Article

Scheduling and Path-Planning for Operator Oversight of Multiple Robots

Sebastián A. Zanlongo [1], Peter Dirksmeier [2], Philip Long [3,*], Taskin Padir [2] and Leonardo Bobadilla [1]

[1] Knight Foundation School of Computing and Information Sciences, Florida International University, Miami, FL 33199, USA; szanl001@fiu.edu (S.A.Z.); bobadilla@cs.fiu.edu (L.B.)
[2] Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115, USA; dirksmeier.p@husky.neu.edu (P.D.); t.padir@northeastern.edu (T.P.)
[3] Robotics & Automation Department, Irish Manufacturing Research, D24 WC04 Dublin, Ireland
* Correspondence: philip.long@imr.ie

Abstract: There is a need for semi-autonomous systems capable of performing both automated tasks and supervised maneuvers. When dealing with multiple robots or robots with high complexity (such as humanoids), we face the issue of effectively coordinating operators across robots. We build on our previous work to present a methodology for designing trajectories and policies for robots such that a few operators can supervise multiple robots. Specifically, we: (1) Analyze the complexity of the problem, (2) Design a procedure for generating policies allowing operators to oversee many robots, (3) Present a method for designing policies and robot trajectories to allow operators to oversee multiple robots, and (4) Include both simulation and hardware experiments demonstrating our methodologies.

Keywords: human robot interaction; multi-robot coordination; humanoid robots; scheduling and coordination; supervisory control

check for updates

Citation: Zanlongo, S.A.; Dirksmeier, P.; Long, P.; Padir, T.; Bobadilla, L. Scheduling and Path-Planning for Operator Oversight of Multiple Robots. *Robotics* **2021**, *10*, 57. https://doi.org/10.3390/robotics10020057

Academic Editor: Beno Benhabib

Received: 21 February 2021
Accepted: 2 April 2021
Published: 6 April 2021

1. Introduction

Multi-robot systems are making a significant impact on fundamental societal areas. From oceanic exploration to border surveillance, from robotic warehousing to precision agriculture, and from automated construction to environmental monitoring, collaborating groups of robots will play a central role in the coming years [1]. In some of these scenarios, however, due to technical, ethical, regulatory or safety issues [2], one or more humans should monitor or help the robot during the execution of its tasks in certain critical parts . These critical segments of the robot trajectory can be kinematically or dynamically complex maneuvers, locations near obstacles, or regions where sensing is poor.

Most teleoperated systems assume more than one human operator per robot. More than one human may be required for each subsystem in more complex scenarios, such as humanoids or mobile manipulator teleoperation (e.g., manipulation, locomotion, head positioning). For instance, in control rooms in Unmanned Aerial Vehicles missions, several operators are needed to operate a single drone. While it may remain infeasible to remove altogether the portion of a task which cannot be automated, we can efficiently allocate human *attention* in these portions. As an application of our ideas, we envision scenarios where a single operator can coordinate a group of automated construction machinery, several agricultural pieces of equipment or even a production line of industrial robots. Indeed, the recent pandemic has demonstrated teleoperation control paradigms are favoured in situations where remote presence is desirable and where complexity precludes the use of fully autonomous systems [3]. Effectively combining the cognitive capabilities of a human operator with robot physical capacities [4] has provided great benefits in industrial applications [5,6] and more general methodologies are needed.

Another motivation behind our ideas is *robot-assisted search and rescue*. In traditional mobile robot search and rescue operations using unmanned vehicles, operators' ratio to robots is commonly 2 to 1 [7]. More recently, motivated by disasters such as the Fukushima nuclear plant, there has been a need for robots with larger degrees of freedom that can operate in environments designed for humans. Concretely, lessons learned analyzing human-robot interfaces used by different teams in the DARPA Robotics Challenge (DRC) [8], gave two important reasons motivating our ideas to reduce the number of operators: (1) fewer operators reduces confusion and coordination overhead, and (2) the number of human errors (one of the main sources of problems in the DRC [9]) is reduced.

This work addresses the question of operator attention at critical moments of a teleoperation task. In most situations, an operator is only required in specific parts of a robot's operation. Knowing this, we can schedule these operator interactions so that a single operator can perform multiple tasks. Thus, our objective is to develop a planning strategy for the remote task involving multiple robots such that the operator can pay sufficient attention to each robot during critical operations. This work's contributions are: extending our preliminary ideas from [10,11] in the following directions: First, we analyze the complexity of this problem. Secondly, we present a sampling-based approach that allows us to design policies for many teleoperators instead of a complete algorithm that only works for a small set of operators. Thirdly, we allow re-planning of the robot's task alongside the operator. Finally, we present the results of both simulated and physical experiments using mobile robots and a humanoid.

Our work deals with planning for robots using a small set of operators that can help the robot when needed. As a convention, we will use the term "robot" throughout this paper; however, the method is formulated within the robots' configuration space and is agnostic to the robot type. It can model multiple robots and a single robot with multiple degrees of freedom such as a humanoid robot. To the best of our knowledge, our contribution is one of the few that attempts to formalize operator scheduling problem using a geometric approach.

The rest of the paper is organized as follows: Section 2 discusses relevant related literature. Section 3 describes the preliminaries and formulates the problems of interest. Section 4 describes algorithms to solve the formulated problems in the previous section. In Section 5, we present an extension of the solution in Section 4 which can also re-plan robot trajectories. Section 6 presents both software and hardware experimental results, and a case study is provided in Section 7. Conclusions and future directions are presented in Section 8.

2. Related Work

Teleoperation is an established robot control paradigm with particular widespread use in surgical and medical operations [12]. While teleoperation is classically defined using 1:1 operator to robot ratio [13,14], there is a growing need for systems that facilitate operator oversight of multiple robots [15]. This work focuses on reducing operator supervision to only temporally critical passages. Previous work took a different approach, aiming to reduce the cognitive burden on the operator by, for instance, using virtual fixtures [16,17]. Virtual fixtures create zones where the robot can operate and thus reduce the operators' supervision load. These zones can be obtained using point cloud data [18], shape primitives, [19], manually created [20], selected interactively [21] or generated on-line based on obstacle proximity and manipulator capabilities [22]. For the teleoperation of multiple agents, Farkhatdinov et al. [23] proposed a discrete switching control algorithm where an operator can trigger a switch to control different robots or different inputs, i.e., position, velocity of the same agent. In [24], the authors propose modeling operator behavior in a multi-robot control task and hypothesize that these models can be used to improve the teleoperation control strategies. Alternatively, for more complex systems, the introduction of a degree of autonomy in the robots' behavior, often denoted as shared control, can enable operators to control multi-agent or complex systems [25]. Using operators alongside partially autonomous robots yields systems that are less brittle and more effective than either one working alone [26]. A study case of a foraging task is presented in [27] where queuing techniques were used to schedule the operator's attention. In [28], Dardona et al. have shown that the output side sensor configuration of a teleoperated system is altered to reduce the workload of a remote surgeon.

While the above techniques enhance the operators' control of a robot system, we focus on complex autonomous systems that *require* supervision and analyse how this supervising burden may be reduced. Using one operator to control multiple robots has benefits in terms of cost and control coherency but leads to a higher workload and decrease in situation awareness [29]. The human operator is often overlooked when supervising robots [30], despite their workload influencing task performance [31] and in fact having long-term negative effects on well-being [32]. Indeed, J. M. Riley et al. [33] have demonstrated that an increase in the number of robots per human significantly degrades performance and situation awareness. An increase in supervision burden has been shown to increase accidents during multi-robot control trials [34]. While this may be mitigated by smart alert systems [35,36], it has been shown in [37] that too many robot systems will eventually saturate operator capabilities and, in turn, lead the operators to neglect some robots. Neglect is mainly due to the robots competing for operator attention [38]. In [39], the authors propose real-time measurements of neurophysiological parameters to estimate workload as a potential input to new forms of adaptive automation.

Our work focuses on eliminating this failure mode by judicious scheduling events that are likely to require concentrated operator attention.

We build upon our recent work [10,11], to perform multi-robot planning [40,41]. We also find complementary goals in [42] where a robot attempts to move from one location in its environment to another by calculating which obstacles can be minimally displaced to generate a feasible trajectory. In our work, we will similarly generate a coordination space, where operator "collision obstacles" must be avoided, and seek to find the minimal displacement needed to avoid them. In work by LaValle and Hutchinson [43,44], as well as by Wang et al. [45], the complexity of coordinating both many robots and operators is handled by separating the planning and scheduling aspects into two separate steps. This division greatly assists in devising a feasible solution and is echoed here as well. Our work develops techniques for planning multi-robot missions that can assist in outlining mission requirements and robot policies. There are relevant approaches such as Crandall et al. [46] which investigates the effects of allocating operator attention to robots, and [47–49] which investigate additional methods of distributing operators across robots and the effects this has. Particularly relevant to our research ideas are [50,51] where the expected behaviors of humans in an environment are incorporated into the planning phase of robots, allowing them to perform more elaborate plans than without this prediction. This argument also extends into more industrial settings, where it is often repeated, scheduled interaction between robots and operators [5]. Our work also relates to motion planning approaches that generate joint plans for humans and robots [52–54].

3. Preliminaries

We start with a set of m of bodies, which can be kinematic chains or mobile robots, $\mathcal{A} = \{\mathcal{A}^1, \cdots, \mathcal{A}^m\}$. Each robot $\mathcal{A}^i \in \mathcal{A}$ has a configuration space $\mathcal{C}^i$ representing the set of all possible transformations, where the set of valid configurations is called the free space $\mathcal{C}^i_{free}$. Robots also have initial $q^i_I \in \mathcal{C}^i_{free}$ and goal $q^i_G \in \mathcal{C}^i_{free}$ configurations, where the trajectory $\lambda^i : [0, t^i_f] \to \mathcal{C}^i_{free}$ takes the robot from $\lambda^i(0)$-corresponding to q^i_I-through $\mathcal{C}^i_{free}$ to the final configuration $\lambda^i(t^i_f)$-corresponding to q^i_G, where t^i_f is the total runtime for $\mathcal{A}^i$ to execute λ^i given a dedicated operator.

