A Summary of Multi-Robot Planning Methods

David A Smart  
Clark School of Engineering  
University of Maryland, College ParkCollege Park, Maryland, USA  
dsmart1@terpmail.umd.edu

*Abstract*—Multirobot teams have clear advantages over single robot systems, however, they also introduce inherent challenges in harmonization. A variety of methods have been formulated for the planning and control of multirobot systems. This paper summarizes the various multirobot planning methods, mentions the challenges remaining to be solved, and presents several “state-of-the-art” algorithms.

Keywords—robotics, planning, multirobot, swarm, quadcopters

# Introduction: Inspiration and Motivation

The advantages of multiple agent systems can be seen in insects, animals, and even humans, so it is clear that teams of robots should have advantages over individual robots. In theory, multiple robot systems can accomplish tasks which single robots cannot (or may be cheaper than a single complex robot capable of accomplishing the task), can accomplish them more quickly and efficiently, are more tolerant to failures, and they are more versatile than any single robot [1], [2], [3]. Sensor fusion from multiple agents can also reduce the uncertainty of measurements for tasks such as detection and localization [2], [4], [5], [6]. However, all multi-agent systems also have an inherent coordination challenge, which can put in plain terms as “who is going to do what?” The decomposition of a global task into subtasks and the efficient allocation of tasks to agents is a whole new problem, unique to multi-agent systems. And even once that is completed, there is still an increased complexity in motion planning for several robots because each robot is a moving obstacle to each of the other robots.

Research in multirobot systems began in the late 1980s and a significant amount of work has been accomplished since then, and yet many challenges remain [4]. This paper attempts to summarize the various approaches and their tradeoffs as well the problems which are still open for research and improvement. This paper also devotes a section to quadcopter planning because quadcopters hold great potential for multirobot systems.

The remainder of this paper is organized as follows: Section II explains the method used to research the topic of multi-robot systems; Section III gives some basic explanations of acronyms and terms, including the special subset of multi-robot systems known as swarm robotics; Section IV reviews the various steps and methods of solving the multi-robot planning problem; Section V mentions the challenges remaining to be concurred; Section VI lists several so called “state-of-the-art” planning algorithms; Section VII delves into multiple quadcopter specific planning; and finally Section VIII provides a summary of the topics discussed.

# Methods

## In this section, the process used to research the topic of multirobot systems is explained.

## First, a search was conducted to find if any scholarly journals exist solely for multi-agent planning. None were found. Although journals exist solely for multi-agent systems and journals exist exclusively for robot planning methods, none could be found for exactly multi-agent planning.

## Second, a search was conducted to find if any survey papers had already been written on multi-agent planning. And many were found from as far back as 1998 to as recently as 2017. The concepts covered, and the works referenced by those survey papers severed as a starting point for the remainder of the research.

## Next, an exhaustive search on IEEE Explore and Springer-Link as well as Google Scholar was conducted using the keywords, “Homogenous, Heterogenous, Centralized, Decentralized, Decoupled, Sample-Based, Fast, Optimal, Complete, and Efficient” independently and in various combinations. This produced an enormous volume of interesting journal and conference papers, which were first skimmed to determine algorithm classification and evaluate credibility. Most focused on the distributed and/or decoupled planning approach and did not consider heterogeneous teams. These key terms will be explained in the sections bellow.

## Lastly, the papers were read and included as support to the review papers (because often the authors included a concise evaluation of the prior related works), examples of the many algorithms summarized, or examples of “state-of-the-art” methods.

# Terms and Classification

In this section provides explanations of acronyms and terms, including the subset of multirobot systems known as swarm robotics.

## Acronyms

Like most technical fields, people involved in the research of multi-robot systems have devised a plethora of acronyms that can be quite confusing if not known. Therefore, the foremost acronyms are spelled out in the subsection below.

Multi-robot systems (MRS) are any system of robots consisting of two or more robots. Multi-agent systems (MASs) is a most general term, where the agents can be stationary computers or software agents, which cannot physically interact with their environment [2]. This paper will exclusively use the word robot (except in the case of centralized computation, where a computer is typically used), however, in the literature, the word agent is often used interchangeably with robot. In fact, the first acronym presented in the following paragraph uses agent in place of robot.

Multi-agent planning (MAP) can be divided into multi-robot task planning (MRTP), and Multi-robot motion planning (MRMP) [2]. Multi-robot task planning can also be further divided into multi-robot task decomposition (MRTD) and Multi-robot task allocation (MRTA) [2].

Despite the prevalence of these acronyms in the literature, they shall not be used in the remainder of this paper to avoid confusion.

## Swarm Robotics

At first glance, one may assume that swam robotics is the same as multirobot systems and admittedly there is a good deal of overlap, however, there are also some key distinctions to be made.

Multi-robot systems, referring to any number of robots greater than one, can be differentiated from swarms which refers to many robots. There is no universally accepted threshold, but a swarm could be any group larger than 20 [7]. The motivation for swarms of robots is the same as multi-robot systems with the belief that an increase in the number of robots will correspond to an increase in the benefits of a multi-robot system. Therefore, a major concern in swarm robotics is scalability. To this end, swarms must be composed of homogeneous groups which are restricted to distributed decision making and limited communication among neighbors [7]. The terms used here (homogeneous and distributed) shall all be explained in the following two sub-sections.

