Title
Resource Allocation Algorithm for Multi-Vehicle Systems with Non holonomic Constraints

Permalink
https://escholarship.org/uc/item/2qr347j1

Authors
Rathinam, Sivakumar
Sengupta, Raja
Darbha, Swaroop

Publication Date
2005-05-01
A Resource Allocation Algorithm for Multi-Vehicle Systems with Non holonomic Constraints

Sivakumar Rathinam, Raja Sengupta, and Swaroop Darbha
A Resource Allocation Algorithm for Multi-Vehicle Systems with Non holonomic constraints

Sivakumar Rathinam¹, Raja Sengupta², Swaroop Darbha³

Abstract— Multi-vehicle systems are naturally encountered in civil and military applications. Cooperation amongst individual “miniaturized” vehicles allows for flexibility to accomplish missions that a single large vehicle may not readily be able to accomplish. While accomplishing a mission, motion planning algorithms are required to efficiently utilize a common resource (such as the total fuel in the collection of vehicles) or to penalize a collective cost function (such as to minimize the maximum time taken by the vehicles to reach their intended target). The objective of this paper is to present a constant factor approximation algorithm for planning the path of each vehicle in a collection of vehicles, where the motion of each vehicle must satisfy non-holonomic constraints.

Keywords— Cooperative control, Travelling salesman, Motion Planning, Non holonomic, Dubins

I. INTRODUCTION

This paper is about the assignment of m targets to n vehicles. The motion of the vehicles is assumed to be non-holonomic, i.e., the yaw rate of the vehicle is constrained. Each target is to be visited by one and only one vehicle. Given a set of targets and the path constraints on the vehicles, the problem P addressed in this paper is

• to assign each vehicle, a sequence of targets to visit, and
• to find the paths of the vehicles to their respective targets that satisfy yaw rate constraints, so that the total distance travelled by the collection of vehicles to reach their assigned targets is minimized.

This resource allocation problem that is addressed in this paper belongs to a class of cooperative control problems of unmanned aerial vehicles that has received significant attention in the recent years [2], [3], [4], [5], [7]. Without the non-holonomic (yaw rate) constraints, our problem is essentially a Multi Vehicle Travelling Salesman Problem with symmetric costs (distances) between the targets. However, the non-holonomy makes the costs non-Euclidean and asymmetric. The minimum distance that can be travelled by a non-holonomic vehicle going from target A to target B depends on its heading at target A and at target B. This distance can be determined using the well-known result of L.E. Dubins [1]. In general, it will not be the length of the straight line joining the two targets.

The distance between the two targets is also not symmetric. The minimum distance that can be travelled by a vehicle starting at target A with heading $\psi_A$ and arriving at target B with heading $\psi_B$ is not equal to the minimum distance that can be travelled by a vehicle starting at target B with heading $\psi_B$ and arriving at target A with heading $\psi_A$. Thus we can think of our problem as a type of Multi Vehicle-Asymmetric Travelling Salesman Problem with the costs satisfying the triangular inequality and the motion being non-holonomic.

The Asymmetric Travelling Salesman Problem (ATSP) is well known in combinatorial optimization to be NP-hard. Currently, there are no algorithms with a constant approximation factor available for solving ATSP problems even when the costs satisfy triangular inequality. A approximation factor $\beta(P, A)$ of using an algorithm $A$ to solve the problem $P$ (objective is minimize some cost function) is defined as

$$\beta(P, A) = \sup S \frac{C(S, A)}{C_*(S)},$$

(1)

where $S$ is a problem instance, $C(S, A)$ is the cost of the solution by applying algorithm $A$ to the instance $S$ and $C_*(S)$ is the cost of the optimal solution of $S$. The algorithm by Markus Blaser given in [16] for a single vehicle Asymmetric Travelling Salesman Problem problem (visiting $n$ targets) has a approximation factor $0.9999\log n$. Hence, the bound $\to \infty$ as $n \to \infty$. There are also other kind of algorithms where the bound $\to \infty$ due to the data but are independent of $n$. For example, the algorithm by Kumar and Li given in [17] has a approximation ratio which is a increasing function of $\frac{d_{\max}}{d_{\min}}$. Here, $d_{\max} = \max_{i,j} C(i,j)$ and $d_{\min} = \min_{i,j} C(i,j)$, where $C(i,j)$ denotes the costs between two targets $i$ and $j$.

