Coordinating the Motions of Multiple Robots with Specified Trajectories

Srinivas Akella
Department of Computer Science
Rensselaer Polytechnic Institute
Troy, NY 12180
sakella@cs.rpi.edu

Seth Hutchinson
Beckman Institute
University of Illinois, Urbana-Champaign
Urbana, IL 61801
seth@uiuc.edu

Abstract

Coordinating the motions of multiple robots operating in a shared workspace without collisions is an important capability. We address the task of coordinating the motions of multiple robots when their trajectories (defined by both the path and velocity along the path) are specified. This problem of collision-free trajectory coordination arises in welding and painting workcells in the automotive industry. We identify sufficient and necessary conditions for collision-free coordination of the robots when only the robot start times can be varied, and define corresponding optimization problems. We develop mixed integer programming formulations of these problems to automatically generate minimum time solutions. This method is applicable to both mobile robots and articulated arms, and places no restrictions on the number of degrees of freedom of the robots. The primary advantage of this method is its ability to coordinate the motions of several robots, with as many as 20 robots being considered. We show that, even when the robot trajectories are specified, minimum time coordination of multiple robots is NP-hard.

1 Introduction

Coordinating the motions of multiple robots without collisions as they perform a task in a shared workspace is an important capability. We focus on coordinating the motions of multiple robots constrained to follow specified trajectories. By *trajectory*, we mean both the geometric specification of the path and the velocity at which the robot traverses the path. We outline this *trajectory coordination* problem and define corresponding optimization problems where the goal is to find the minimum-time collision-free robot coordinations when only the robot start times can be changed.

There are a number of applications in which this trajectory coordination task is the exact problem to be solved. Consider scheduling the motions of multiple robots in a welding, spray painting, or assembly workcell to minimize the cycle time. Since the robots have overlapping workspaces, we must coordinate their motions to avoid collisions between robots. We assume that the given trajectory of each individual robot should not be modified since it may take into account collisions with stationary obstacles, have a desired velocity profile, or have desired wait times at critical points. Alternative approaches to minimizing the completion time, such as velocity tuning of the robots, may be inappropriate; for example, a painting robot must follow a given trajectory to spray paint uniformly.

We identify sufficient and necessary conditions for collision-free coordination of multiple robots and formulate the task as an optimization problem using a mixed integer programming formulation that can be solved using commercial solvers. We use collision detection software to identify potential collision conditions. The primary advantage of this method is its ability to handle many robots, each with several degrees of freedom. We place no restrictions on the number of degrees of freedom of the robots. This approach also applies to mobile robots and Automated Guided Vehicles (AGVs) moving along fixed paths with specified trajectories, and can also incorporate the motions of manipulator arms on mobile robots.

The paper is organized as follows. Section 2 briefly discusses related work. Section 3 defines the problem, formulates a set of optimization problems, and describes sufficient conditions for collision-free motion of multiple robots. Section 4 presents a mixed integer programming formulation for coordinating the motions of multiple robots with specified trajectories. Section 5 discusses necessary conditions for collision-free motion and describes a follow-the-leader strategy. Section 6 describes useful extensions to the basic problem. Section 7 discusses the computational complexity of the coordination problem. Section 8 describes our preliminary implementation of the planner and experimental results. Section 9 outlines directions for future work.

2 Related Work

Motion planning for multiple robots is a broad research area (see [11] for an overview). In the most general case, the problem is to have each robot move from its initial to its goal configuration, while avoiding collisions with static obstacles or with other robots. This problem is highly underconstrained, and very few researchers have attempted to deal with it directly. Hopcroft, Schwartz, and Sharir [7] showed that even a simplified two-dimensional case of the problem is PSPACE-hard.

A slightly more constrained version of the problem is obtained when all but one of the robots have specified trajectories. This is essentially the problem of planning a path for a single robot among moving obstacles, which has been treated by Reif and Sharir [17] and Kant and Zucker [9]. One can generalize this problem to obtain a heuristic solution to the problem of planning the motions of multiple robots. Erdmann and Lozano-Perez [3] assign priorities to robots and sequentially search for collision-free paths for the robots, in order of priority, in the configuration-time space. At each iteration, previous robots are treated as moving obstacles.

If the problem is further constrained so that the paths of the robots are specified, one obtains a path coordination problem. O'Donnell and Lozano-Perez [16] developed a method for path coordination of two robots. LaValle and Hutchinson also addressed a similar problem in [12], where each robot was constrained to remain on a specified configuration space roadmap during its motion. The work most closely related to ours is that of Leroy, Laumond, and Simeon [14]. They perform path coordination for over a hundred robots. However the size of the largest subset of robots with intersecting paths is 10.

In this paper, we address an even more constrained version of the multiple robot motion planning problem: the trajectory coordination problem where the trajectory of each robot, including the time derivatives along the path, is specified. Previous work on trajectory coordination has focused almost exclusively on dual robot systems (Bien and Lee [1], Chang, Chung and Lee [2]). Shin and Zheng [19] show that for a two-robot system, generating time-optimal trajectories for each robot independently and then delaying the start time of one of the robots leads to a minimal finish time provided the collision region satisfies a strong connectivity assumption. (A sufficient condition for this assumption is that the robots may collide only once during their motion.)

The trajectory coordination problem for multiple robots is closely related to jobshop scheduling problems (Garey, Johnson, and Sethi [5], Lawler et al. [13]). Here space is the common resource, and there are additional trajectory constraints. We model coordination of robots with fixed trajectories as no-wait jobshop problems (Sahni and Cho [18], Goyal and Sriskandarajah [6]).