When executing λ^i, $\mathcal{A}^i$ may enter critical configurations $\mathcal{C}^i_{att} \subset \mathcal{C}^i_{free}$ during which it will require one of the p operator's supervision. A conflict occurs when more than p robots require supervision at the same time. Given a range of time $T = [0, t_f]$ where the mission is executing, we will attempt to minimize $t_f = max(t_f^1, \ldots, t_f^m)$ when all robots have finished, while also providing operator attention when required.

Problem 1. ***Scheduling for Multiple Operators:*** *Given p the number of operators, a set of robots $\mathcal{A}$—each with their trajectories λ^i, and a set of critical configurations $\mathcal{C}_{att}^i$—determine a policy $\pi^i : T \to \mathcal{C}_{free}^i$ for each robot such that (1) all robots are only in critical configurations when an operator can supervise them, (2) the number of operators requested at any time does not exceed p, and (3) attempt to reduce the total runtime of the mission t_f.*

Building on this problem, we can add the following condition: Is it possible to yield a shorter mission runtime by generating alternative trajectories for bodies such that they do not require supervision simultaneously as other robots in the first place, thus avoiding operator attention "collisions" altogether? This question leads us to a concrete extension of Problem 1.

Instead of a pre-determined trajectory, we use a sequence of *waypoints* $\tau^i = [\tau_1^i, \ldots, \tau_o^i]$—where each waypoint is a specific configuration the robot must achieve, and the application-specific function $plan(A^i, \tau^i, t_{den})$ yields a trajectory that visits τ^i while avoiding $\mathcal{C}_{att}^i$ during operator-denied times t_{den}—an example of which can be found in Section 6.

Problem 2. ***Scheduling with Re-Planning:*** *Given p operators, a set of robots $\mathcal{A}$ each with a sequence of sub-goals τ^i, and a set of critical configurations $\mathcal{C}_{att}^i$. Determine a trajectory λ^i and policy $\pi^i : T \to \mathcal{C}_{free}^i$ for each robot satisfying the waypoints such that (1) robots are in critical configurations only when an operator can supervise them, (2) the number of operators requested at any time is less than or equal to p, and (3) an effort is made to minimize the ending time of the mission t_f.*

4. Scheduling Operator Attention

This section will propose solutions to Problem 1 defined in Section 3. A schematic representation of the steps of our approach is outlined in Figure 1. Details of the method will be explained below.

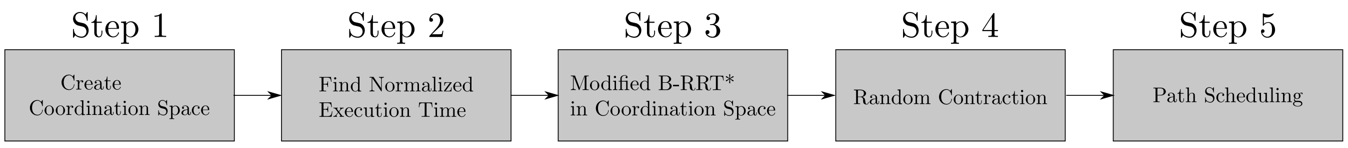

Step 1
Create Coordination Space
Step 2
Find Normalized Execution Time
Step 3
Modified B-RRT* in Coordination Space
Step 4
Random Contraction
Step 5
Path Scheduling

Figure 1. Overall steps involved in the proposed scheduling approach.

4.1. Computational Complexity of Scheduling for Multiple Operators

In our previous work [11], we described the operator scheduling problem and presented a geometric approach for its solution. There were several issues with the proposed methodology related to the computational complexity of creating the entire set of obstacles with the coordination space. To solve this problem, we give a proof sketch proving the complexity of this problem.

We prove that Problem 1 is NP-Hard by using the technique or *restriction* ([55], p. 63). An NP-Hardness proof by restriction consists of showing that a problem Π (in our case, Problem 1) contains a known NP-Hard Π' as a special case.

In our proof, Pi' is the *Multiprocessor Scheduling* problem ([55], p. 238), which consists of a set of J jobs, each job j^i has a corresponding length l^i. Given p processors, we must schedule this set of jobs so that they (1) do not overlap and (2) execute in the minimum amount of time.

Starting from problem 1 (our operator scheduling problem), assume that all possible configurations for the robot will require operator attention, meaning that the entire execution of λ^i will need an operator. This plan's runtime is t_f^i, and is analogous to the length of a job in the original Multiprocessor Scheduling problem. These jobs are scheduled and allocated to p operators, which would be the processors in the original formulation. This problem then reduces to the Multiprocessor Scheduling problem where we schedule j jobs across p processors and indicates that the problem we are trying to solve is NP-hard.

4.2. A Sampling-Based Solution

Knowing that the problem is NP-hard we ask, we will propose heuristics to find feasible solutions.

We start by by creating a *Coordination Space* $X = [0, \tilde{t_f^1}] \times \cdots \times [0, \tilde{t_f^m}]$ (following a procedure similar to [56]) representing all possible configurations of the robots along their trajectories. Each of the m axes corresponds to the normalized execution time $\tilde{t_f^i}$ of robot $\mathcal{A}^i$, given by $\tilde{t_f^i} = \dfrac{t_f^i}{max(t_f^1, \ldots, t_f^m)}$, with the position along the axis corresponding to progress along the trajectory. Let X_{obs} be the set of invalid configurations where the number of robots requesting supervision exceeds p, and $X_{free} = X \backslash X_{obs}$ be the set of all valid configurations where the number of requests does not exceed p. At $x_{init} = (0, ..., 0) \in X_{free}$ all robots are in their initial configurations, and at $x_{goal} = (\tilde{t_f^1}, ..., \tilde{t_f^m}) \in X_{free}$ all robots are in their final configuration.

We define auxiliary functions, borrowing the notation from [57]: $d(x_1, x_2)$ is the Euclidean distance between two points, and $c(\cdot)$ is the cost of a path corresponding to the sum of the pairwise Euclidean lengths of the pairwise linear points within it.

The above formulation serves to create a coordination space where the position along axes represents robot configurations and invalid configurations where multiple robots request obstacles represent an operator. This process allows us to convert the coordination problem into a path-planning problem. We must find a path $h : [0,1] \rightarrow X_{free}$ from $h(0) = x_{init}$ to $h(1) = x_{goal}$. Following h will give us an implicit representation of time with each robot's positions along their trajectory, such that each robot will move from its initial state to its goal state, with at most p robots requiring operator attention. We performed this calculation by mapping h to the trajectory λ^i corresponding to a particular robot. Define $\sigma : h \rightarrow [0, t_f^i]$, which indicates the position of the robot along its trajectory λ^i at the corresponding point of path h through X_{free}. We then perform the composition $\phi : \lambda \circ \sigma$, which yields $\phi : h \rightarrow \mathcal{C}_{free}$, mapping from the path h to $\mathcal{C}_{free}$. This allows us to determine the configuration of a robot at any point q in h via $\phi(q) = \lambda(\sigma(q))$. We can now obtain the series of configurations $\tilde{x}$ for each robot that will guarantee that at most p robots require operator attention at any given time and reduces the total run-time of the mission.

Our preliminary solution [11] required generating the entire set of obstacles within the coordination space. Here, we instead use a lazy approach which only checks sampled locations. This is combined with a modified version of the Bidirectional RRT^* originally described in [57–59], and shown in Algorithm 1 for reference. Define graphs $\mathcal{G}_a = (V_a = \{x_{init}^a\}, E = \emptyset) \in X_{free}$, $\mathcal{G}_b = (V_b = \{x_{init}^b\}, E = \emptyset) \in X_{free}$, where $x_{init}^a = x_{init}$ and $x_{init}^b = x_{goal}$. The objective will be to derive an obstacle-free path $h : [0,1] \rightarrow X_{free}$ such that $h(0) = x_{init}, h(1) = x_{goal}$. Given a user-defined function that can estimate when robots will enter a critical section $\mathcal{S} \leftarrow CriticalSegments(A)$ we can check if a point $x \in X$ is obstacle-free as in Algorithm 2, where for the point being evaluated, we iterate over each robot's critical segments (lines 3, 4) and check if the corresponding axis of x lies within the segment (line 5). If the number of collisions is greater than the number of operators (line 7), then the location is not obstacle-free. With some abuse of notation, we also use this to refer to checking if an edge is obstacle-free by sampling along the edge and checking if the samples are all within X_{free}.

The modified $BidirectionalRRT^*$ is presented in Algorithm 1. In lines 1, 2, we initialize the final path as currently being none, and the corresponding cost to be infinite. Subsequently, we perform the following procedure over N samples: Beginning with G_a—the graph starting at the origin—in lines 4, 5 we draw a randomly selected point from X_{free}.

Checking if the point lies within X_{free} is done using Algorithm 2, and select the nearest point in the graph (we use an r-tree to accomplish this efficiently). In line 6, create a point x_{new} that is closer to x_{rand} than $x_{nearest}$. Then in lines 7–9, select the r points in $\mathcal{G}_a$ that are nearest to x_{new} and sort them in order of increasing distance from x_{new}, where the sorted list L_s consists of tuples of the form (x', c', σ'), where $x' \in X_{near}$, σ' is an edge from x' to x_{new}, and c' is the cost of that path, and select the closest one with an obstacle-free path to x_{new} as in [60]. If there is a valid "best parent" —defined as the vertex with the lowest combined cost-to-come and cost-to-go—we insert it into the graph and rewire as in [60] (lines 10–13). We then attempt to connect both trees. In lines 14–17, we select the nearest vertex in the opposite graph $\mathcal{G}_b$ and attempt to draw a straight path from the newly added vertex $x_{new} \in \mathcal{G}_a$ to $\mathcal{G}_b$, if possible. We then check if the resulting path is better than our current best-path σ_{best} and update σ_{best} if necessary.