The problem P dealt in this paper also assumes that the angles of approach at the targets are not given and hence, the distances (or costs) required to travel between any two targets vary depending on the path chosen by the vehicle. We assume the distance between targets is greater than twice the minimum turning radius of the vehicles (assumption on $d_{\min}$). When there is one vehicle and several targets we derive a constant factor of 4.64. When there are many vehicles we derive a constant factor of 6.08. Our algorithms are essentially derived by combining constant factor approximation algorithms for Euclidean TSP’s with the Dubins results.

A. Related work

A more general version of problem P with multiple tasks required for each target was formulated in [15]. Task allocation and multi-assignment problems are solved using

1. Graduate Student, Department of Civil Engg, University of California, Berkeley, CA 94703, corresponding author e-mail: rsiva@berkeley.edu
2. Assistant Professor, Department of Civil Engineering, University of California, Berkeley CA 94702
3. Associate Professor, Department of Mechanical Engg, Texas A and M University, College Station, TX 77843-3123.
network flow and auction algorithms in [6], [7]. In [7], the number of vehicles are assumed to be greater than the number of tasks to be performed for solving the bilinear assignment problem. In this paper, we make no assumptions about the relative number of vehicles or the number of targets.

Theju et. al. in [8] present methods for solving the multi-vehicle, target assignment problem in the presence of threats with the goal of minimizing the maximum path length. In the absence of the threats, this problem is actually related to the dual of the multiple vehicle problem addressed in this paper where the sum of the distances travelled by the vehicles is minimized. Problem $P$ can also be formulated as a mixed integer linear programming problem and related versions of this problem with timing constraints are tackled in [10] [11]. This approach yields an optimal solution but is computationally expensive. The paper that is most related to our multiple vehicle problem is the work by Zhijun et.al [18] where they provide heuristics for multiple vehicles tracking moving targets using clustering and gradient techniques. Even though [18] consider moving targets, their main results are for static targets which is essentially the problem that is addressed in this paper. Also heuristics for more general versions of the problem $P$ are presented in [9] [13], but there are no bounds. Here, we present a polynomial time approximation algorithm with a constant factor bound. We conjecture that the output solution of the multiple vehicle algorithm presented in this paper can also be used as an initial solution and say, coupled with the heuristics presented in [18] to get better results.

Also related to this work is the paper by Yang et. al. [14] where they consider path planning for a UAV with kinematic constraints given fixed initial and final positions in the presence of obstacles. The UAV in their work is required to visit a target and then reach a final position avoiding threats and other obstacles. This is related to the single vehicle problem addressed in this paper in the absence of obstacles when there is one target on the tour. The single vehicle problem addressed in this paper has been previously addressed by [19]. In their work, they bound the distance of the dubins path between any points $(x_1, y_1, \theta_1)$ and $(x_2, y_1, \theta_2)$ in terms of the Euclidean distance between the corresponding points. Also, using this result, they propose an algorithm which bounds the total distance travelled by the vehicle in terms of the Euclidean distance tour. In this paper, we provide an algorithm for multiple vehicles. Also, by making assumptions about the target positions, the bound that we propose in for the approximation factor of the algorithm.

We present an approximation algorithm for multiple vehicles that is independent of the number of targets or vehicles by making an assumption about the minimum distances between the targets. The assumption is that the targets are at least $2r$ apart, where $r$ is the minimum turning radius of the vehicle. This is reasonable assumption in the context of unmanned aerial vehicles which carry sensors that have footprints that are greater than $2r$. For example, figure 1 shows a unmanned aerial vehicle that we use for tracking and other mapping applications. The vehicle has a minimum turning radius of around 100 meters. If the vehicle is flying at a height of at least 150 meters using a 80 degree wide angle camera, then the width of the area covered in the images is at least 200 meters. Therefore, the targets within 200 meters can be seen from the same vehicle position. Hence, we assume that $d_{min}$, the minimum distance between the targets, is $\geq 2r$. The algorithm given in this paper for multiple vehicles require $O((n+m)^3)$ steps where $n$ is the number of vehicles and $m$ is the number of targets.

II. RESOURCE ALLOCATION PROBLEM

Let $x_t(v_i,t) = (x(v_i,t), y(v_i,t), \theta(v_i,t))$ denote the position of vehicle $v_i$ at time $t$. Let each vehicle start at an initial heading $\theta(v_i,0) = \alpha_i$. Similarly, let $x_d(j,t) = (x(d_j,t), y(d_j,t))$ denote the position of target $d_j$ at time $t$. Since the targets are assumed to be static, let $\dot{r}(d_j) = r(d_j,t) = r(d_j,t') \forall t,t'$. Given a set of vehicles $\{v_1, v_2, ... v_n\}$ and targets $\{d_1, d_2, ... d_m\}$, the problem is to