The main source of inspiration for swarm robotics is insects, but also includes colonies of single-celled organisms, such as bacteria. The goal of swarm robotics is to mimic these biological systems as closely as possible. This leads to a few more factors that differentiate swarms from the general multi-robot systems. Often swarm research is focused on 1) using robots that are incapable of completing tasks individually, and 2) using implicit communication (through sensing) to achieve an emergent behavior [1], [2], [7].

## Robot Types

Homogeneous simply means the same throughout, and heterogeneous is the opposite. Therefore, it only takes one robot to be different from the rest for the group to be considered heterogeneous.

Homogeneous and heterogeneous are often associated with the physical structure of the robots, but the distinction is determined by capability [2]. If the robots in the group have identical capabilities then they are homogeneous, regardless of their form. And the corollary is true for heterogeneous teams; if the robots have identical physical forms, but different capabilities, then the group is heterogeneous.

The potential advantage of heterogeneous teams is naturally evident. Tasks can be done better by robots which are better suited, even if not designed specifically for that specific task.

## Computation and Decsision Making Structures

The computations and decisions being referred to in this section occur in every stage of the planning process: task decomposition, task allocation, or motion planning.

There are two main divisions in decision making structures: centralized and decentralized. The names are self-explanatory. Centralized methods have a single agent (usually an offboard supercomputer) responsible for making decisions whereas decentralized methods have multiple robots making decisions. Decentralized methods can be further divided into hierarchical, distributed, and factored [1]. Hierarchical methods have several supervisory robots that each make decisions for several subordinate robot, and distributed methods have each robot make decisions for itself based only on local knowledge. In the factored method, several robots decide for a single robot [8]. Despite being purely semantic, it is worth mentioning that some consider hierarchical methods to be hybrid or centralized and decentralized and consider distributed and decentralized to be one and the same [2]. Another form of “hybridization” is in when a robot temporarily takes on the role of auctioneer in a decentralized approach [9].

Centralized approaches have the potential to be complete and to find globally optimal or pareto-optimal (one of several equally optimal) solutions because it has a global view of the world but if the central agent fails the entire team fails, which negates one of the desired advantages of a robot team [3], [10], [11]. Physical distance and obstacles between robots can also prevent direct communication between robots, requiring communication chains to relay information to and from a centralized agent [8].

Decentralized approaches have no guarantees of optimality but are significantly faster and are therefore quicker to react to changes in the environment and will not fail due to any individual failure [3], [10], [11]. Fully distributed approaches take full advantage of parallel processing and require less communication between the robots, which can be a major advantage since a limited bandwidth prevents robots from sharing large amounts of data simultaneously [2], [3], [12], [13].

When multi-robot planning research began, the work primarily focused on centralized and homogeneous robot systems but have transitioned to decentralized and heterogeneous robot systems [9].

# Planning

Multi-robot planning is often divided into three steps: 1) task decomposition, 2) task allocation, and 3) motion planning. In this section, the three steps and the plethora of methods devised to solve the multi-robot planning problem are reviewed and compared.

## Task decomposition

The first step in multi-robot planning is the decomposition of the group’s global task into subtasks. A simple example is dividing up an area to be explored into equally sized regions according to the number of robots in the team.

These subtasks are not always as simply divided as that example. The tasks may be such that they can be accomplished by an individual robot (“individual”), several robots working independently and concurrently (“group”) or sequentially (“partitioned”), or several robots working together (“team”) [11]. Another potential task classification is “complex”, where the task could be completed by an “individual” but may also be more efficiently handled as a “team”.

Within task decomposition and assignment, one can make another distinction between roles and tasks, where roles can be defined by set of tasks which are generally assigned indefinitely, whereas tasks are transient [11]. For a clear example of what a role is, consider offensive and defensive positions on a sports team. Roles are not always considered in the planning methods because they are not applicable to every mission.

Methods of task decomposition are often not discussed in the literature and are beyond the scope of this paper.

## Task allocation

The second step in multi-robot planning is dividing the tasks among the robots. Despite the division between task allocation and motion planning made here and elsewhere, the two are truly co-dependent, because to make the correct task assignments, the plans must be known. A complete plan is not worth computing in highly uncertain or dynamic environments which will require re-planning or when the task is time-sensitive, but this obviously can result in sub-optimal task assignments [3], [11].

As stated in the previous subsection, these tasks may be such that they can be accomplished by an individual robot, several robots working independently, or several robots working together. Even the relatively simple case of assigning each robot to a single “individual” task is not a simple matter. It is a global optimization problem of minimizing either the sum, average, or maximum cost of the individual robots. This cost can be time discrepancy in task completion, path length, or energy consumption [9], [13]. In the coupled planning scheme (to be discussed further in the next subsection), every permutation of task assignment must be checked. Although this has the potential for optimality it obviously becomes an intractable approach as the number of robots in the group increases.