Algorithm 1: *B-RRT**

```
Input  : Coordination Space $X$, Operators $p$; Critical Segments $\mathcal{S}$; Samples N,
         Probability of early exit $p_{early} \in [0,1]$
Output : Obstacle-free path $\sigma_{best}$ through $X$
$\sigma_{best} \leftarrow \emptyset$;
$c_{best} \leftarrow \infty$;
for $i \in [0, N]$ do
    $x_{rand} \leftarrow SampleFree$;
    $x_{nearest} \leftarrow Nearest(x_{rand}, \mathcal{G}_a)$;
    $x_{new} \leftarrow Extend(x_{nearest}, x_{rand})$;
    $X_{near} \leftarrow Near(x_{new}, \mathcal{G}_a, r)$;
    $L_s \leftarrow Sort(x_{new}, X_{near})$;
    $x_{min} \leftarrow BestParent(L_s)$;
    if $x_{min} \neq \emptyset$ then
        $\mathcal{G}_a \leftarrow Insert(x_{new}, x_{min}, \mathcal{G}_a)$;
        $\mathcal{G}_a \leftarrow Rewire(x_{new}, L_s, E)$;
    end
    $x_{conn} \leftarrow Nearest(x_{new}, \mathcal{G}_b)$;
    $\sigma_{new} \leftarrow Connect(x_{new}, x_{conn}, \mathcal{G}_b)$;
    if $\sigma_{new} \neq \emptyset$ and $c(\sigma_{new}) < c(\sigma_{best})$ then
        $\sigma_{best} \leftarrow \sigma_{new}$;
    end
    $RandomContraction(\sigma_{best})$;
    $u \sim U([0,1])$;
    if $\sigma_{best} \neq \emptyset$ and $u \leq p_{early}$ then
        return $\sigma_{best}$;
    end
    $SwapTrees(\mathcal{G}_a, \mathcal{G}_b)$;
end
return $\sigma_{best}$;
```

At this point in the algorithm, we may have a valid path σ_{best} through X_{free}. We then perform *RandomContraction* as in [60] to attempt reducing the length of σ_{best}. The user may assign a probability p_{early}, corresponding to the likelihood of checking for an early-exit solution; this is to balance between the run-time of *B-RRT** and yielding a better path. We evaluate this in lines 20–23, returning a valid solution if one exists. Otherwise, we swap $\mathcal{G}_a$ and $\mathcal{G}_b$ and continue until all N samples have been drawn and return σ_{best}.

We then proceed, in Algorithm 3, by mapping h to the sequence of configurations $\tilde{x}^i$ that correspond to robot $\mathcal{A}^i$. Movement parallel to an axis corresponds to that robot moving at full speed, perpendicular segments indicate the robot is paused, and diagonal segments to velocity-tuning depending on the slope.

To the best of our knowledge, our approach is one of the first to use geometric and motion planning techniques to schedule operators' attention. Since most previous methods are based on human factor techniques or combinatorial scheduling algorithms, head-to-head comparison is difficult. Furthermore, our study cases (multi-robot control and humanoid manipulation) are different from the ones presented in related work (e.g., foraging [27]). In the near term, one direction for comparison would be applying our techniques to previously used study cases and benchmark the approach.

We believe that our proposed method has a good scaling behavior. An additional robot and its constraints represent an additional variable in our coordination space. Since we are using sampling-based methods for finding a feasible solution (which have been used in large dimensions [61]), we believe that our method can scale to larger groups. Furthermore, in sampling-based motion planning, a significant part of the computational cost is collision checking, and since this is simple in our formulation (obstacles are hypercubes), there is good potential for scaling.

5. Scheduling with Re-Planning

The previous solution provides us with a coordination space and corresponding path that yields a velocity-tuning approach preventing operator collisions. We now look for a solution that yields a shorter mission runtime by also altering the robot trajectories. This solution is found by comparing the current path through the coordination space h and the desired shortest-path path h_{des} which would be a straight line. Given the example in Figure 2a,b, where we see the robots and environment, and the resulting coordination space, we indicate an "ideal" path as in Figure 2c. When searching for a path through the coordination space, we may find a point $x \in X$ such that $h_{des}(x) \bigcap X_{obs} \neq \emptyset$, representing an obstacle. In the example shown in Figure 2c, this is indicated by the blue region, meaning that the ideal path is not valid as it intersects the obstacle. In these situations, the solution is to either plan around the obstacle, corresponding to tuning the velocity of the robots involved—as in the solution for Problem 1—or creating alternative plans for the robots. In the latter case, the number of operators requested during the original set of times corresponding to the obstacle can now be fulfilled, potentially reducing the overall mission runtime if the resulting plans are shorter than the wait times.

Algorithm 2: CollisionCheck

```
Input  : Point $x$; Number of operators $p$; robots $\mathcal{A}$
Output : True if obstacle-free, False otherwise
$n_{colls} \leftarrow 0$
for $i \in [1, m]$ do
    $q \leftarrow \lambda^i(x_i)$ if $q \in \mathcal{C}^i_{att}$ then
        $n_{colls} \leftarrow n_{colls} + 1$
        if $n_{colls} \geq p$ then
            return False
        end
    end
end
return True
```

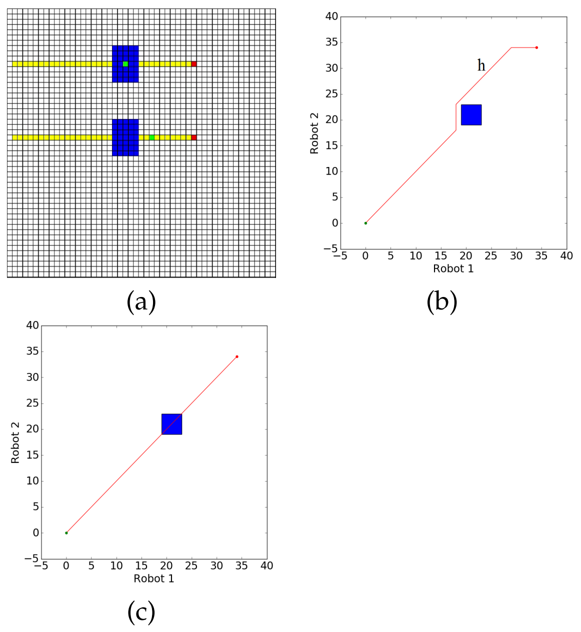


Figure 2. Example Environment and resulting Coordination Space (**a**) A planar environment with dangerous regions requiring operator supervision to traverse shown in blue, and robot trajectories in yellow. (**b**) The two-dimensional Coordination Space resulting from (**a**). Each axis corresponds to the positions of robots along with their trajectories. The red line indicates an attention-conflict-free path through the coordination space. (**c**) Coordination space from (**b**), with the desired (optimal) policy shown as the red line.

A critical side-effect to keep in mind is that by modifying robots' trajectories when avoiding collisions caused by conflicting operator attention requests, we are also potentially changing later parts of their trajectory. This change will lead to a different coordination space and the possibility of shifting, creating, or removing subsequent obstacles. As an illustrative example, Figure 3a shows two robots, which enter regions requiring supervision at the same time and produce the coordination space in Figure 3b. The vertical segment of the path h shown in red corresponds to the collision being resolved by pausing robot 1 until robot 2 has finished its operator request before continuing. This scenario could also be solved by re-planning robot 2 so that it avoids operator requests during the original times. However, robot 1 will then require more time to travel around the dangerous region, causing it to encounter its second critical section at a later time—precisely when robot 1 is entering its second request as well (Figure 3c)—creating another conflict that must be solved.

This setup yields our initial solution via velocity-tuning. Then create an ideal path h_{opt}, given by a straight line that assumes no robots require supervision (line 3). Next, we verify if the optimal solution is valid by checking for collisions between h_{des} and obstacles in the coordination space and return the first obstacle encountered—if any in line 4. *FirstObstacle* returns the robots involved in the "collision" $o_{\mathcal{A}_{inv}}$, along with the corresponding configurations $o_{\mathcal{C}_{att}}$ and times that each robot has in conflict $o_{t_{den}}$. If the ideal path is invalid (line 5), we can resolve this in two ways:

1. Alter the involved robots policies (as in the previous solution).
2. Re-plan the involved robots trajectories to eliminate the obstacle.

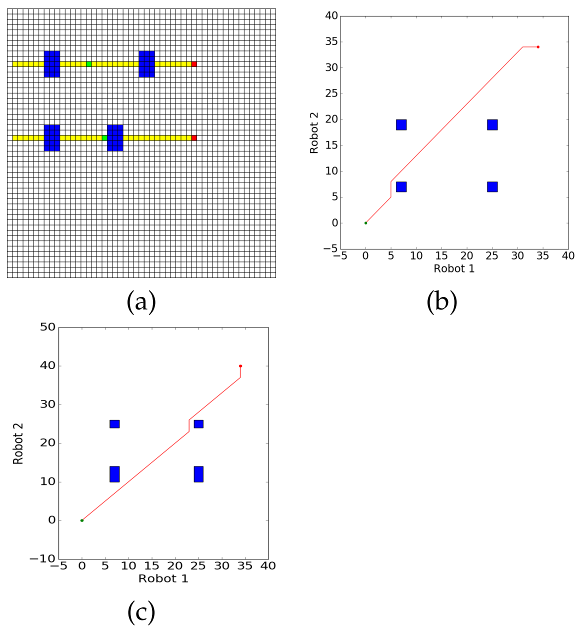


Figure 3. Example Environment, resulting Coordination Space, and Shifting Conflict Regions (**a**) Robots in their environment, and their expected trajectories; (**b**) Original Coordination Space resulting from (**a**); (**c**) Final Coordination Space after re-planning around the first attention obstacle.

