Hungarian algorithm $(O(N^3))$ is not used. Computations are done online. Hence it can adapt to changes in the environment. But, in most cases, the path obtained from D-CAPT is never optimal. We can see in fig. 10, robots follow trajectories which are highly inefficient in terms of time and distance travelled. Moreover, the communication delays is not explicitly modelled. This can cause collisions. Also, we can state that the number of messages sent becomes very large as the number of robots grow, $i.e.$, the same message will be sent to multiple robots which is inefficient. This can be avoided by using our proposed methodology to run C-CAPT locally using an ad-hoc network. In all the D-CAPT videos the robot extent is shown in orange color, the communication range is shown in green color, start locations are shown as blue squares, yellow lines denote the trajectories followed and the red stars show the goal locations. The robot number is shown inside the orange circle and the current assigned goal is shown as a number next to the red stars. Some special initial conditions (start locations) will lead to non collision free trajectories. So, we have to modify the trajectories to follow a specific sequence of trajectories if it detects an immediate collision.

Recapitulating, the centralized CAPT, unlike D-CAPT, performed as expected by generating collision free, close-to-complete, time parameterized optimal trajectories satisfying the boundary conditions. The Hungarian algorithm decreased the time needed to solve the assignment problem. C-CAPT was tested on 2-dimensional circular robots and on 3-dimensional quadrotors. C-CAPT was also tested on $N = M$, $N > M$ and $M = 2N$, it performed as expected. C-CAPT had to be run twice for the last case. Due to the obvious disadvantages of C-CAPT being in real-world the whole environment is seldom known, a decentralized solution was proposed (D-CAPT). D-CAPT performed as expected to change the goal locations as to avoid collisions. However, as the number of robots increases, D-CAPT becomes inefficient in terms of number of messages sent and hence a solution to combat this was proposed. Also the effect of time delay on D-CAPT was studied. Effect of various parameters of C-CAPT and D-CAPT with respect to the number of robots was studied. In future, we would try to solve the goal assignment problem using both supervised learning techniques such as LSTM/RNN networks or unsupervised learning like Reinforcement learning (Q-Learning).

References

[1] V. Kumar and N. Michael, "Opportunities and challenges with autonomous micro aerial vehicles", IJRR, vol. 31, no. 11, pp. 1279-1291, Aug 2012.

[2] M. Tuprin, N. Michael and V. Kumar, "CAPT: Concurrent assignment and planning of trajectories for multiple robots", IJRR, vol 33, no. 1, pp. 98-112

[3] Chaimowicz L, Michael N and Kumar V (2005) Controlling swarms of robots using interpolated implicit functions. In: Proceedings of the IEEE International conference on robotics and automation , Barcelona, 18−22 April 2005, pp. 2487

[4] J. Fink, N. Michael, S. Kim, and V. Kumar, "Planning and control for cooperative manipulation and transportation with aerial robots", IJRR, vol. 30, no. 3, pp. 324, Sept. 2010

[5] M. Bernard, K. Kondak, I. Maza, and A. Ollero, "Autonomous transportation and deployment with aerial robots for search and rescue missions", Journal of Field Robotics, vol. 28, no. 6, pp. 914, 2011.

[6] Lynne E. Parker, Path planning and motion coordination in multiple mobile robot teams, Encyclopedia of Complexity and System Science, Springer (2009).

[7] C. Goerzen , Zhaodan Kong, and Bernard Mettler. A survey of motion planning algorithms from the perspective of autonomous UAV guidance, Journal of Intelligent and Robotic Systems, Vol. 57. No. 14 (pp. 65-100, 2010.

[8] Preetha Bhattacharjee, P. Rakshit, I. Goswami, A. Konar, and, A. K. Nagar, Multi-robot path-planning using artificial bee colony optimization algorithm, Third World Congress on Nature and Biologically (2011).

[9] Algorithms for the Assignment and Transportation Problems, James Munkres, Journal of the Scoiety for Industrial and Applied Mathematics, Vol. 5, No. 1 (Pg 32-38)

[10] Hungarian Algorithm for Linear Assignment Problems (V2.3), http://www.mathworks.com/matlabcentral/fileexchange/20652-hungarian-algorithm-for-linear-assignment-problems–v2-3-

[11] On Multi Robot Task Allocation, PhD dissertation by Brian Paul Gerkey(2003).

[12] . Li, Y. M. Yang, and J. L. Lian, Layered task allocation in multi-robot systems, in the IEEE 2009 WRI Global Congress on Intelligent Systems , Xiamen, China, May 2009, pp. 6267

[13] . T. Tian, M. Yang, X. Y. Qi, and Y. M. Yang, Multi-robot task allocation for fire-disaster response based on reinforcement learning, in the IEEE 2009 International Conference on Machine Learning and Cybernetics, Baoding, China, July 2009, pp. 23122317.

[14] Yu, Jingjin, Soon-Jo Chung, and Petros G. Voulgaris. "Target assignment in robotic networks: Distance optimality guarantees and hierarchical strategies." IEEE Transactions on Automatic Control 60.2 (2015): 327-341.

[15] Y. T. Tian, M. Yang, X. Y. Qi, and Y. M. Yang, Multi-robot task allocation for fire-disaster response based on reinforcement learning, in the IEEE 2009 International Conference on Machine Learning and Cybernetics, Baoding, China, July 2009, pp. 23122317