3 Problem Formulation

The general problem that we are trying to solve can be expressed as an optimization problem: *Given a set of robots with specified paths and velocity profiles on those paths, find a set of parameterizations for these paths such that the total execution time for the ensemble of robots is minimized, the velocity constraints on the paths are satisfied, and no collisions occur.*

To make this problem more precise, we first turn to a brief review of paths and their parameterizations (Section 3.1). This will lead to a precise and straightforward characterization of the set of parameterizations under which the robots' velocity profiles remain invariant. We then develop a characterization for collisions that can occur between robots (Section 3.2), and a set of sufficient conditions for collision-free coordination of the robots (Section 3.4).

3.1 Trajectories and Their Parameterizations

We denote the i^{th} robot by $\mathcal{A}_i$, a configuration space by $\mathcal{C}$, and a configuration by $\mathbf{q} \in \mathcal{C}$. By *path* we mean the geometric specification of a curve in configuration space

$$\gamma : \zeta \in [0,1] \mapsto \gamma(\zeta) = \mathbf{q} \in \mathcal{C}$$

A differentiable function τ given by

$$\tau : t \in [0,T] \mapsto \tau(t) = \zeta \in [0,1]$$

with $\tau(0) = 0$ and $\tau(T) = 1$ is a reparameterization of the path γ. For our problem, t is a time variable, and T is some constant such that all robots will have completed their tasks prior to time T. A path together with a parameterization defines a *trajectory*. By trajectory we mean a path with the velocity of the robot specified at every point along the path. We will often simplify notation, and denote a trajectory as $\gamma(t)$ rather than explicitly representing the parameterization.

For our problem, robot velocities are specified a priori. One way to do this is to specify an original parameterization for γ, say τ, such that the time derivatives of τ provide the desired velocity profile. Thus, any reparameterization, say τ', that gives the desired velocity profile will be such that, for any ζ value along the path, the time derivatives of τ' and τ agree. It is easy to show that all such reparameterizations are obtained by merely changing the start time of task execution.

Without loss of generality, we will consider only the case where the start times for the robots are delayed, i.e.,

$$\tau_i'(t) = \begin{cases} \tau_i(t - t_i^{start}) & : \quad t \geq t_i^{start} \\ 0 & : \quad t < t_i^{start} \end{cases}, \qquad (1)$$

in which $t_i^{start} \geq 0$ is the time at which robot $\mathcal{A}_i$ begins its motion, and τ_i is the originally specified parameterization. Note that this equation also implies that $\mathcal{A}_i$ remains motionless until t_i^{start}. This restriction on possible reparameterizations leads to the following optimization problem.

Optimization Problem I: *Given a set of robots with specified trajectories, find the starting times for the robots such that the total execution time for the ensemble of robots is minimized and no collisions occur.*

We now turn our attention to a set of sufficient conditions for collision-free motion for this optimization problem. As will be seen in Section 4, these sufficient conditions lead to an optimization problem that can be solved using mixed integer linear programming.

3.2 Collision Zones: Geometry

Here we develop the representation for the relevant interactions between robots, using the above terminology for an individual robot moving on a path with a specified velocity profile.

We first develop notation to represent the set of points at which the i^{th} robot, $\mathcal{A}_i$, could possibly collide with the j^{th} robot, $\mathcal{A}_j$. For a specific value of ζ_i, the subset of the workspace that is occupied by the i^{th} robot is denoted by $\mathcal{A}_i(\gamma_i(\zeta_i))$. A collision between two robots corresponds to the situation in which $\mathcal{A}_i(\gamma_i(\zeta_i)) \cap \mathcal{A}_j(\gamma_j(\zeta_j)) \neq \emptyset$. For the i^{th} robot, we denote by $\mathcal{PB}_{ij}$ the set of values of ζ_i such that when robot $\mathcal{A}_i$ is at configuration $\gamma_i(\zeta_i)$ there exists a configuration of another robot, $\mathcal{A}_j$, such that the two robots collide:

$$\mathcal{PB}_{ij} = \{\zeta_i \mid \exists\ \zeta_j \in [0,1]\ s.t.\ \mathcal{A}_i(\gamma_i(\zeta_i)) \cap \mathcal{A}_j(\gamma_j(\zeta_j)) \neq \emptyset\}$$

In other words, $\mathcal{PB}_{ij}$ is the set of all points on the path of robot $\mathcal{A}_i$ at which $\mathcal{A}_i$ could collide with $\mathcal{A}_j$. (Our choice of the notation $\mathcal{PB}_{ij}$ derives from the usual convention of using the notation $\mathcal{CB}$ to denote points in the configuration space at which collisions occur.)

The set $\mathcal{PB}_{ij}$ can be represented as a set of intervals

$$\mathcal{PB}_{ij} = \{[\zeta_{is}^1, \zeta_{if}^1], \ldots, [\zeta_{is}^m, \zeta_{if}^m]\} \tag{2}$$

where each interval is a *collision zone*, and the subscripts s and f refer to the start and finish of the kth collision, indexed by the superscript k, and m denotes the number of collision zones for the robot $\mathcal{A}_i$ with $\mathcal{A}_j$. There is a natural correspondence between the collision zones of $\mathcal{PB}_{ij}$ and the collision zones of $\mathcal{PB}_{ji}$. In particular, for each collision zone in $\mathcal{PB}_{ij}$ there is at least one collision zone in $\mathcal{PB}_{ji}$ that could result in collision of the two robots. We will refer to these corresponding pairs of collision zones as *collision zone pairs*, denoted by $\mathcal{PI}_{ij}$. The set of collision zone pairs can be represented by a set of pairs of intervals:

$$\mathcal{PI}_{ij} = \{< [\zeta_{is}^k, \zeta_{if}^k], [\zeta_{js}^k, \zeta_{jf}^k] >\}. \tag{3}$$

Note that the superscript k serves to index the set of collision zone pairs. As we show in Section 3.4, it is straightforward to use $\mathcal{PI}_{ij}$ to establish a set of sufficient conditions for collision free scheduling of the robots. Note that $\mathcal{PI}_{ij}$ and $\mathcal{PI}_{ji}$ contain equivalent information.