We now describe how to re-plan the robot's trajectories. Given the robots involved in the collision, $o_{\mathcal{A}_{inv}}$, we sort them in order of ascending length of execution time and select the shortest $|o_{\mathcal{A}_{inv}}| - p$—the minimum number of robots to re-plan to remove the attention collision (lines 6, 7). This procedure is performed on the robots with the shortest current plans so that extensions to their plans due to re-planning should have a minimal effect on the mission's overall length. Then generate alternative trajectories for the robots, provided operator-denied times $o_{t_{den}}$, and create an alternative goal location $x_{altgoal}$ to account for any shifts in the ending times of the robot plans (lines 8–11).

Suppose the distance between x_{init} and the alternative $x_{altgoal}$ is longer than the current solution. In that case, velocity-tuning will yield a better solution, and we incorporate the obstacle into the "desired" set of obstacles (lines 12, 22). Otherwise, we test if the alternative, a re-planned solution is better (lines 13, 14). If it is, then update the robots with their re-planned trajectories, and replace the current coordination space and goal to account for any changes in execution times (lines 15–17); else we incorporate the obstacle into the "desired" set of obstacles as before (line 19).

We repeat this process of generating desired solutions (line 24) and testing them until the desired path h_{des} no longer intersects any obstacles. At this point, we return the final h_{des} that will have no operator conflicts.

6. Experimental Results

In this section, we cover the design and of both simulated and physical experiments, and the results obtained.

6.1. Software Simulation for Scheduling with Re-Planning

Here we describe our simulation and provide an example *plan* algorithm that re-plans a robot's trajectory around unsafe areas in the environment—which would require operator supervision—given operator-denied times.

The simulated environment consisted of a discretized 2-dimensional grid-world where robots can only move either horizontally or vertically. The environment also contains hazardous regions (shown in blue) which require operator supervision to traverse, corresponding to configurations in $\mathcal{C}_{att}$.

Example Re-plan Algorithm: The *plan* algorithm used in this example attempts to find the shortest path between x^i_{init} and x^i_{final} within the robot's environment, which can be easily attained via the A^* algorithm [62,63]. However, this path may intersect with regions requiring supervision. First, denote the starting time of the mission as $T_i = 0$. Given times when an operator will not be available for the robot, t_{den}, we modify A^* as follows: Augment A^*'s nodes with an additional *time* parameter. When visiting a node, update its neighbor's *time* attributes to *time* + *travel_time* where *time* is the current time, and *travel_time* is the time required to move from the current node to the neighbor. If the neighbor physically resides within $\mathcal{C}_{att}$ and the neighbors *time* is inside t^i_{den}, then we treat it as an obstacle. This modification of A^* provides paths that circumvent obstacles during operator-denied times, with an example shown in Figure 4.

Algorithm 3: Scheduler

```
Input  : $\mathcal{A}$, robots to plan
Output: $h$, path through $X$ used to derive policy
$x_{init} \leftarrow (0,\dots,0); x_{goal} \leftarrow (\tilde{t^1_f}, \tilde{t^m_f})$
$X_{curr} \leftarrow [\tilde{t^1_f},\dots,\tilde{t^m_f}]; h_{curr} \leftarrow$ B-RRT$^*(X_{curr}, x_{init}, x_{goal}, p, \mathcal{C}_{att})$
$X_{des} \leftarrow [0, \tilde{t^1_f},\dots,\tilde{t^m_f}]; h_{des} \leftarrow line(x_{init}, x_{goal}); \mathcal{C}_{desatt} \leftarrow \emptyset$
$o \leftarrow FirstObstacle(h_{des}, C_{att})$
while $o \neq \emptyset$ do
    $\mathcal{A}_{inv} \leftarrow Sort(o_{\mathcal{A}_{inv}})$
    $\mathcal{A}_{min} \leftarrow \mathcal{A}_{inv}[0 : |o_{\mathcal{A}_{inv}}| - p]$
    $\mathcal{A}_{alt} \leftarrow (\mathcal{A} \backslash \mathcal{A}_{min})$
    $plan(\mathcal{A}^i, t^i_{den}) \forall \mathcal{A}^i \in \mathcal{A}_{min}$
    $\mathcal{A}_{alt} \leftarrow \mathcal{A}_{alt} \bigcup \mathcal{A}_{min}$
    $x_{altgoal} \leftarrow (\tilde{t^1_f},\dots,\tilde{t^m_f}) \forall \mathcal{A}^i \in \mathcal{A}_{alt}$
    if $d(x_{init}, x_{altgoal}) \leq c(h_{curr})$ then
        $X_{alt} \leftarrow [0, \tilde{t^1_f}] \times \cdots \times [0, \tilde{t^m_f}]; h_{alt} \leftarrow$ B-RRT$^*(X_{alt}, x_{init}, x_{goal}, p, \mathcal{C}_{att})$
        if $c(h_{alt} \leq c(h_{curr})$ then
            $x_{goal} \leftarrow x_{altgoal}; h_{curr} \leftarrow h_{alt}$
            $X_{curr} \leftarrow X_{alt}; X_{des} \leftarrow X_{alt}$
            $\mathcal{A} \leftarrow (\mathcal{A} \backslash \mathcal{A}_{min}) \bigcup \mathcal{A}_{at}$
        else
            $\mathcal{C}_{desatt} \leftarrow \mathcal{C}_{desatt} \bigcup o_{\mathcal{C}_{att}}$
        end
    else
        $\mathcal{C}_{desatt} \leftarrow \mathcal{C}_{desatt} \bigcup o_{\mathcal{C}_{att}}$
    end
    $h_{des} \leftarrow$ B-RRT$^*(X_{des}, x_{init}, x_{goal}, p, C_{att})$
    $o \leftarrow FirstObstacle(h_{des}, C_{att})$
end
return $h_{des}$
```

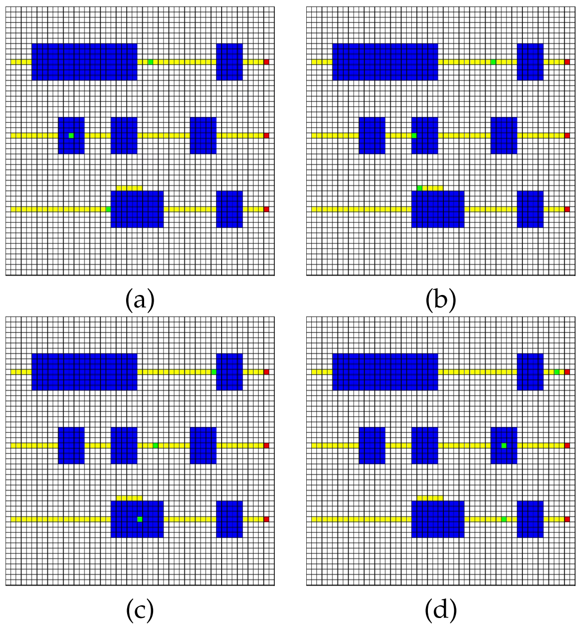


Figure 4. Example Simulation Environment Example simulation. The robots are numbered 1, 2, 3 from top to bottom. (**a**) Robot 3 stops while Robot 2 passes through its dangerous region. (**b**) Robot 3 has re-planned its trajectory and is going around the dangerous area, allowing Robot 2 to be supervised. (**c**) Robot 1 stops to allow Robot 3 enter its dangerous area with supervision. (**d**) All robots continue to their final goal locations.

In Figure 4, we show a simulated example given an environment with three robots. The blue areas in the environment are dangerous, and require operator supervision to prevent an accident. The example was designed to show several operator attention "collision" scenarios. As the robots move from left to right, the following operator requests might arise:

- $\mathcal{A}^1$ requiring an operator
- $\mathcal{A}^1$ and $\mathcal{A}^2$ require an operator at the same time
- $\mathcal{A}^1, \mathcal{A}^2, \mathcal{A}^3$ require an operator at the same time
- $\mathcal{A}^3$ requiring an operator while $\mathcal{A}^1$ and $\mathcal{A}^2$ leave their critical regions
- $\mathcal{A}^2$ requiring an operator
- $\mathcal{A}^1$ and $\mathcal{A}^2$ require an operator at the same time

The resulting coordination space is shown in Figure 5, where (a, b) is only velocity-tuning, and (c, d) is with re-planning the robot trajectories, which yields a slightly shorter mission ending time than strictly velocity-tuning.

For further validation, simulations were run using a two-dimensional environment populated with a set of randomly sized, randomly placed dangerous regions, and robots placed in randomized obstacle-free starting and goal locations along with a corresponding path between them as shown in Figure 6. Across each iteration of the simulations, environments and the starting and goal positions for the robots were randomly generated. In each generated environment, trials were run using 2, 4, or 8 robots, moving at 1 cell/second. These trials were then solved using the solutions for Problem 1 (Scheduling) and Problem 2 (Scheduling with Re-Planning), with 1, 2, 4, or 8 operators. The results can be found in Table 1.