Decentralized task allocation is currently dominated by market-based methods, many of which use the “contract net protocol” [2]. Market-based methods are based on the idea of free-market economics, where resources and tasks have measurable worth [3]. Market-based methods use what is known as an auction, where robots make bids for a task based on the resources they expect to use to accomplish the task. Auctions are held by the supervisors in a hierarchy or temporary auctioneers in a distributed organization. In the case of a highly dynamic environment where suboptimal task assignments are bound to occur, it has been proposed to allow for re-auctioning (or peer-to-peer trading) of task assignments [3], [14]. However, switching from one task to another can also be problematic, therefore a cost associated with switching tasks must also be included in the bids [3].

To improve the efficiency of these auctions, minimum bid values can be required to limit the number of bidders, and time-limits can be set so that the auction does wait for a non-optimal robot to complete a plan only to lose the bid. Many market-based algorithms also assign utilities values to the completion of tasks, which the robots are each trying to win. This functions in a similar manner to a minimum bid threshold, by limiting the number of robots which bid for a certain task. For example, one research team solved the problem of task assignment for the mission of exploration by assigning a utility to unexplored regions which was dependent on the number of robots moving toward that region [15].

Many other variations on the standard market-based methods have been proposed and implemented with success in increasing the efficiency. Auctions have been expanded to assign multiple tasks at once, such as a sequence of goal points for exploration [16]. Bidding has been conducted at multiple levels, tournament style, where the first level is a group of neighboring robots [17]. Auctions have been “bi-directional”, where the bidders themselves are part of the process in determining the winning bid [18]. Auctions can even be conducted probabilistically with higher probabilities corresponding to lower plan costs [19].

Another method which is becoming increasingly popular for decentralized task allocation is game-theory methods. At first glance, game-theory methods may appear to be identical to market-based methods, but they are subtly more complicated. Game-theory methods attempt to optimize both execution cost and planning time with budget limits and deadlines for individual bidders. But the added complexity comes from the auctioneer estimating future bid values based on the individual robot’s budgets and deadlines in order to determine when to close an auction. Game-theory based methods have been shown to improve upon the market-based methods both in efficiency and in optimality [20].

Another group of methods, called threshold-based methods, are primarily used for optimizing the number of robots within a swarm assigned to a certain role. These methods are based on the concepts of “motivation” and “shame”, where a robot is more motivated to assign itself to a certain role (or ashamed of not claiming that role) if it has not taken up the role despite being notified of the need [11]. In these methods it is possible for robots to incessantly oscillate between roles. To minimize switching, one can forbid robots from switch roles for a period of time but determining how long to disallow role switching while retaining the ability of the team to quickly react to environmental changes is difficult [11].

The one weakness of all the methods discussed above is the assignment of “team” and “complex” tasks. A distributed algorithm known as S+T was introduced for this purpose. In this algorithm, a robot which has already been assigned a task can ask other robots for help in the form of “services” [21]. The current best known to the author uses a centralized agent and dynamic programming [22].

Heterogenous teams also make task allocation more difficult because the planner(s) must understand the robot’s unique capabilities and how that factors into task efficiency. If centralized, the planner must understand the unique capabilities of all the robots, but if decentralized, the robots only need to understand themselves, making decentralized methods preferable for heterogeneous teams. One way to simplify task assignment is to assign permanent roles based on robot capabilities. Another approach is to split the assignment into two steps, where initially robots are assigned every task they can accomplish (according to their unique capabilities) [23], and then using market-based peer-to-peer trading to reduce the number of assignments per robot optimally [24].

## Motion Planning

The final step is motion planning, also known as path planning and trajectory planning. In single-robot planning, this is the only phase of the planning process considered. Often, even in multi-robot planning, journal papers will only focus on this part of the planning problem.

The challenge in multirobot planning is not the planning itself, which is substantially equivalent to single-robot planning. This can be an optimal algorithm such as A\* or a suboptimal method such as cell decomposition, potential fields [25], or roadmaps. The challenge in multirobot planning is coordination; that is, ensuring the plans work together. Multirobot motion planning approaches are commonly divided into coupled and decoupled algorithms according to how they ensure this coordination. Coupled algorithms conduct planning and coordination simultaneously and have the potential for completeness and optimality. Decoupled (also known as unthreaded) algorithms conduct planning and coordination separately which results in sub-optimal solutions (in terms of global optimality) [1]. Although each robot may start with an individually optimal path before considering conflicts with other robots, when coordinating, the plans optimality is lost.

It should be stressed that coupled planning is not limited to centralized robots, and decoupled planning is not limited to decentralized robots. Plan-merging is a decentralized coupled planning method, where planning is conducted in an incremental fashion by all the robots until each has a complete plan without conflict [8], [26], [27], [28]. However, this requires near-constant communication between all robots, and is therefore not often considered. Once such planner was published in 2009 using a spanning tree (generated from a roadmap) to plan for 100s of robots [29].

Unfortunately, the time required for coupled methods scales exponentially with the number of robots due to an exponential increase dimensionality of the configuration space. Even sample-based methods, which are not complete, may be intractable for coupled planners with more than 10 robots [10]. Comparatively, decoupled planners can compute solutions orders of magnitude faster than coupled planners [10] making more popular.