Conceptually, collision zone pairs are generated by computing the volume swept by each robot and determining where it intersects the volume swept by another robot. The intersection regions of the swept volumes of pairs of robots give the collision zone pairs. Figure 1 is an example of two translating robots with specified trajectories that overlap in two collision zones. For this example $\mathcal{PB}_{12} = \{[a_1, a_2], [a_3, a_4]\}$ and $\mathcal{PB}_{21} = \{[b_1, b_2], [b_3, b_4]\}$. Collisions can occur only when $\zeta_1 \in [a_1, a_2]$ and $\zeta_2 \in [b_1, b_2]$ or when $\zeta_1 \in [a_3, a_4]$ and $\zeta_2 \in [b_3, b_4]$. Thus, $\mathcal{PI}_{12} = \{< [a_1, a_2], [b_1, b_2] >, < [a_3, a_4], [b_3, b_4] >\}$.

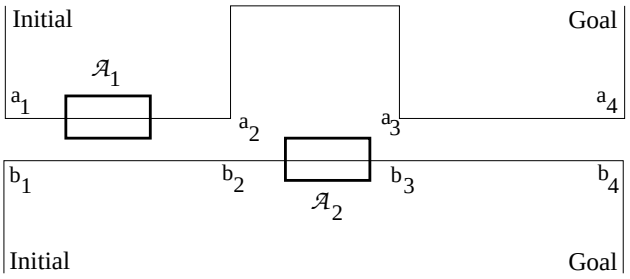

Figure 1: Example with two translating robots.

3.3 Collision Zones: Timing

The collision zone pairs describe the geometry of possible collisions, but for scheduling the robots, we are interested in the timing of the collisions. Thus, it is useful to develop a corresponding representation for the times at which two robots might collide. For a specified parameterization, τ_i, the set of times at which it is possible that robot $\mathcal{A}_i$ could collide with robot $\mathcal{A}_j$ is given by:

$$\begin{aligned} \mathcal{TB}_{ij}(\tau_i) &= \{t \mid \mathcal{A}_i(\gamma_i(\tau_i(t))) \cap \mathcal{A}_j(\gamma_j(\zeta_j)) \neq \emptyset, \\ &\qquad \text{for some } \zeta_j \in [0,1], i \neq j\} \\ &= \tau_i^{-1}(\mathcal{PB}_{ij}). \end{aligned}$$

As with $\mathcal{PB}_{ij}$, the set $\mathcal{TB}_{ij}(\tau_i)$ can be represented by a set of intervals, indexed by superscript k, the endpoints of which are obtained by applying the inverse parameterization (i.e., τ_i^{-1}) to the endpoints of the intervals of $\mathcal{PB}_{ij}$ given in (2):

$$\mathcal{TB}_{ij}(\tau_i) = \{[\tau_i^{-1}(\zeta_{is}^k), \tau_i^{-1}(\zeta_{if}^k)]\} \tag{4}$$

We refer to each interval as a *collision-time interval*.

As with collision zones, there is a natural correspondence between collision-time intervals in $\mathcal{TB}_{ij}$ and $\mathcal{TB}_{ji}$, and we refer to these pairs as *collision-time interval pairs*. For the two robots, $\mathcal{A}_i$ and $\mathcal{A}_j$, we denote the set of all collision-time interval pairs by $\mathcal{CI}_{ij}$. We represent $\mathcal{CI}_{ij}$ as a set of pairs of intervals

$$\mathcal{CI}_{ij} = \{< I_i^1, I_j^1 >, \ldots, < I_i^n, I_j^n >\}, \tag{5}$$

where the first interval I_i^k of each pair $< I_i^k, I_j^k >$ corresponds to robot $\mathcal{A}_i$ and the second interval I_j^k corresponds to robot $\mathcal{A}_j$. During the time interval I_i^k, $\mathcal{A}_i$ is in a specific collision zone and $\mathcal{A}_j$ is in a corresponding collision zone during time interval I_j^k. Note that $\mathcal{CI}_{ij}$ and $\mathcal{CI}_{ji}$ contain equivalent information. The interval pairs in $\mathcal{CI}_{ij}(\tau_i, \tau_j)$, indexed by k, can be determined from the mapping specified in (3) by applying the appropriate inverse parameterization to the endpoints of the collision zone intervals in each collision zone pair. That is,

$$\begin{aligned} \mathcal{CI}_{ij}(\tau_i, \tau_j) &= \{< [\tau_i^{-1}(\zeta_{is}^k), \tau_i^{-1}(\zeta_{if}^k)], \\ &\qquad [\tau_j^{-1}(\zeta_{js}^k), \tau_j^{-1}(\zeta_{jf}^k)] >\}. \end{aligned} \tag{6}$$

Note that if I_i^k and I_j^k do not overlap, then the two robots cannot be in the k^{th} collision zone pair simultaneously, and therefore no collision will occur in this collision zone pair. This observation forms the basis for the sufficient conditions given in Section 3.4.