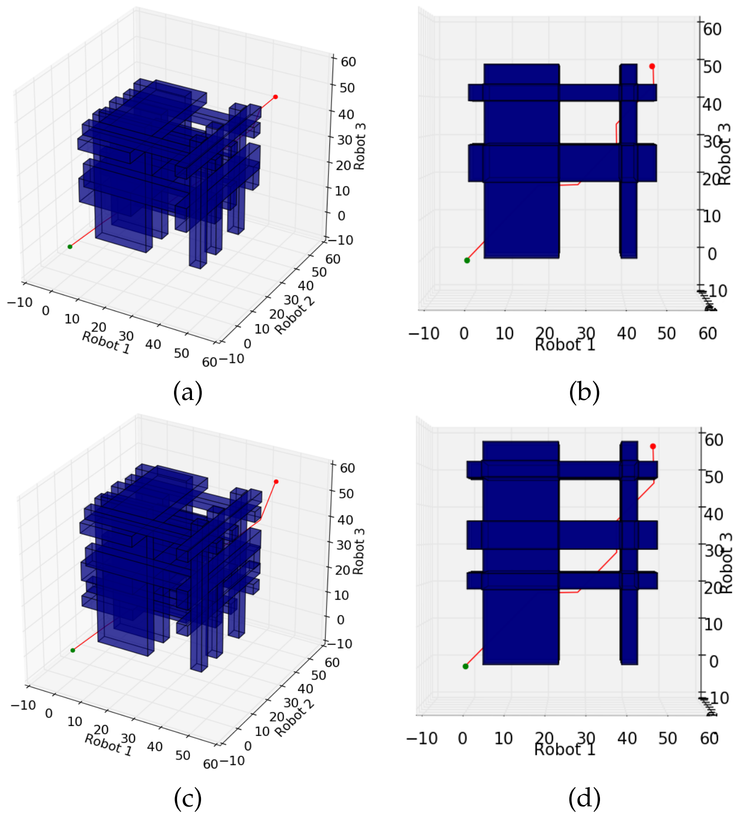


Figure 5. Example Simulation Coordination Space resulting from the example shown in Figure 4. (**a**) Original Coordination Space resulting from the environment and robots in Figure 4; (**b**) Side view of (**a**); (**c**) Final Coordination Space after replanning; (**d**) Side view of (**c**).

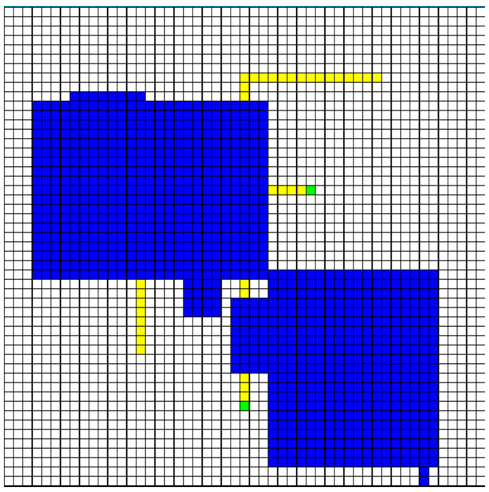


Figure 6. Example Random Environment Example of a randomly generated environment and trajectories intersecting critical regions.

Table 1. Average time savings via re-planning vs velocity-tuning.

Robots	Operators	Average Savings
2	1	1.126
2	2	0
2	4	0
2	8	0
4	1	1.937
4	2	3.402
4	4	0
4	8	0
8	1	NA
8	2	0.218
8	4	5.284
8	8	0

There is an increase in the average time saved when dealing with larger numbers of robots, as re-scheduling can simultaneously resolve multiple robots at once. We purposefully ran the simulations with equal numbers of robots and operators to ensure that there would be no time saved—as there would be no obstacles generated in the first place—and this performed as expected. All tests with two and four robots completed successfully. In trials with eight robots and a single operator, a solution was not found with the RRT^* parameters that were used. Given 2 operators, $\sim$30% completed, and $\sim$60% for four operators. This result was due to the low sample count used when running Attention RRT^*, and the large *steer* length, which prevented it from exploring paths in narrow gaps between obstacles. The tuning of the sample count, steer length and rewire count lie outside the scope of this work, but is nonetheless an interesting problem we expect to incorporate in future work.

6.2. *Hardware Experiment for Scheduling with Re-Planning*

Here, we further illustrate the problem and solution via a hardware example. This example consisted of a single operator that had to be allocated across three line-following robots in a discrete grid environment.

The robots use a deterministic finite state machine to keep track of the position and orientation, and a transition function given by a second transition-state machine that ensures the robots inter-state path does not deviate from a grid line.

The hardware experiment in Figure 7 has an equivalent simulated environment shown in Figure 7a. The robots have initial trajectories shown in yellow, which pass through dangerous areas of the environment (blue) requiring operator supervision. The physical implementation represents the dangerous areas using red/yellow squares, in the same locations as in the virtual simulation. The resulting coordination space in Figure 7b provides a set of policies enabling the robots to execute their trajectories while ensuring that the operator is not split among multiple robots at the same time. The robots then executed their corresponding policies, moving and pausing when appropriate, with at most one robot entering a dangerous region at a time. Additional experiments and videos can be found at: http://users.cis.fiu.edu/~jabobadi/oa/ (accessed on 4 April 2021).

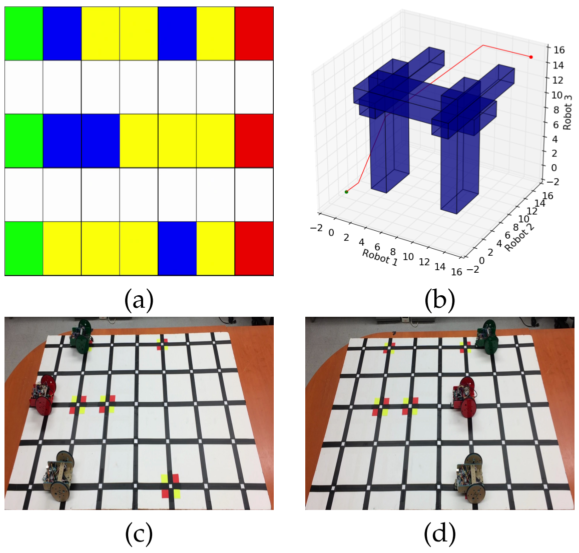


Figure 7. Hardware Experiment Example (**a**) Simulated Environment; (**b**) Coordination Space resulting from (**a**); (**c**) Analogous hardware simulation at $t = 1$; (**d**) Hardware simulation at $t = 5$.

The hardware experiments that were run and shown in the above link show successful runs using the above procedures to design trajectories and policies for three different robots under the supervision of a single operator. The mission ended in the shortest time possible, and the operator did not receive multiple concurrent requests.

7. Study Case: Humanoid Robots

In this section, an application of the proposed method to NASA's humanoid robot Valkyrie [64] as shown Figure 8, is presented. Humanoid robots are high degree of freedom complex systems that have been proposed for diverse applications including nuclear-decommissioning tasks [65,66], disaster response assistance [67], and vehicles of space exploration [64]. For many of these tasks, it is desirable to have a *human-in-the-loop* controller to ensure critical and hazardous sub-tasks are completed. The supervised autonomy frameworks to make humanoid robots applicable in performing complex tasks require an effective design for a shared operator control interface which remains an open question. As seen during the DRC, completion of complex tasks in simulated environments with humanoids requires large teams of operators and shared control is indispensable [67]. Indeed even a simple manipulation task requires coherent operator collaboration or inter-operator communication problems can have detrimental effects [9]. Thus, it is preferable to enforce a 1:1 ratio between humanoids and operator [8].

7.1. Methodology

We propose partitioning the humanoid robot into two serial kinematic chains, the left and right arm, which are denoted as $\mathcal{A}^l$ and $\mathcal{A}^r$ respectively. The desired task is modeled as a typical pick and place operation where the robots must visit designated picking and placing zones defined by the bounding boxes $\mathcal{X}_{i=1\ldots n}$. For example, $\mathcal{A}^r$ picks an object from $\mathcal{X}_1$ and places it in $\mathcal{X}_2$. Next, $\mathcal{A}^l$ collects the object from $\mathcal{X}_2$ and places it in a final location $\mathcal{X}_3$. The picking and placing actions are executed by the end effectors of the right and left arms whose positions are respectively given by $\mathbf{p}^r$ and $\mathbf{p}^l$. When an end effector (robot's hand) is within a bounding box $\mathcal{X}_{i=1\ldots n}$, it requires operator attention, i.e., the action is considered sensitive and require operator supervision. Thus, $\mathcal{X}_{i=1\ldots n}$ constitute configuration space constraints that must be transformed into critical regions in the coordination space. Thus, the constraints are represented in the configuration space as follows:

$$\lambda^l(t) \bigcap \lambda^r(t) = \emptyset | \forall t \in [0, t_f]; \lambda^l(t), \lambda^r(t) \in \mathcal{X}$$

Additionally, the re-planning algorithm is modified as follows: Given a set of waypoints τ and operator-denied times t_{den}, $plan$ will re-plan sections of λ that reside within $\mathcal{X}$ during times t_{den} if possible. If re-planning is not possible, or if there are critical waypoints that should not be altered (such as waypoints denoting pick and place actions) the waypoints and relevant sections of λ will be untouched and returned to the scheduler as is.