In 2011 a somewhat successful attempt was made to reduce the computation time of the coupled method. The algorithm known as M\* retains the completeness and optimality of an A\* solution while reducing the dimensionality of the configuration space by grouping the robots according to which ones could collide [30]. The M\* algorithm extended the practicality of coupled algorithms such that 10 robots could be planned for in under 5 minutes and 100 robots in under 25 minutes. Another centralized-coupled algorithm published in 2011 was capable of plan paths for 100s of robots in a graph with only n+2 nodes in under 5 seconds, suggesting the limitations of coupled algorithms may not be as severe as previously believed [31].

Decoupled algorithms can be subdivided by the way in which they settle conﬂicts as active and passive. In active methods, conflicts are dealt with during the initial planning. In passive methods, conflicts are handled as they are detected during plan execution [4].

Active decouple planning methods can be further subdivided into pre-planning, post-planning, and iterative-response [8]. Pre-planning (also known as static coordination) uses preset conventions sometimes referred to as social laws, such as “keep right” to ensure that plans will work in harmony [2], [12], [32]. Pre-planning is not general or adaptable to new situations and is often used in well-known environments such factories. Post-planning repairs plans to avoid conflict by localized re-planning or velocity-tuning. In 2013, a post-planning algorithm was published which reduced the computation time by checking for conflicts and re-planning as soon as individual robots completed their path plans, rather than waiting for all the robots to finish their plans [33]. This algorithm was applied to a team of heterogeneous UAVs with “super conflicts” (path collisions between more than two robots) in simulation as well as in real life [33]. Iterative-response is where the plan for each agent is made in sequence according to a priority while taking constraints from those robots which have already completed their plans [8]. To be clear, in post-processing, and iterative-response, the higher priority robots are moving obstacles for the lower priority robots which must plan accordingly [10]. Iterative-response essentially reduces the multi-robot planning problem to several single-robot planning problems [34].

Velocity-tuning was mentioned in the preceding paragraph as a scheme for coordination in post-planning but deserves a more thorough explanation. A new path-time space is created using a parameterization of the current robot’s path plan and obstacles are created according to the times when higher-priority obstacles intersect that path plan. This path-time space can then be searched using any of the standard planning methods but lends itself well to cell decomposition [10].

Priorities are used in both post-planning and iterative response. Assigning priorities, prior to planning is simple and fast but results in an algorithm which is not complete as it cannot guarantee a solution to the problem of opposing traffic in a narrow passageway (and if velocity tuning is used, this problem is impossible) [10]. A better method is to optimize a cost criterion like in task allocation. One such method is to assign priorities according to the distance between the start and goal positions of the robots [34]. Another maximizes the number of robots that can move in straight lines to their goals [35]. Others have taken advantage of simple randomized searches to find a good prioritization in a short amount of time [36]. Now, it is worth pointing out that minimizing the global cost is best for efficiency, but minimizing the maximum individual cost is the best for time-critical tasks [9].

In the passive method, when robots come within a certain radius of each other they exchange plans and evaluate whether a collision shall occur and then make the necessary adjustments as a post-processor would [2], [37]. An alternative to plan sharing is sensing and treating each other as moving obstacles, typically through “velocity obstacles” [38], [39]. This method is quite practical in cases of slow moving ground vehicles but is impractical for fast moving aerial vehicles.

Despite being one of the motivations for multi-robot teams, “team” tasks prove to be a significant challenge in multi-robot planning. One approach is to create plans for synchronous “team” actions beforehand, which can then be strung together in sequence during planning to complete a “team” task [40].

## Communication

Communication is an integral part of the planning processes and could be discussed in the preceding section, the ideas were extensive enough to necessitate an entire subsection.

Multi-robot teams cannot broadcast everything with everyone, because communication is not completely free –bandwidth is limited. The key is to have the robots communicate only the information that is most pertinent to the problem at hand [11]. This can be accomplished by only sharing information with certain individuals within a certain branch of a hierarchy or within groups assigned to a certain role. Another approach is to only share information which was requested with those who made the request, but this raises the question of how much information is required to coordinate [13]? The bottom line is, explicit communication loads increase with the number of robots.

Since 2009, the concept of privacy has become prevalent in research as well. Privacy is usually spoken of in terms of robots working together from competing agencies, where they do not want to give away any information that could give their competitors an advantage [8]. This idea of privacy honestly sounds pointless. Privacy (and encryption) for military robots, where one does not want to give away one’s plans to the enemy, appears to be a much more realistic constraint, but is not often discussed.

Although, explicit communication is the predominant method, there is an alternative, known as implicit communication. Implicit communication does not have the issue of communication load, but it is impossible to guarantee the accurate exchange of information and is limited to distributed computation. Implicit communication is typically reserved for swarm robotics and passive decoupled methods, but it is possible that a combination of implicit and explicit communication could make coordination more efficient and robust [2].

# Current Challenges and Open Questions

Despite the great extent of research conducted in multi-robot planning, many challenges are still open to be solved and optimized. This section attempts to cover some of the major challenges to be addressed in the coming years.