For notational convenience, we introduce the variables T_{js}^k and T_{jf}^k given by

$$T_{js}^k = {\tau_j}^{-1}(\zeta_{js}^k) \quad (7)$$

$$T_{jf}^k = {\tau_j}^{-1}(\zeta_{jf}^k) \quad (8)$$

where T_{js}^k (respectively T_{jf}^k) denotes the time at which $\mathcal{A}_j$ enters (resp. exits) the k^{th} collision zone if $t_j^{start} = 0$. Note that with multiple robots, the notation T_{js}^k is ambiguous since it does not specify the particular other robot that is involved in the collision. When we use this notation, the context will make clear which other robot is involved. See Figure 2 for a graphical illustration of these quantities.

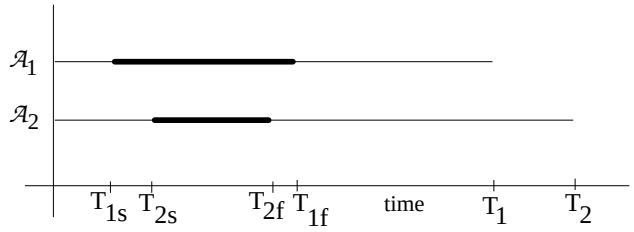

Figure 2: Timelines for robots $\mathcal{A}_1$ and $\mathcal{A}_2$. The bold lines correspond to the collision-time intervals for the robots.

Since our parameterizations are restricted to those that only delay the robot start times, we will always have parameterizations of the form

$$\tau'(t + t^{start}) = \zeta = \tau(t), \quad (9)$$

for each value of $\zeta \in [0, 1]$. Inverting the parameterizations τ' and τ we obtain

$$\tau'^{-1}(\zeta) = \tau^{-1}(\zeta) + t^{start}. \quad (10)$$

Using this notation, we can write $\mathcal{CI}_{ij}(\tau_i', \tau_j')$ as

$$\begin{aligned}\mathcal{CI}_{ij}(\tau_i', \tau_j') = \{ &< [T_{is}^1 + t_i^{start}, T_{if}^1 + t_i^{start}], \\ &\quad [T_{js}^1 + t_j^{start}, T_{jf}^1 + t_j^{start}] >, \\ &\quad \vdots \\ &< [T_{is}^n + t_i^{start}, T_{if}^n + t_i^{start}], \\ &\quad [T_{js}^n + t_j^{start}, T_{jf}^n + t_j^{start}] >\}.\end{aligned}$$

3.4 Sufficient Conditions for Collision-free Scheduling

To prevent collisions between two robots $\mathcal{A}_i$ and $\mathcal{A}_j$, it is sufficient to ensure that the times at which $\mathcal{A}_i$ could collide with robot $\mathcal{A}_j$ do not coincide with the times at which $\mathcal{A}_j$ could collide with robot $\mathcal{A}_i$, which can be assured if the two robots are not in any collision zone pair belonging to $\mathcal{PI}_{ij}$ at the same time. This amounts to ensuring that there is no overlap between the two intervals of any collision-time interval pair for the two robots. If $I_i^k \cap I_j^k = \emptyset$ for every collision-time interval pair $< I_i^k, I_j^k > \in \mathcal{CI}_{ij}(\tau_i, \tau_j)$, then no collision can occur. (Note that it is not necessary to also check the interval pairs in $\mathcal{CI}_{ji}$, since preventing collision of $\mathcal{A}_j$ with $\mathcal{A}_i$ necessarily prevents collision of of $\mathcal{A}_j$ with $\mathcal{A}_i$.) This sufficient condition leads to an optimization problem:

Optimization Problem II: *Given a set of robots with specified trajectories, find the starting times for the robots such that the total execution time for the ensemble of robots is minimized and no two intervals of any collision-time interval pair overlap.*

In Section 4, we will present a Mixed Integer Linear Program that solves this optimization problem. The sufficient condition is clearly not a necessary condition. For example, in a follow-the-leader situation where the robots move in the same direction along their paths in the collision zone, the follower robot is delayed unduly since it waits for the leader to exit the collision zone before it enters the collision zone. For now, we note that this is a conservative strategy that guarantees that no collision occurs between the two robots. We will discuss an alternative strategy that provides the minimum time collision-free schedule in Section 5.

3.5 Assumptions

We make the following assumptions to generate a collision-free coordination of the robot trajectories:

1. The only moving obstacles in the workspace are the robots, and the specified trajectory for each robot does not result in collisions with any static obstacles.
2. Each robot does not collide with the other robots when they are at their start or goal configurations.
3. The starting velocity of each of the robots is zero.
4. Each robot path is monotonic, that is, the robot does not back up along its path.
5. Each robot executes its specified trajectory, with no changes to its specified velocities, once it starts moving.
6. The robot motions are sampled at sufficient resolution so that no collisions occur during the motion between successive collision-free configurations.

4 An Integer Programming Formulation

We first develop a mixed integer linear programming (MILP) formulation for Optimization Problem II for the two robot case, and then the general case with multiple robots. t_i^{start} is the start time for robot $\mathcal{A}_i$, which is to be computed,

and T_i is the motion time required for robot $\mathcal{A}_i$ to traverse its entire trajectory when starting at time $t_i^{start} = 0$.

4.1 The Two Robot Case

First consider trajectory coordination of two robots $\mathcal{A}_i$ and $\mathcal{A}_j$. Assume the trajectory of each robot is given and that the robots can collide with each other in only one region and that the robots do not collide multiple times in the region. For each robot, identify its collision zone and compute the time interval during which it is in its collision zone. The collision-time interval $[T_{is}, T_{if}]$ of robot $\mathcal{A}_i$, where subscripts s and f indicate start and finish times respectively, indicates when robot $\mathcal{A}_j$ can collide with it. The collision-time interval $[T_{js}, T_{jf}]$ of robot $\mathcal{A}_j$ is similarly computed.