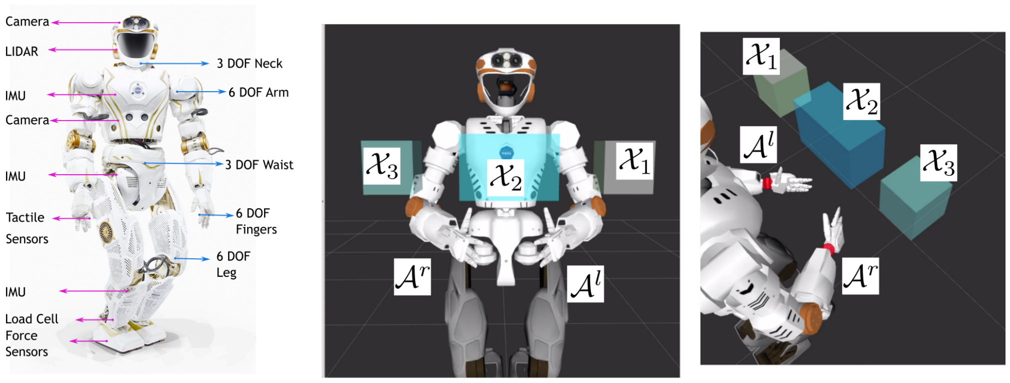
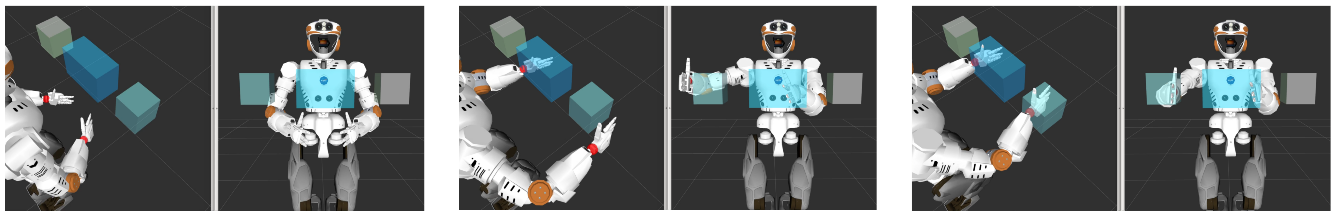


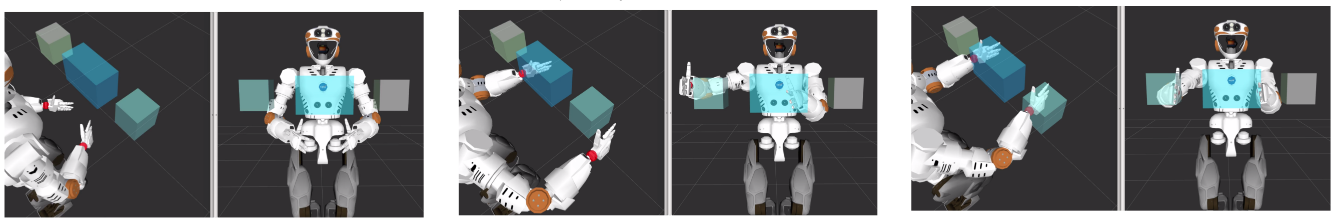
Figure 8. (**Left**) NASA's humanoid robot Valkyrie. (**Middle**, **Right**) Experimental setup showing coordination space obstacles and kinematic chains that are treated as independent robots.

7.2. Results

The simulation experiments are executed using the dynamic simulator *Gazebo*. An initial set of waypoints are defined for $\mathcal{A}^l$ and $\mathcal{A}^r$. These waypoints consist of a set of Cartesian positions and velocities for the kinematic chains such that λ^r and λ^l satisfy the pick and place task constraints. The initial waypoints are passed to the scheduling algorithm which generates a new set of waypoints that—when separated by a monotonic time step—satisfy both the configuration and coordination space constraints. A cubic interpolation of the waypoints is used to generate a continuous trajectory for execution on the robot. A comparison between the executions before and after the scheduling algorithm is shown in Figures 9 and 10. The coordination space of these trajectories is shown in Figure 11.

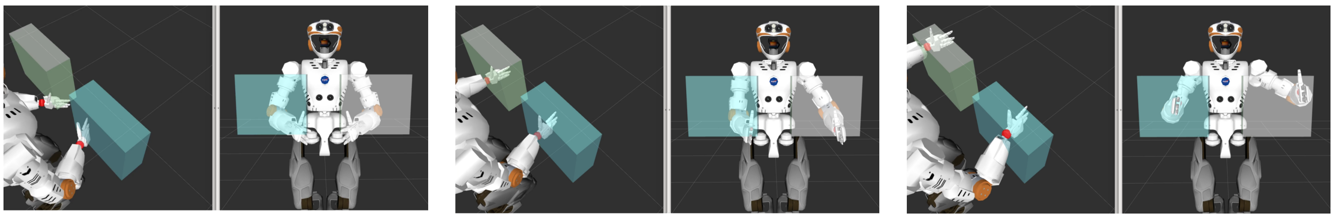


(**a**). Initial trajectory with three attention zones

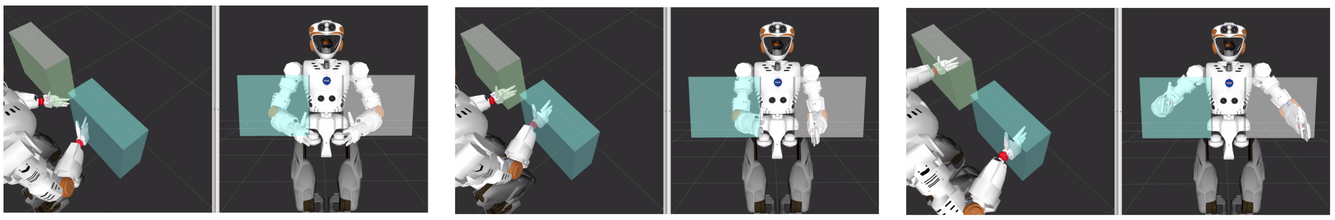


(**b**). Rescheduled and re-planned trajectory with three attention zones

Figure 9. Pick and place task with three attention obstacles. The planning reference frame is located at the wrist of the respective arms and is highlight by a red square. (**Left**): Both plans start in a valid position. Middle: Both plans approach the bounding in the same manner, but in the rescheduled case, the right arm execution is slowed down to ensure that before entering the bounding box the left hand has already left the attention zone (**Right**).

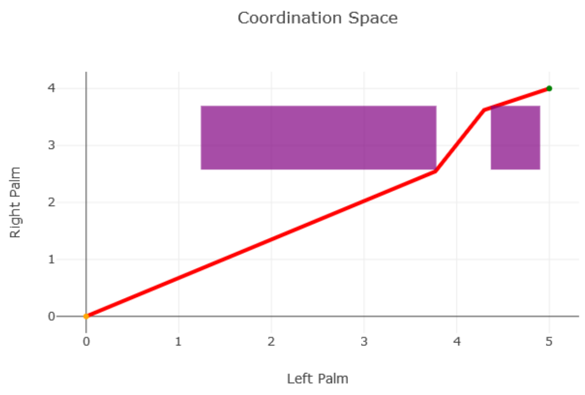


(**a**). Initial trajectory with two attention zones

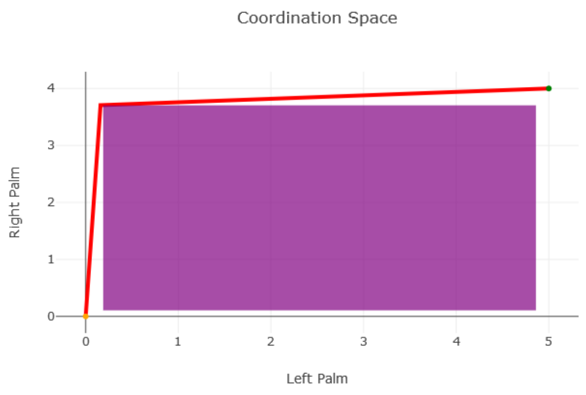


(**b**). Rescheduled and re-planned trajectory with two attention zones

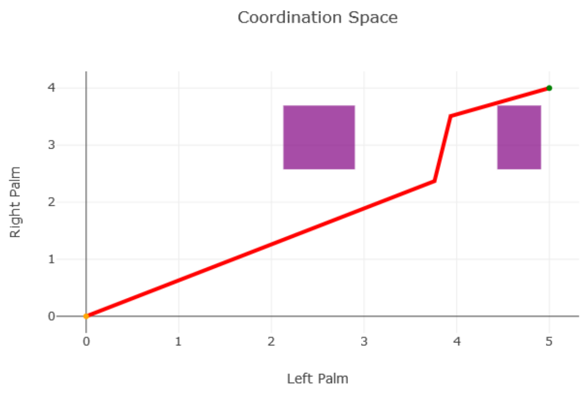
Figure 10. Pick and place task with two attention obstacles. The planning reference frame is located at the wrist of the respective arms and is highlight by a red square. (**Left**): Both plans start in a valid position. Middle: The initial trajectory immediately violates attention constraints while the rescheduled trajectory slows the left arm to prevent entry into the area. (**Right**): The right arm is slightly withdrawn (re-planning) to ensure target frame is outside the bounding box before the left has to enter.



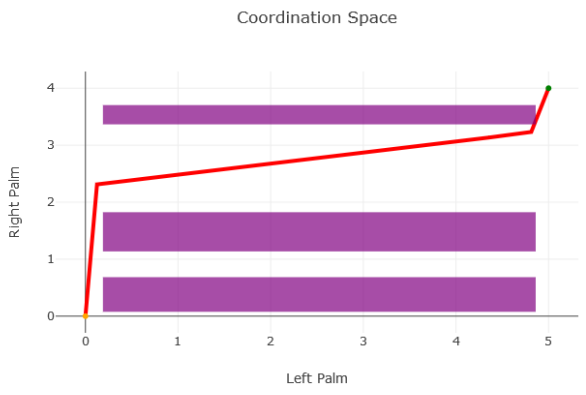
(**a**). Trajectory of Figure 9a



(**b**). Trajectory of Figure 10a



(**c**). Trajectory of Figure 9b



(**d**). Trajectory of Figure 10b

Figure 11. Purple areas represent times when both palms will be in a critical zone while the red line is the scheduled times to reach a point for each palm.

The two original trajectories shown in Figure 11a,b have conflicts in critical areas as illustrated by the line passing through purple areas. The reduced purple areas in Figure 11c,d demonstrate the re-planning of waypoints, and the altered slope of the line through space indicates a change in time through the waypoints. Both trajectories use a combination of re-planning and rescheduling to generate a collision-free path through the coordination space.