One challenge is detection and response to robot failures. If communication is lost, how can one know if the robot was unsuccessful in its mission or failed entirely [3]? If the robot did fail entirely should it be retrieved and repaired or left behind? This second question is obviously cost and application dependent. If the robot is cheap or if it would be risky or retrieve it, then it is probably not worth scavenging.

Despite proving elusive for many year, formations, have been concurred by decentralized planners using local communication [17]. Moving in formations however is still a challenge. Some centralized planning schemes such as that in [41xxx] have proved successful, but all the computational limitations of centralized planners still exist for those solutions and the challenge remains for decentralized planners.

Another issue, commonly referred to as “crossing the reality gap”, is the challenge of transferring from simulation to implementation on real-world robots. The concern always exists that an approach which produces amazing results in simulation may never be realizable with real robots because simulation models make simplifying and idealized assumption and are therefore inaccurate [2], [3]. In 2016 a coupled algorithm was published which could create a time-continuous trajectory’s which minimized energy consumption without making simplifying (linearizing) assumptions about the vehicle dynamics or constraints [42]. To date, most algorithms are implemented on relatively small teams of robots despite being used for hundreds or even thousands in simulation. For example, in 2009 a plan-merging centralized planner was able to plan for 141 robots in simulation but was only implemented on 3 robots [29]. Another distributed formation-control algorithm was used to simulate 106 robots in simulation but was only implemented on 5 quadrotors [43]! Part of the issue is lack in availability of mass-produced robots and the required maintenance to preserve their functionality. It is currently impossible to possess hundreds of working robots.

Another challenge worth mentioning, is the representation of tasks. Often a group’s mission will be assigned by a human, who might think of the task in words which cannot be directly understood by the robots. Translation from human thought to tasks which are understood and are decomposable by the robot teams is an essential aspect still open to research [44].

Finally, what happens when humans are part of the team [3]? How can task allocation and motion planners account for humans? What kind of information exchange will be required? What about “team” tasks, where a human must work with a robot side-by-side?

# Quadcoptor Planning

Aerial vehicles are more versatile than ground vehicles in that terrain is largely not an issue. Despite their limitations in range, payload, and speed, quadcopters are the most agile aerial vehicles, capable of hovering and changing directions almost arbitrarily, making them the well-suited for a variety of tasks and easier to plan for than fixed wing vehicles. Quadcopters are also relatively common and inexpensive, making them well suited for a team or swarm. Therefore, planning for multiple quadcopters is focused upon for this section. Rather than focusing on coordination like in section IV, this section talks about the motion planning algorithms used to create smooth trajectories that satisfy the real-world dynamic constraints of quadcopters.

In 2011 an algorithm enabling real-time quadcopter trajectory planning through a sequence of specified waypoints (3D positions and yaw angles) while satisfying smooth jerk was formulized as a Quadratic Program [45]. The following year, the algorithm was expanded for use on a team of heterogeneous quadcopters by framing the collision avoidance aspect as Mixed-Integer Constraints on the Quadratic Program [46]. This method had guaranteed optimality, but the computational time of solving the mixed-integer quadratic program limited the algorithm to a small team in an environment with few obstacles. Another research group applied this to a larger number of quadcopters by controlling groups in formation rather than individuals, but this semantic because the groups acted as a whole, rather than as individuals [47].

As a tangent, it is possible for a team or swarm of robots to move as a group while retaining individuality. This is known as “flocking”. In 1987, a man named Craig Reynold’s formulated thee criteria for “flocking”: attempt to stay close, avoid collisions with, and match the velocity of one’s neighboring robots [48]. In 2006, Reynolds’ three criteria were incorporated into a potential-field path planning algorithm [49]. In 2017, with the addition of a preplanning rule to “always go right” when bypassing obstacles, this was used to develop a distributed swarm algorithm [50]. This was further developed and has proved effective for controlling swarms of real-life quadcopters [51].

The problem of planning for a heterogeneous team of quadcopters (without the constrain of smooth jerk) was solved later in 2012 by using Sequential Convex Programming to solve linearized non-convex collision avoidance constraints [52]. This alternative formulation and approach reduced the computation time significantly but was not optimal nor complete, and could not handle non-robot obstacles. In 2015 another research team proposed an improvement on the method which incrementally constrained the Sequential Convex Program and thereby, increased the probability of successful path planning while reducing the planning time, and while accounting for obstacles [53]. However, even this was quite limited in completeness for the number of convex obstacles.

In 2018, a two-phase planning scheme was published using a Level-Set method to initialize a piecewise-linear trajectory, and a Pseudo-Spectral method to then create a smooth-jerk trajectory [54]. This is the fastest planner yet and can be applied to numerous robots but has only been implemented for simulated 2D spaces whereas the preceding methods were applied to 3D spaces on real robots. A similar two-phase trajectory planner was successfully used for 3D motion planning for a team of 32 quadcopters in a cluttered environment but required several minutes of planning time [55].

All the preceding algorithms were centralized-decoupled iterative-response approaches, which relied on constant communication with a computer station. A similar planning method was implemented in a distributed fashion for a relatively large group of quadcopters [56] and was very recently extended for use on a heterogeneous team [57].