The maximum completion time for the two robots is equal to the time when the last robot completes its task, i.e., maximum $\{t_i^{start} + T_i, t_j^{start} + T_j\}$. Since we wish to minimize the completion time while ensuring the robots are not in their collision zones at the same time, the trajectory coordination problem can be stated as:

Minimize $\max\{t_i^{start} + T_i, t_j^{start} + T_j\}$
subject to
$t_i^{start} + T_{if} < t_j^{start} + T_{js}$ or $t_i^{start} + T_{is} > t_j^{start} + T_{jf}$
$t_i^{start} \geq 0$
$t_j^{start} \geq 0$

Since the objective function and the constraints are not linear, we transform them to a linear form. Let the maximum time for robots $\mathcal{A}_i$ and $\mathcal{A}_j$ to complete their motions be $t_{complete}$. Clearly $t_{complete} \geq t_i^{start} + T_i$ and $t_{complete} \geq t_j^{start} + T_j$. The disjunctive "or" constraint can be converted to an equivalent pair of constraints using an integer zero-one variable δ_{ij} and M, a large positive number ([15]). Here M can be chosen to be $T_i + T_j$. When robot $\mathcal{A}_i$ enters the collision zone first, $\delta_{ij} = 0$ and the constraint $t_i^{start} + T_{if} < t_j^{start} + T_{js}$ is active, and when robot $\mathcal{A}_j$ enters the collision zone first, $\delta_{ij} = 1$ and the constraint $t_j^{start} + T_{jf} < t_i^{start} + T_{is}$ is active. The equivalent MILP formulation is:

Minimize $t_{complete}$
subject to
$t_{complete} - t_i^{start} - T_i \geq 0$
$t_{complete} - t_j^{start} - T_j \geq 0$
$t_i^{start} + T_{if} - t_j^{start} - T_{js} - M\delta_{ij} \leq 0$
$t_j^{start} + T_{jf} - t_i^{start} - T_{is} - M(1 - \delta_{ij}) \leq 0$
$t_i^{start} \geq 0$
$t_j^{start} \geq 0$
$\delta_{ij} \in \{0, 1\}$

4.2 The Multiple Robot Case

In the general case, multiple robots, pairs of which may have multiple collision regions, must be coordinated. Here $< [T_{is}^k, T_{if}^k], [T_{js}^k, T_{jf}^k] >$ denotes the k^{th} collision-time interval pair for the two robots $\mathcal{A}_i$ and $\mathcal{A}_j$. Let N_{ij} denote the number of collision-time interval pairs for robots $\mathcal{A}_i$ and $\mathcal{A}_j$, i.e., $N_{ij} = |\mathcal{CI}_{ij}|$ and let N_{robots} be the number of robots. The binary variable δ_{ijk} is defined to be 0 if robot $\mathcal{A}_i$ enters its k^{th} collision zone with robot $\mathcal{A}_j$ before robot $\mathcal{A}_j$ and to be 1 if robot $\mathcal{A}_j$ enters its corresponding k^{th} collision zone before robot $\mathcal{A}_i$. A valid value for M is $M = \sum_{i=1}^{N_{robots}} T_i$. The MILP formulation to coordinate the motions of the robots is:

Minimize $t_{complete}$
subject to
$t_{complete} - t_i^{start} - T_i \geq 0, \quad 1 \leq i \leq N_{robots}$
$t_i^{start} + T_{if}^k - t_j^{start} - T_{js}^k - M\delta_{ijk} \leq 0,$
for all $< [T_{is}^k, T_{if}^k], [T_{js}^k, T_{jf}^k] > \in \mathcal{CI}_{ij},$
for $1 \leq i < j \leq N_{robots}$
$t_j^{start} + T_{jf}^k - t_i^{start} - T_{is}^k - M(1 - \delta_{ijk}) \leq 0$
for all $< [T_{is}^k, T_{if}^k], [T_{js}^k, T_{jf}^k] > \in \mathcal{CI}_{ij},$
for $1 \leq i < j \leq N_{robots}$
$\delta_{ijk} \in \{0, 1\}, \quad 1 \leq i < j \leq N_{robots}, \quad 1 \leq k \leq N_{ij}$
$t_i^{start} \geq 0, \quad 1 \leq i \leq N_{robots}.$

The resulting solution is guaranteed to be a collision-free trajectory coordination strategy for all the robots. The completion time constraints and collision-time interval constraints are necessary for only those robots that may collide. Note that the MILP always has a feasible solution — move the robots in sequence with only one robot in motion at any given instant. Figure 3 shows the timelines for two robots with multiple collision intervals, and Figure 4 shows the collision-free sequencing of the start times of the robots.

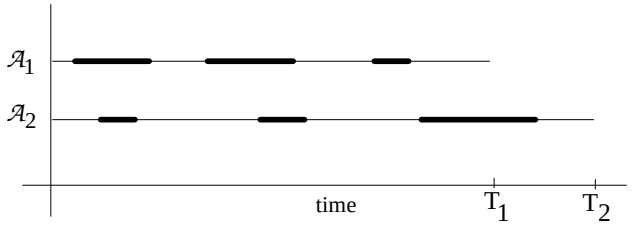

Figure 3: Timelines for robots $\mathcal{A}_1$ and $\mathcal{A}_2$ with multiple collision intervals.