8. Conclusions and Future Work

This work provides a geometric approach for converting robot trajectories and supervision requests into a set of policies for the robots that permit operators to oversee critical sections of robot plans without being over-allocated. The provided solution is also capable of determining when re-planning robots would yield a better solution than velocity-tuning. There are exciting avenues for future work.

In the short term, we would like to look at the effects that robot movement has on an operator's effectiveness in overseeing them [68], and incorporate this effect into our solution. As an example, operators require time to switch their attention from one robot to another. This context switching time might be represented by extending obstacles in the coordination space towards the origin. Similarly, a robot's path may have some element of uncertainty, especially when outside of a factory setting. In this case, we can "inflate" the obstacles within the coordination space, which would provide a more cautious solution.

We are also interested in improving our modeling of context switching times by using constructions from Human-Robot Interaction research. Potential sources of information that can be incorporated are mental states (MS) modeling and physiological factors [30]. We believe that the combination of realistic human cognition models and algorithmic, scalable methodologies such as the one we proposed in this paper can lead to fundamental insights.

Searching through the coordination space might be modified to use a receding horizon approach to allow for more rapidly changing robot plans if presented with a dynamic environment. We would like to include the stability constraints and interdependence between kinematic chains when working with robots with large degree of freedom.

We studied the complexity of problem 1 and argued that it is NP-Hard by using the technique or restriction, and we proceeded to propose feasible heuristics to solve it. A natural direction will be to carefully study approximation algorithms [69] for scheduling problems [70] that can be translated into our framework. This can help us calculate approximation ratios and performance guarantees for our approach.

In our paper, we presented two study cases to show our approach's practical feasibility and range of applications. The first scenario is on a set of mobile robots, and the second is on a robot with several degrees of freedom. We want to continue exploring applications and extend this work to human studies to investigate the framework's effectiveness for complex teleoperation tasks. A domain of interest where our ideas can apply are one-to-many (OTM) scenarios where a human operator needs to monitor and coordinate multiple multiple autonomous vehicles [39].

Author Contributions: Conceptualization, S.A.Z., P.L., L.B. and T.P.; Methodology, S.A.Z., P.L. and L.B.; Software, P.D., S.A.Z., P.L. and L.B.; Validation, S.A.Z. and P.D.; Writing—original draft preparation, S.A.Z., P.L., L.B.; Writing—review and editing, P.L., L.B. and T.P.; Supervision, P.L., L.B. and T.P.; Project administration L.B. and T.P.; Funding acquisition, L.B. and T.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by the Department of Energy under Award Number DE-EM0004482, by the National Aeronautics and Space Administration under Grant No. NNX16AC48A issued through the Science and Technology Mission Directorate and by the National Science Foundation under Award No. 1451427. This research is also supported in part by the National Science Foundation through awards IIS-2034123 and IIS-2024733 and by the U.S. Department of Homeland Security under Grant Award Numbers 2017-ST-062000002.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviation is used in this manuscript:

RRT	Rapidly Exploring Random Tree

References

1. Alam, T.; Bobadilla, L. Multi-Robot Coverage and Persistent Monitoring in Sensing-Constrained Environments. *Robotics* **2020**, *9*, 47. [CrossRef]
2. Bandala, M.; West, C.; Monk, S.; Montazeri, A.; Taylor, C.J. Vision-based assisted tele-operation of a dual-arm hydraulically actuated robot for pipe cutting and grasping in nuclear environments. *Robotics* **2019**, *8*, 42. [CrossRef]
3. Murphy, R.R.; Gandudi, V.B.M.; Adams, J. Applications of robots for Covid-19 response. *arXiv* **2020**, arXiv:2008.06976.
4. Shah, J.; Wiken, J.; Williams, B.; Breazeal, C. Improved human-robot team performance using chaski, a human-inspired plan execution system. In Proceedings of the 6th International Conference on Human-Robot Interaction, Lausanne, Switzerland, 6–9 March 2011; ACM: New York, NY, USA, 2011; pp. 29–36.
5. Wilcox, R.; Nikolaidis, S.; Shah, J. Optimization of temporal dynamics for adaptive human-robot interaction in assembly manufacturing. *Robot. Sci. Syst. VIII* **2012**, 441–448. [CrossRef]
6. Helms, E.; Schraft, R.D.; Hagele, M. rob@ work: Robot assistant in industrial environments. In Proceedings of the IEEE International Workshop on Robot and Human Interactive Communication, Berlin, Germany, 27–27 September 2002; IEEE: Piscataway, NJ, USA, 2002; pp. 399–404.
7. Murphy, R.R. Human-robot interaction in rescue robotics. *IEEE Trans. Syst. Man Cybern. Part Appl. Rev.* **2004**, *34*, 138–153. [CrossRef]
8. Yanco, H.A.; Norton, A.; Ober, W.; Shane, D.; Skinner, A.; Vice, J. Analysis of human-robot interaction at the darpa robotics challenge trials. *J. Field Robot.* **2015**, *32*, 420–444. [CrossRef]
9. Atkeson, C.G.; Benzun, P.B.; Banerjee, N.; Berenson, D.; Bove, C.P.; Cui, X.; De Donato, M.; Du, R.; Feng, S.; Franklin, P.; et al. What happened at the DARPA robotics challenge finals. In *The DARPA Robotics Challenge Finals: Humanoid Robots to the Rescue*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 667–684.
10. Zanlongo, S.A.; Rahman, M.; Abodo, F.; Bobadilla, L. Multi-robot Planning for Non-overlapping Operator Attention Allocation. In Proceedings of the IEEE International Conference on Robotic Computing, Taichung, Taiwan, 10–12 April 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 109–112.
11. Zanlongo, S.; Abodo, F.; Long, P.; Padir, T.; Bobadilla, L. Multi-Robot Scheduling and Path-Planning for Non-Overlapping Operator Attention. In Proceedings of the 2018 Second IEEE International Conference on Robotic Computing (IRC), Laguna Hills, CA, USA, 31 January–2 February 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 87–94.
12. Sandoval Arévalo, J.S.; Laribi, M.A.; Zeghloul, S.; Arsicault, M. On the design of a safe human-friendly teleoperated system for doppler sonography. *Robotics* **2019**, *8*, 29. [CrossRef]
13. Lasota, P.A.; Fong, T.; Shah, J.A. *A Survey of Methods for Safe Human-Robot Interaction*; Now Publishers: Delft, The Netherlands, 2017.
14. Sun, D.; Liao, Q.; Loutfi, A. Single master bimanual teleoperation system with efficient regulation. *IEEE Trans. Robot.* **2020**, *36*, 1022–1037. [CrossRef]
15. Li, Y.; Liu, K.; He, W.; Yin, Y.; Johansson, R.; Zhang, K. Bilateral teleoperation of multiple robots under scheduling communication. *IEEE Trans. Control Syst. Technol.* **2019**, *28*, 1770–1784. [CrossRef]
16. Rosenberg, L.B. *The Use of Virtual Fixtures as Perceptual Overlays to Enhance Operator Performance in Remote Environments*; Technical Report; Stanford University Center for Design Research: Stanford, CA, USA, 1992.
17. Bowyer, S.A.; Davies, B.L.; y Baena, F.R. Active constraints/virtual fixtures: A survey. *IEEE Trans. Robot.* **2014**, *30*, 138–157. [CrossRef]
18. Yamamoto, T.; Abolhassani, N.; Jung, S.; Okamura, A.M.; Judkins, T.N. Augmented reality and haptic interfaces for robot-assisted surgery. *Int. J. Med. Robot. Comput. Assist. Surg.* **2012**, *8*, 45–56. [CrossRef]
19. Bettini, A.; Marayong, P.; Lang, S.; Okamura, A.M.; Hager, G.D. Vision-assisted control for manipulation using virtual fixtures. *IEEE Trans. Robot.* **2004**, *20*, 953–966. [CrossRef]
20. Quintero, C.P.; Dehghan, M.; Ramirez, O.; Ang, M.H.; Jagersand, M. Flexible virtual fixture interface for path specification in tele-manipulation. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 5363–5368.
21. Pruks, V.; Farkhatdinov, I.; Ryu, J.H. Preliminary study on real-time interactive virtual fixture generation method for shared teleoperation in unstructured environments. In Proceedings of the International Conference on Human Haptic Sensing and Touch Enabled Computer Applications, Pisa, Italy, 13–16 June 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 648–659.