I was stated earlier that the main challenge in multi-robot motion planning is the coordination of robots to avoid collisions. This is an even more critical issue for aerial vehicles for which collisions result in the destruction of the robots involved. To overcome this planning challenge, some have reframed it as a controls problem. It was recently proposed not to do any collision avoidance in planning, but instead to equip quadcopters with protective cages and controllers which can recover from in-air collisions [58].

Despite the advantages listed before, Quadcopters are not without limitations, so sometimes it is advantageous to make a team consisting of quadcopters and other autonomous vehicles, such as ground vehicles. For example, in target tracking, a quadcopter can more quickly identify and locate a target, but would require a ground vehicle to intercept the target. For an example of such UAV-UGV coordination in pursuit, see [59].

# Conclusions

This section serves as a quick review of what was covered in this paper.

Swarm robotics is a subset of multi-robot systems for large numbers of homogeneous robots which are controlled in a distributed fashion.

Centralize computation uses an off-board computer to control all the robots, and decentralized planning has each robot computing for itself. Centralized computation has the potential for completeness and optimality, whereas decentralized does not, but centralized planning is slower and poses a reliability risk.

Tasks must be divided among the robots and the most common distributed method is to use a market-based auction where robots bid for a task. There are many variations of this approach. Team tasks and “complex” tasks are difficult to allocate. Roles are collections of tasks and could be useful in the assignments of heterogeneous teams.

Coupled planning treats all the robots as a system and thus accounts for inter-robot collisions while doing motion planning, and decoupled planning treats the robots as individuals and separately tries to coordinate their plans. Coupled planning has the potential to be optimal and complete, whereas decoupled planning does not, but coupled planners are too slow for many real-time applications. (Centralized and Decentralized computation do not necessarily correspond to coupled and decoupled planning, but they often do). Decoupled planning can be done in several ways: pre-planning, post-planning, and iterative-response. There is also the option in certain applications to coordinate through plan adjustments during plan execution.

There are limitations to communication which must be considered, which motives the use of distributed computation with local interaction as well as implicit communication via sensing.

Despite the massive amounts of research conducted in multi-robot planning, much work is still to be done. One example is planning for multiple quadcopters, which has the potential for many applications, but current planning methods are far from perfected. Other areas of future research include the incorporation of formations and human-robot interactions.