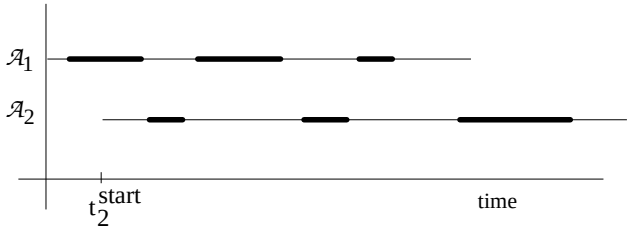

Figure 4: Collision-free timelines for robots $\mathcal{A}_1$ and $\mathcal{A}_2$, with robot $\mathcal{A}_2$ being delayed at its start.

5 Necessary Conditions for Optimality

We have so far computed start times to ensure that no two robots are simultaneously in their shared collision zones. This criterion for collision avoidance can be overly conservative, for example, when two robots $\mathcal{A}_i$ and $\mathcal{A}_j$ are moving in the same direction in a collision zone pair. We can reduce the completion time and derive the necessary conditions for collision avoidance in such cases by permitting the robots to play "follow the leader". Assume robot $\mathcal{A}_i$ moves first in its collision zone and $\mathcal{A}_j$ follows it. We need to compute how much earlier the lead robot $\mathcal{A}_i$ should start moving in its collision zone, before the follower robot $\mathcal{A}_j$ can enter its collision zone, to avoid a collision.

Shin and Zheng [19] proved that for two robots with a single collision region, delaying the start time of one of the robots provides the time-optimal trajectory modification. They compute the minimum delay time for the collision-free coordination of two robots that have a single collision zone pair by using a bisection search. The delay time of the follower robot, or equivalently, the lead time of the lead robot, is initialized to a value that guarantees the lead robot will exit its collision zone before the follower robot enters its collision zone. The minimum lead time in the collision zone for which the lead robot can still avoid a collision with the follower robot is then computed using bisection search.

We extend this idea of computing the necessary conditions for collision avoidance to multiple robots, where pairs of robots may have multiple collision zone pairs. Given two robots $\mathcal{A}_i$ and $\mathcal{A}_j$ that have more than one collision zone pair, we treat each collision zone pair independently when computing the lead times using bisection. For the kth collision zone pair, we compute the minimum time T_{ijk}^{lead} that robot $\mathcal{A}_i$ must lead robot $\mathcal{A}_j$ by at the start of its kth collision zone to avoid a collision, and the minimum time T_{jik}^{lead} that robot $\mathcal{A}_j$ must lead robot $\mathcal{A}_i$ by at the start of its kth collision zone to avoid a collision. The corresponding follow-the-leader constraints are $t_i^{start}+T_{is}^k+T_{ijk}^{lead} < t_j^{start}+T_{js}^k$ when $\mathcal{A}_i$ leads through the collision zone, or $t_j^{start}+T_{js}^k+T_{jik}^{lead} < t_i^{start}+T_{is}^k$ when $\mathcal{A}_j$ leads through the collision zone.

The maximum value of T_{ijk}^{lead} is T_i^k, the time taken for robot $\mathcal{A}_i$ to traverse its kth collision zone. Since $T_{ijk}^{lead} \leq T_i^k$, we define a new variable $T_{ie}^k = \min\{T_{is}^k + T_{ijk}^{lead}, T_{if}^k\}$ where $T_{if}^k = T_{is}^k + T_i^k$. T_{ie}^k, the collision-free *entry time*, is the time from start in robot $\mathcal{A}_i$'s trajectory, when $\mathcal{A}_i$ enters its kth collision zone pair before $\mathcal{A}_j$, at which robot $\mathcal{A}_j$ can enter its collision zone without causing a collision. Similarly, define $T_{je}^k = \min\{T_{js}^k + T_{jik}^{lead}, T_{jf}^k\}$. The updated follow-the-leader constraints are $t_i^{start} + T_{ie}^k < t_j^{start} + T_{js}^k$ when $\mathcal{A}_i$ leads through the collision zone, or $t_j^{start} + T_{je}^k < t_i^{start} + T_{is}^k$ when $\mathcal{A}_j$ leads through the collision zone. The robots $\mathcal{A}_i$ and $\mathcal{A}_j$ do not collide when their start times satisfy these follow-the-leader constraints over all their collision zone pairs.

To extend this formulation to multiple robots, we include these disjunctive constraints for every pair of robots that can potentially collide. The minimum completion time over all robots is obtained using the following formulation:

Minimize $t_{complete}$
subject to
$t_{complete} - t_i^{start} - T_i \geq 0, \quad 1 \leq i \leq N_{robots}$
$t_i^{start} + T_{ie}^k - t_j^{start} - T_{js}^k - M\delta_{ijk} \leq 0$
for all $< [T_{is}^k, T_{if}^k], [T_{js}^k, T_{jf}^k] > \in \mathcal{CI}_{ij}$
$1 \leq i < j \leq N_{robots}$
$t_j^{start} + T_{je}^k - t_i^{start} - T_{is}^k - M(1 - \delta_{ijk}) \leq 0$
for all $< [T_{is}^k, T_{if}^k], [T_{js}^k, T_{jf}^k] \in \mathcal{CI}_{ij}$,
$1 \leq i < j \leq N_{robots}$
$\delta_{ijk} \in \{0, 1\}, \quad 1 \leq i < j \leq N_{robots}, 1 \leq k \leq N_{ij}$
$t_i^{start} \geq 0, \quad 1 \leq i \leq N_{robots}.$

The solution to the above MILP solves Optimization Problem I and gives the minimum time coordinated trajectories of the robots when only their start times can change.

6 Extensions

Our problem formulation so far has focused on single body robots with specified trajectories. We now discuss useful extensions to the basic formulation.