22. Long, P.; Keleştemur, T.; Önol, A.Ö.; Padir, T. Optimization-Based Human-in-the-Loop Manipulation Using Joint Space Polytopes. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 204–210.
23. Farkhatdinov, I.; Ryu, J.H. Teleoperation of multi-robot and multi-property systems. In Proceedings of the 2008 6th IEEE International Conference on Industrial Informatics, Daejeon, Korea, 13–16 July 2008; IEEE: Piscataway, NJ, USA, 2008; pp. 1453–1458.
24. Roldán, J.J.; Díaz-Maroto, V.; Real, J.; Palafox, P.R.; Valente, J.; Garzón, M.; Barrientos, A. Press start to play: Classifying multi-robot operators and predicting their strategies through a videogame. *Robotics* **2019**, *8*, 53. [CrossRef]
25. Luo, J.; Lin, Z.; Li, Y.; Yang, C. A teleoperation framework for mobile robots based on shared control. *IEEE Robot. Autom. Lett.* **2019**, *5*, 377–384. [CrossRef]
26. Hughes, T. Human-Automation Coordination in Multi-UAV Control. In Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit, Honolulu, HI, USA, 18–21 August 2008; p. 6315.
27. Chien, S.Y.; Lewis, M.; Mehrotra, S.; Brooks, N.; Sycara, K. Scheduling operator attention for multi-robot control. In Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura, Algarve, 7–12 October 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 473–479.
28. Dardona, T.; Eslamian, S.; Reisner, L.A.; Pandya, A. Remote presence: Development and usability evaluation of a head-mounted display for camera control on the da Vinci Surgical System. *Robotics* **2019**, *8*, 31. [CrossRef]
29. Wong, C.Y.; Seet, G. Workload, awareness and automation in multiple-robot supervision. *Int. J. Adv. Robot. Syst.* **2017**, *14*, 1729881417710463. [CrossRef]
30. Roy, R.N.; Drougard, N.; Gateau, T.; Dehais, F.; Chanel, C.P. How Can Physiological Computing Benefit Human-Robot Interaction? *Robotics* **2020**, *9*, 100. [CrossRef]
31. Dybvik, H.; Løland, M.; Gerstenberg, A.; Slåttsveen, K.B.; Steinert, M. A low-cost predictive display for teleoperation: Investigating effects on human performance and workload. *Int. J. Hum. Comput. Stud.* **2021**, *145*, 102536. [CrossRef]
32. Lu, S.; Zhang, M.Y.; Ersal, T.; Yang, X.J. Workload management in teleoperation of unmanned ground vehicles: Effects of a delay compensation aid on human operators' workload and teleoperation performance. *Int. J. Hum. Comput. Interact.* **2019**, *35*, 1820–1830. [CrossRef]
33. Riley, J.M.; Endsley, M.R. Situation awareness in HRI with collaborating remotely piloted vehicles. In Proceedings of the Human Factors and Ergonomics Society Annual Meeting, Orlando, FL, USA, 26–30 September 2005; SAGE Publications: Los Angeles, CA, USA, 2005; Volume 49, pp. 407–411.
34. Adams, J.A. Multiple robot/single human interaction: Effects on perceived workload. *Behav. Inf. Technol.* **2009**, *28*, 183–198. [CrossRef]
35. Al-Hussaini, S.; Gregory, J.M.; Guan, Y.; Gupta, S.K. Generating Alerts to Assist With Task Assignments in Human-Supervised Multi-Robot Teams Operating in Challenging Environments. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Las Vegas, NV, USA, 24–30 October 2020.
36. Al-Hussaini, S.; Gregory, J.M.; Shriyam, S.; Gupta, S.K. An Alert-Generation Framework for Improving Resiliency in Human-Supervised, Multi-Agent Teams. *arXiv* **2019**, arXiv:1909.06480.
37. Velagapudi, P.; Scerri, P.; Sycara, K.; Wang, H.; Lewis, M.; Wang, J. Scaling effects in multi-robot control. In Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France, 22–26 September 2008; IEEE: Piscataway, NJ, USA, 2008; pp. 2121–2126.
38. Prewett, M.S.; Johnson, R.C.; Saboe, K.N.; Elliott, L.R.; Coovert, M.D. Managing workload in human–robot interaction: A review of empirical studies. *Comput. Hum. Behav.* **2010**, *26*, 840–856. [CrossRef]
39. Lim, Y.; Pongsarkornsathien, N.; Gardi, A.; Sabatini, R.; Kistan, T.; Ezer, N.; Bursch, D.J. Adaptive Human-Robot Interactions for Multiple Unmanned Aerial Vehicles. *Robotics* **2021**, *10*, 12. [CrossRef]
40. Parker, L. Multiple mobile robot systems. In *Springer Handbook of Robotics*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 921–941.
41. Ponda, S.; Johnson, L.; Geramifard, A.; How, J. Cooperative mission planning for Multi-UAV teams. In *Handbook of Unmanned Aerial Vehicles*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 1447–1490.
42. Hauser, K. Minimum Constraint Displacement Motion Planning. *Robot. Sci. Syst.* **2013**, *6*, 2.
43. LaValle, S.; Hutchinson, S. Optimal Motion Planning for Multiple Robots Having Independent Goals. In Proceedings of the IEEE International Conference on Robotics and Automation, Minneapolis, MN, USA, 22–28 April 1996; pp. 2847–2852.
44. LaValle, S.; Hutchinson, S. Optimal Motion Planning for Multiple Robots Having Independent Goals. *IEEE Trans. Robot. Autom.* **1998**, *14*, 912–925. [CrossRef]
45. Wang, J.; Zhang, Y.; Geng, L.; Fuh, J.; Teo, S. Mission planning for heterogeneous tasks with heterogeneous UAVs. In Proceedings of the International Conference on Control Automation Robotics & Vision (ICARCV), Singapore, 10–12 December 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 1484–1489.
46. Crandall, J.; Cummings, M.; Della Penna, M.; de Jong, P.M. Computing the effects of operator attention allocation in human control of multiple robots. *IEEE Trans. Syst. Man Cybern. Part Syst. Hum.* **2011**, *41*, 385–397. [CrossRef]
47. Cummings, M.; Mitchell, P. Operator scheduling strategies in supervisory control of multiple UAVs. *Aerosp. Sci. Technol.* **2007**, *11*, 339–348. [CrossRef]

48. Murphy, R.; Shields, J. *The Role of Autonomy in DoD Systems*; Technical Report; Department of Defense, Defense Science Board Task Force Report: Washington, DC, USA, 2012.
49. Ramchurn, S.; Fischer, J.; Ikuno, Y.; Wu, F.; Flann, J.; Waldock, A. A study of human-agent collaboration for multi-UAV task allocation in dynamic environments. In Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI), Buenos Aires, Argentina, 25–31 July 2015.
50. Trautman, P. Probabilistic tools for human-robot cooperation. In Proceedings of the Human Agent Robot Teamwork Workshop HRI, Boston, MA, USA, 5–8 March 2012.
51. Trautman, P.; Ma, J.; Murray, R.M.; Krause, A. Robot navigation in dense human crowds: Statistical models and experimental studies of human—Robot cooperation. *Int. J. Robot. Res.* **2015**, *34*, 335–356. [CrossRef]
52. Rahman, M.M.; Bobadilla, L.; Mostafavi, A.; Carmenate, T.; Zanlongo, S.A. An Automated Methodology for Worker Path Generation and Safety Assessment in Construction Projects. *IEEE Trans. Autom. Sci. Eng.* **2018**, *15*, 479–491. [CrossRef]
53. Rahman, M.M.; Carmenate, T.; Bobadilla, L.; Zanlongo, S.; Mostafavi, A. A coupled discrete-event and motion planning methodology for automated safety assessment in construction projects. In Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 26–30 May 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 3849–3855.
54. Rahman, M.M.; Carmenate, T.; Bobadilla, L.; Mostafavi, A. Ex-ante assessment of struck-by safety hazards in construction projects: A motion-planning approach. In Proceedings of the 2014 IEEE International Conference on Automation Science and Engineering (CASE), Taipei, Taiwan, 18–22 August 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 277–282.
55. Gary, M.R.; Johnson, D.S. Computers and Intractability: A Guide to the Theory of NP-completeness. *J. Symb. Log.* **1979**, *48*, 498–500.
56. LaValle, S. *Planning Algorithms*; Cambridge University Press: Cambridge, UK, 2006.
57. Karaman, S.; Frazzoli, E. Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res.* **2011**, *30*, 846–894. [CrossRef]
58. Karaman, S.; Frazzoli, E. Incremental sampling-based algorithms for optimal motion planning. *Robot. Sci. Syst. VI* **2010**, *104*, [CrossRef]
59. Jordan, M.; Perez, A. *Optimal Bidirectional Rapidly-Exploring Random Trees*; Technical Report; Computer Science and Artificial Intelligence Laboratory: Cambridge, MA, USA, 2003.
60. Qureshi, A.H.; Ayaz, Y. Intelligent bidirectional rapidly-exploring random trees for optimal motion planning in complex cluttered environments. *Robot. Auton. Syst.* **2015**, *68*, 1–11. [CrossRef]
61. Shkolnik, A.; Tedrake, R. Path planning in 1000+ dimensions using a task-space Voronoi bias. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 2061–2067.
62. Beardwood, J.; Halton, J.; Hammersley, J. The shortest path through many points. In *Mathematical Proceedings of the Cambridge Philosophical Society*; Cambridge University Press: Cambridge, UK, 1959; Volume 55, pp. 299–327.
63. Matthews, J. Basic A* pathfinding made simple. *AI Game Programming Wisdom, Section 3*; Charles River Media: Hingham, MA, USA, 2002; pp. 105–113.
64. Radford, N.A.; Strawser, P.; Hambuchen, K.; Mehling, J.S.; Verdeyen, W.K.; Donnan, A.S.; Holley, J.; Sanchez, J.; Nguyen, V.; Bridgwater, L.; et al. Valkyrie: Nasa's first bipedal humanoid robot. *J. Field Robot.* **2015**, *32*, 397–419. [CrossRef]
65. Long, P.; Padir, T. Constrained Manipulability for Humanoid Robots Using Velocity Polytopes. *Int. J. Humanoid Robot.* **2020**, *17*, 1950037. [CrossRef]
66. Long, P.; Padir, T. Evaluating robot manipulability in constrained environments by velocity polytope reduction. In Proceedings of the 2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids), Beijing, China, 6–9 November 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–9.
67. DeDonato, M.; Dimitrov, V.; Du, R.; Giovacchini, R.; Knoedler, K.; Long, X.; Polido, F.; Gennert, M.A.; Padır, T.; Feng, S.; et al. Human-in-the-loop Control of a Humanoid Robot for Disaster Response: A Report from the DARPA Robotics Challenge Trials. *J. Field Robot.* **2015**, *32*, 275–292. [CrossRef]
68. Koppenborg, M.; Nickel, P.; Naber, B.; Lungfiel, A.; Huelke, M. Effects of movement speed and predictability in human–robot collaboration. *Hum. Factors Ergon. Manuf. Serv. Ind.* **2017**, *27*, 197–209. [CrossRef]
69. Vazirani, V.V. *Approximation Algorithms*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013.
70. Pinedo, M. *Scheduling*; Springer: Berlin/Heidelberg, Germany, 2012; Volume 29.