##### References

1. C. Y. Uny, A. S. Fukunaga, and A. Kahng, "Cooperative Mobile Robotics: Antecedents and Directions," *Autonomous Robots,* vol. 4, no. 1, pp. 7-27, 1997.
2. Z. Yan, N. Jouandeau, and A. A. Cherif, "A survey and Analysis of Multi-Robot Coordination," *Int. Journal of Advanced Robotic Systems,* vol. 10, no. 12, pp. 399-417, 2013.
3. D. M. Bernardine, R. Zlot, N. Kalara, and A. Stentz, "Market-Based Multirobot Coordination: A Survey and Analysis," *Proc. of the IEEE,* vol. 94, no. 7, pp. 1257-1270, 2006.
4. Y. Cai, and S. X. Yang, "A survey on Multi-Robot Systems," *World Automation Congress (WAC)*, 2012.
5. Dan Wu, and H. Su, "An Improved Probabilistic Approach for Collaborative Multi-Robot Localization," *In Proc. of the IEEE Int. Conf. on Robotics and Biomimetics (ROBIO)*, 2009.
6. R. Madhavan, K. Fregene, and L. E. Parker, "Distributed Heterogeneous Outdoor Multi-Robot Localization," *In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA),* vol. 1, 2002.
7. E. Sahin and W.M. Spears (Eds.): Swarm Robotics WS 2004, LNCS 3342, pp. 10–20, 2005.
8. A. Torreno, E. Onaindia, A. Komenda, and M. Stolba, "Cooperative Multi-Agent Planning: A Survey," *ACM Computing Surveys (CSUR),*  vol. 50, no. 6, 2017.
9. X. Jia, and M. Q-H. Meng, "A Survey and Analysis of Task Allocation Algorithms in Multi-Robot Systems," *In Proc. of the IEEE Int. Conf. on* *Robotics and Biomimetics (ROBIO)*, 2013, pp.2280-2285.
10. S. M. Lavalle, *Planning Algorithms,* Cambridge University Press, 2006. [E-book]
11. A. Campbell, and A. S. Wu, "Multi-Agent Role Allocation: Issues, Approaches, and Multiple Perspectives," *Autonomous Agents and Multi-Agent Systems,* vol. 22, no. 2, pp. 317-355, 2011.
12. M. De Weerdt, and B. Clement, "Introduction to Planning in Multiagent Systems," *Multiagent and Grid Systems*, vol. 5, no. 4, pp. 345-355, 2009.
13. E. Todt, G. Rausch, and R. Suarez, "Analysis and Classification of Multiple Robot Coordination Methods," *In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA),* vol. 4, 2000.
14. R. Zlot, A. Stenz, M. B. Dias, and S. Thayer, "Multi-robot Exploration Controlled by a Market Economy," *In Proc. of the IEEE Int. Conf. on. Robotics and Automation (ICRA), vol. 3*, 2002.
15. W. Burgard, M. Moors, C. Stachniss, and F. E. Schneider, "Coordinated Multi-Robot Exploration," *IEEE Transactions on Robotics,* vol. 21, no. 3, pp. 376-386, 2005.
16. J. Faigl, M. Kulich, and L. Preucil, "Goal Assignment using Distance Cost in Multi-Robot Exploration," *In Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2012.
17. N. Michael, M. M. Zavlanos, V. Kumar, and G. J. Pappas, "Distributed Multi-Robot Task Assignment and Formation Control," *In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2008.
18. T. Song, X. Yan, A. Liang, K. Chen, and H. Guan, "A Distributed Bidirectional Auction Algorithm for Multirobot Coordination," *In Proc. of the Int. Conf. on Research Challenges in Computer Science (ICRCCS)*, 2009, pp. 145-148.
19. A. Jevtic, A. Guiterrez, D. Andina, and M. Jamshidi, "Distributed Bees Algorithm for Task Allocation in Swarm of Robots," *IEEE Systems Journal,* vol. 6, no. 2, pp. 296-304, 2012.
20. R. Cui, J. Guo, and B. Gao, "Game Theory-Based Negotiation for Multiple Robots Task Allocation," *Robotica,* vol.31, no. 6, pp. 923-934, 2013.
21. A. Viguria, I. Maza, and A. Ollero, "S+T: An Algorithm for Distributed Multirobot Task Allocation Based on Services for Improving Robot Cooperation," *In Proc. Of the IEEE Int. Conf. on* *Robotics and Automation (ICRA)*, 2008.
22. Julie A. Adams, "Coalition Formation for Task Allocation: Theory and Algorithms," *Autonomous Agents and Multi-Agent Systems*, vol. 22, no. 2, pp. 225-248, 2011.
23. D. Di Paola, A. Gasparri, D. Naso, and F. L. Lewis, "Decentralized Dynamic Task Planning for Heterogeneous Robotic Networks," *Autonomous Robots,* vol. 38, no. 1, pp. 31-48, 2015.
24. D. Di Paola, D. Naso, and B. Turchiano, "Consensus-Based Robust Decentralized Task Assignment for Heterogeneous Robot Networks," *American Control Conference (ACC)*, 2011.
25. H. G. Tanner, and A. Kumar, "Towards Decentralization of Multi-Robot Navigation Functions," *In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA),* 2005.
26. R. Alami, F. Robert, F. Ingrand, and S. Suzuki, "Multi-Robot Cooperation Through Incremental Plan-Merging," *In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA),* vol. 3, 1995.
27. R. Alami, F. Ingrand, and S. Qutub, "Planning Coordination and Execution in Multi-Robots Environment," *In Proc. of the 8th Int. Conf. on Advanced Robotics (ICAR)*, 1997.
28. R. Alami, F. F. Ingrand, and S. Qutub, "A Scheme for Coordinating Multi-robots Planning Activities and Plans Execution," *ECAI,* 1998.
29. M. Peasgood, C. M. Clark, and J. McPhee, "A Complete and Scalable Strategy for Coordinating Multiple Robots within Roadmaps," *IEEE Transactions on Robotics,* vol. 24, no. 2, pp. 283-292, 2008.
30. G. Wagner, and H. Choset, "M\*: A Complete Multirobot Path Planning Algorithm with Performance Bounds," *In Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2011.
31. R. Luna, and K. E. Bekris, "Efficient and Complete Centralized Multi-Robot Path Planning," *In Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2011.
32. J. Wawerla, and R. T. Vaughan, "A Fast and Frugal Method for Team-Task Allocation in a Multi-Robot Transportation System," *In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2010.