6.1 Articulated Robots

To coordinate articulated robots with multiple links, we consider motions of the individual links. An articulated robot R consists of a set of links $\{\mathcal{A}_i\}$. Let $R[i]$ be the robot to which link $\mathcal{A}_i$ belongs. The motions of links of an articulated robot are separated by constant time offsets. Let $\mathcal{A}_i$ begin moving time T_i^R after the first moving link of $R[i]$ begins moving. That is, $t_i^{start} = t_{R[i]}^{start} + T_i^R$ where $t_{R[i]}^{start}$ is the start time of robot $R[i]$. Let N_{links} be the total number of robot links. Note that the start time and motion time of a link may depend on the start and motion times of links that precede it in the articulated chain. Thus the formulation for a set of articulated robots is:

Minimize $t_{complete}$
subject to
$t_{complete} - t_{R[i]}^{start} - T_i^R - T_i \geq 0, \quad 1 \leq i \leq N_{links}$
$t_{R[i]}^{start} + T_i^R + T_{ie}^k - t_{R[j]}^{start} - T_j^R - T_{js}^k - M\delta_{ijk} \leq 0$
for all $< [T_{is}^k, T_{if}^k], [T_{js}^k, T_{jf}^k] > \in \mathcal{CI}_{ij}$,
for $1 \leq i < j \leq N_{links}$ and $R[i] \neq R[j]$
$t_{R[j]}^{start} + T_j^R + T_{je}^k - t_{R[i]}^{start} - T_i^R - T_{is}^k - M(1 - \delta_{ijk}) \leq 0$
for all $< [T_{is}^k, T_{if}^k], [T_{js}^k, T_{jf}^k] > \in \mathcal{CI}_{ij}$
for $1 \leq i < j \leq N_{links}$ and $R[i] \neq R[j]$
$\delta_{ijk} \in \{0, 1\}, \quad 1 \leq i < j \leq N_{links}, \quad 1 \leq k \leq N_{ij}$
$t_{R[i]}^{start} \geq 0, \quad 1 \leq i \leq N_{robots}.$

The completion time constraints are necessary for all links of a robot that can potentially have a collision. The collision-time interval constraints are necessary for only those robots that have one or more links involved in a potential collision.

6.2 Specifying Sequencing Constraints

In certain tasks, it may be necessary for one robot to complete a particular operation or reach a certain point before another robot performs a subsequent operation. This can occur in sequenced assembly tasks, or in welding workcells where the primary welds must be completed before secondary welds. Consider the requirement that $\mathcal{A}_i$ has to reach $\mathbf{q}_i$ before $\mathcal{A}_j$ reaches $\mathbf{q}_j$. For the unmodified trajectories, let the time taken for $\mathcal{A}_i$ to reach $\mathbf{q}_i$ be T_{q_i} and for $\mathcal{A}_j$ to reach $\mathbf{q}_j$ be T_{q_j}. The sequencing constraint can then be written as $t_i^{start} + T_{q_i} < t_j^{start} + T_{q_j}$. Such constraints for multiple robots can be easily added to the formulation.

7 Complexity

The integer programming formulation of our problem suggests it is an NP-complete problem ([4]). We first consider the decision version of the No-wait Jobshop Scheduling problem (Sahni and Cho [18], Goyal and Sriskandarajah [6]), which is NP-complete. Each job consists of an ordered set of tasks, where each task is to be performed by a specific processor. The tasks for each job must be executed in sequence without breaks between them. Each processor can perform no more than one task at any time instant, and each job can be worked on by only one processor at any time instant. The goal is to minimize the makespan (i.e., the maximum time of completion of any task).

The above problem can be transformed to our Multiple Robot Scheduling problem. Let each job j model the trajectory of robot $\mathcal{A}_j$. Let each task $t_k[j]$ model the kth trajectory segment for robot $\mathcal{A}_j$, where each trajectory segment is a contiguous collision zone segment or collision-free segment. Let processor p_i model the region r_i, where each region contains one or more trajectory segments. No two trajectory segments that are in the same region can be executed at the same time. The length of each task is the time taken by the robot to traverse the corresponding segment. The goal is to minimize the completion time of the robots. It follows that the decision version of the Multiple Robot Scheduling problem is NP-complete, and that the optimization problem is NP-hard.

8 Implementation

We have implemented software in C++ to coordinate the motions of polyhedral robots with specified trajectories (Figure 5) and have a preliminary implementation for articulated robots. We compute the collision zones using the PQP collision detection package (Larsen et al. [10]). The robot configurations are specified at constant time intervals. To determine the collision zones, each robot is stepped through its trajectory, and at each trajectory point, all the remaining robots are moved through their complete trajectories to detect collisions. So for N robots where each robot has T trajectory points, collision detection is performed $O(N^2T^2)$ times.

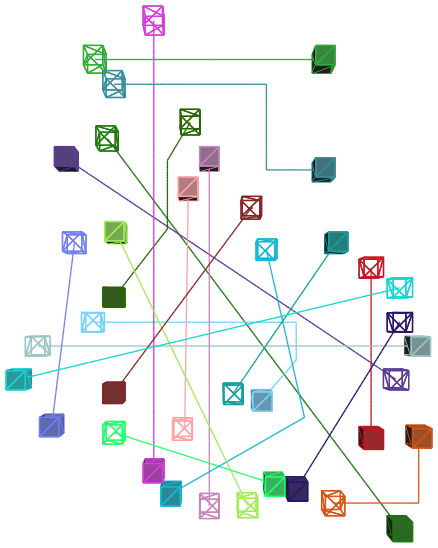

Figure 5: Overhead view of the paths of 20 robots, with their initial configurations indicated by solid cubes.