33. M. Cap, P. Novak, M. Selecky, J. Faigl, and J. Vokrinek, "Asynchronous Decentralized Prioritized Planning for Coordination in Multi-Robot System," *In Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2013.
34. J. P. Van den Berg, and M. H. Overmars, "Prioritized Motion Planning for Multiple Robots," *In Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2005.
35. S. J. Buckley, "Fast Motion Planning for Multiple Moving Robots," *In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 1989.
36. M. Bennewitz, W. Burgard, and S. Thrun, "Optimizing Schedules for Prioritized Path Planning of Multi-Robot Systems," *In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA),* vol. 1, 2001.
37. K. Azarm, and G. Schmidt, "Conflict-Free Motion of Multiple Mobile Robots based on Decentralized Motion Planning and Negotiation," *In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA),* vol. 4, 1997.
38. J. Van den Berg, M. Lin, and D. Manocha, "Reciprocal Velocity Obstacles for Real-Time Multi-Agent Navigation," *In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2008, pp. 1928-1935.
39. J. Snape, J. Van den Berg, S. J. Guy, and D. Manocha, "Independent Navigation of Multiple Mobile Robots with Hybrid Reciprocal Velocity Obstacles," *In Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2009, pp. 5917-5922.
40. E. G. Jones, B. Browning, M. B. Dias, B. Argall, M. Veloso, and A. Stentz, "Dynamically Formed Heterogeneous Robot Teams Performing Tightly-Coordinated Tasks," *In Proc. of the IEEE Int. Conf. on Robotics and Automation ( ICRA)*, 2006.
41. S. Liu, D. Sun, and C. Zhu, “Coordinated Motion Planning for Multiple Mobile Robots Along Designed Paths with Formation Requirement”, *IEEE/ASME Trans. On Mechatronics,* vol. 16, no. 6, pp. 1021-1031, 2011.
42. A. J. Hausler, A. Saccon, A. P. Aguiar, John Hauser, and A. M. Pascoal, "Energy-Optimal Motion Planning for Multiple Robotic Vehicles with Collision Avoidance," *IEEE Trans. on Control Systems Technology,* vol. 24, no. 3, pp. 867-883, 2016.
43. S. Bandyopadhyay, S.-J. Chung, and F. Y. Hadaegh, "Probabilistic and Distributed Control of a Large-Scale Swarm of Autonomous Agents," *IEEE Transactions on Robotics,* vol. 33, no. 5, pp. 1103-1123, 2017.
44. T. Arai, E. Pagello, and L. E. Parker, "Advances in Multi-Robot Systems." *IEEE Trans. on Robotics and Automation,* vol. 18, no. 5, pp. 655-661, 2002.
45. D. Mellinger and V. Kumar, “Minimum Snap Trajectory Generation and Control for Quadrotors,” *In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2011.
46. D. Mellinger, A. Kushleyev, and V. Kumar, “Mixed-Integer Quadratic Program Trajectory Generation for Heterogeneous Quadrotor Teams,” *In Proc. of the IEEE Int. Conf. on Robotics and Automation (IRCA)*, 2012, pp. 477–483.
47. A. Kushleyev, D. Mellinger, C. Powers, and V. Kumar, "Towards a Swarm of Agile Micro Quadrotors," *Autonomous Robots,* vol. 35, no. 4, pp. 287-300, 2013.
48. C. W. Reynolds, "Flocks, Herds and Schools: A Distributed Behavioral Model," *ACM SIGGRAPH computer graphics,* vol. 21, no. 4, 1987.
49. R. Olfati-Saber, "Flocking for Multi-Agent Dynamic Systems: Algorithms and Theory," *IEEE Transactions on automatic control*, vol. 51, no. 3, pp. 401-420, 2006.
50. S. Iovino, A. Savvaris, A. R. Vetrella, G. Fasano, and D. Accardo, "Implementation of a Distributed Flocking Algorithm with Obstacle Avoidance Capability for UAV Swarming," *AIAA Information Systems*, 2017.
51. G. Vasarhelyi, Cs. Viragh, G. Somorjai, N. Tarcacai, T. Szorenyi, T. Nepusz, and T. Vicsek, "Outdoor flocking and formation flight with autonomous aerial robots, " *Proc. of Int. Conf. on Intelligent Robots and Systems (IROS)*, 2014.
52. F. Augugliaro, A. P. Schoellig, and R. D’Andrea, “Generation of Collision Free Trajectories for a Quadrocopter Feet: A Sequential Convex Programming Approach,” *In Proc. of the IEEE/RSJ Int. Conf. on Intelligetnt Robotic Systems*, 2012, pp. 1917–1922.
53. Y. Chen, M. Cutler, and J. P. How, “Decoupled Multiagent Path Planning via Incremental Sequential Convex Programming,” *In Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2015, pp. 5954–5961.
54. D. R. Robinson, R. T. Mar, K. Estabridis, and Gary Hewer, “An Efficient Algorithm for Optimal Trajectory Generation for Herterogeous Multi-Agent Systems in Non-Convex Environments,” *IEEE Robotics and Automation Letters,* vol. 3, no. 2, 2018.
55. J. A. Preiss, W. Honig, N. Ayanian, and G. S. Sukhatme, “Downwash-Aware Trajectory Planning for Large Quadcopter Teams,” *arXiv Preprint,* 2017.
56. J. A. Preiss, W. Honig, G. S. Sukhatme, and N. Ayanian, "Crazyswarm: A Large Nano-Quadcopter Swarm," *In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2017.
57. M. Debord, “Heterogeneous Trajectory Planning with Crazyflie and BigQuad Deck,” *Bitcraze*, Apr. 2, 2018, [Online] Available: [www.bitcraze.io/2018/04/heterogeneous-trajectory-planning-with-crazyflie-and-bigquad-deck/](http://www.bitcraze.io/2018/04/heterogeneous-trajectory-planning-with-crazyflie-and-bigquad-deck/). [Accessed April 26, 2018]
58. Y. Mulgaonkar, A. Makineni, L. Guerro-Bonilla, and V. Kumar, "Robust Aerial Robot Swarms Without Collision Avoidance," *IEEE Robotics and Automation Letters,* vol. 3, no. 1, pp. 596-603, 2018.
59. H. Yu, R. W. Beard, M. Argyle, and C. Chamberlain, "Probabilistic Path Planning for Cooperative Target Tracking using Aerial and Ground Vehicles," 2011 *American Control Conference (ACC)*, 2011.