Num. of robots	Num. of collision zones	Collision detection time (secs)	MILP time (secs)
3	2	< 1	0.02
5	10	2.4	0.02
10	27	9.8	0.11
15	65	23.4	0.53
20	79	36.8	1.83

Table 1: Comparison of sample run times for 100 frames.

Using the computed collision-time interval pairs, we generate the corresponding MILP formulation and solve it using CPLEX [8], a commercial optimization package. See Table 1 for runtime data on a Sun Ultra 10 for single body robots. Note that the problem complexity depends primarily on the number of collision zones, to a lesser extent on the number of robots, and is relatively independent of the number of degrees of freedom of the robots. Our preliminary experiments indicate that the MILP time dominates the running time as the number of collision zones increases. Example animations may be seen at `www.cs.rpi.edu/~sakella/multiplerobots/`.

9 Conclusion

We have developed an optimization formulation that enables the minimum time collision-free coordination of multiple robots with specified trajectories when only their start times can be changed. The principal advantage of our MILP formulation is that it permits the collision-free coordination of a large number of robots (up to 20 robots). The problem complexity depends on the number of robots and the number of potential collisions, and is relatively independent of the number of degrees of freedom of the robots. Although the optimal trajectory coordination of multiple robots is NP-hard, the availability of efficient collision detection software and integer programming solvers makes this approach practical.

There are several issues for future work. Developing polynomial time approximation algorithms for the task of selecting start times and characterizing the quality of these solutions is important. An alternative approach to minimizing the completion time is modifying trajectories by tuning the velocity of each of the robots. Identifying the conditions under which we can do this, and developing techniques to generate the optimized trajectories is important. Exploring stochastic versions of the task that involve timing uncertainties would be useful. Finally, exploring applications of this work in computer graphics for choreographing animation characters is another interesting direction.

Acknowledgments

Srinivas Akella was supported in part by RPI and the Beckman Institute at UIUC. Seth Hutchinson was supported by NSF under Award Nos. CCR-0085917 and IIS-0083275. Thanks to Prasad Akella and Charles Wampler at General Motors for suggesting the problem and helpful discussions. Andrew Andkjar implemented animation software that interfaced with PQP and helped generate examples. Discussions with Jufeng Peng helped clarify several points in the paper.

References

[1] Z. Bien and J. Lee. A minimum time trajectory planning method for two robots. *IEEE Transactions on Robotics and Automation*, 8:414–418, June 1992.

[2] C. Chang, M. J. Chung, and B. H. Lee. Collision avoidance of two robot manipulators by minimum delay time. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(3):517–522, Mar. 1994.

[3] M. Erdmann and T. Lozano-Perez. On multiple moving objects. *Algorithmica*, 2(4):477–521, 1987.

[4] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, New York, 1979.

[5] M. R. Garey, D. S. Johnson, and R. Sethi. The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research*, 1:117–129, 1976.

[6] S. K. Goyal and C. Sriskandarajah. No-wait shop scheduling: Computational complexity and approximate algorithms. *Opsearch*, 25(4):220–244, 1988.

[7] J. E. Hopcroft, J. T. Schwartz, and M. Sharir. On the complexity of motion planning for multiple independent objects: PSPACE-hardness of the "warehouseman's problem". *International Journal of Robotics Research*, 3(4):76–88, 1984.

[8] ILOG, Inc., Incline Village, NV. *CPLEX 6.0 Documentation Supplement*, 1998.

[9] K. Kant and S. W. Zucker. Toward efficient trajectory planning: The path-velocity decomposition. *International Journal of Robotics Research*, 5(3):72–89, Fall 1986.

[10] E. Larsen, S. Gottschalk, M. Lin, and D. Manocha. Fast distance queries using rectangular swept sphere volumes. In *IEEE International Conference on Robotics and Automation*, San Francisco, CA, Apr. 2000.

[11] J.-C. Latombe. *Robot Motion Planning.* Kluwer Academic Publishers, Norwell, MA, 1991.

[12] S. M. LaValle and S. A. Hutchinson. Optimal motion planning for multiple robots having independent goals. *IEEE Transactions on Robotics and Automation*, 14(6):912–925, Dec. 1998.

[13] E. L. Lawler, J. K. Lenstra, A. H. G. R. Kan, and D. B. Schmoys. Sequencing and scheduling: Algorithms and complexity. In S. C. Graves, A. H. G. R. Kan, and P. H. Zipkin, editors, *Handbooks in Operations Research and Management Science, Vol. 4, Logistics of Production and Inventory*, pages 445–522. North-Holland, 1993.

[14] S. Leroy, J.-P. Laumond, and T. Simeon. Multiple path coordination for mobile robots: a geometric algorithm. In *16th International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 1118–1123, Stockholm, Sweden, Aug. 1999.

[15] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization.* John Wiley and Sons, New York, 1988.

[16] P. A. O'Donnell and T. Lozano-Perez. Deadlock-free and collision-free coordination of two robot manipulators. In *IEEE International Conference on Robotics and Automation*, pages 484–489, Scottsdale, AZ, May 1989.

[17] J. Reif and M. Sharir. Motion planning in the presence of moving obstacles. In *Proceedings of the 26th Annual Symposium on the Foundations of Computer Science*, pages 144–154, Portland, Oregon, Oct. 1985.

[18] S. Sahni and Y. Cho. Complexity of scheduling shops with no wait in process. *Mathematics of Operations Research*, 4(4):448–457, Nov. 1979.

[19] K. G. Shin and Q. Zheng. Minimum-time collision-free trajectory planning for dual robot systems. *IEEE Transactions on Robotics and Automation*, 8(5):641–644, Oct. 1992.