- assign a sequence of targets $P_i$ for each vehicle to visit such that $\{d_1, d_2, ... d_m\} = \bigcup P_i$ and $P_i \cap P_j = \emptyset \text{ if } i \neq j$.
- assign for each vehicle $v_i$, a path through the sequence $P_i$ such that the path of each vehicle $v_i$ satisfies the following equations of motion:

$$\begin{align*}
\frac{dx(v_i,t)}{dt} &= v_o \cos(\theta(v_i,t)), \\
\frac{dy(v_i,t)}{dt} &= v_o \sin(\theta(v_i,t)), \\
\frac{d\theta(v_i,t)}{dt} &= \Omega \text{ where } \Omega \in [-\omega, +\omega],
\end{align*}$$

where, $v_o$ denotes the speed, $\omega$ represents the bound on the yaw rate and $r = \frac{\pi}{2} \beta_{d_i}$ is the minimum turning radius of the vehicle.

Let the sequence $P_i$ for vehicle $v_i$ be $d_{i_1}, ... d_{i_k}$. Assigning a path for a vehicle $v_i$ through its sequence $P_i$ of targets also implies assigning the angles of approach $\beta_{d_i}$ at each target and assigning the angle of return $\beta_{v_i}$.
III. Algorithms

In this section, we present approximation algorithms for the single vehicle resource allocation problem (i.e., when \( n = 1 \)) and the multiple vehicle resource allocation problem (\( n > 1 \)). Before we present the algorithms, we present the result by L.E. Dubins which forms the motivation for the paths chosen in the algorithms.

L.E. Dubins [1] gives the optimal path the vehicle must travel subject to the path constraints given by equations 2. Dubins result states that the curve joining the two points \((x_1, y_1, \theta_1)\) and \((x_2, y_2, \theta_2)\) that has minimal length subject to equations 2, consists of at most three pieces, each of which is either a straight line or an arc of of a circle of radius \(r\). This curve must necessarily be

1. an arc of a circle of radius \(r\), followed by a line segment, followed by an arc of a circle of radius \(r\), or
2. a sequence of three arcs of circles of radius \(r\), or
3. a subpath of a path of type 1 or 2.

An example of such a path is given in figure 2. The vehicle in this path first turns clockwise at the minimum turning radius, then travels straight, and finally turns in the counter clockwise direction to reach the final target. Henceforth, the turning clockwise motion at minimum turning radius is denoted by \(R\), the turning counter clockwise motion at minimum turning radius is denoted by \(L\) and the travelling straight motion is denoted by \(S\). Thus the path in figure 2 is an \(RSL\) path. In the special case where the points \((x_1, y_1), (x_2, y_2)\) are at least \(2r\) apart and when the angle \(\theta_2\) is not specified, the following lemma follows from the results in [1]:

**Lemma III.1:** Given \((x_1, y_1, \theta_1), (x_2, y_2)\), if \(\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \geq 2r\), there exists a unique RS path and a unique LS path from \((x_1, y_1, \theta_1)\) and \((x_2, y_2)\) satisfying equations 2.

A. Single Vehicle Algorithm (SVA).

First, we give a simple algorithm \(S\) for the vehicle \(v_1\) to find a path to travel from positions \((x(v_1), y(v_1), \alpha_1)\) to \(\bar{r}(d_j)\). Note that the final approach angle at the position \(\bar{r}(d_j)\) is not specified. Algorithm \(S\) is as follows:

1. Find the distances of two possible paths the vehicle could take: \(RS\) and \(LS\).
2. Choose the path that has the minimum distance.

   Once, this path is followed, the vehicle reaches the position \(\bar{r}(d_j)\) at some final angle \(\theta\). Hence, the final approach angle is automatically selected here.

The algorithm \(SVA\) to solve the single vehicle problem is as follows:

1. Use Christofides algorithm [20] to solve the Euclidean TSP problem assuming the kinematic constraints are absent. The output is a sequence of targets \(\{d_1, d_2, ...d_m\}\) for the vehicle to visit.
2. Use the above sequence and construct paths using algorithm \(S\) between any two consecutive targets. For example, use algorithm \(S\) to construct a path from \((x(v_1), y(v_1), \alpha_1)\) to \(\bar{r}(d_1)\). Say, the vehicle reaches the target \(d_1\) at an angle \(\theta\). Again, use algorithm \(S\) to construct a path from \(\bar{r}(d_1), \theta\) to \(\bar{r}(d_2)\) and so on.

**Lemma III.2:** The path by the algorithm \(SVA\) satisfies equations 2.

**Proof:** Follows from the construction of the path in step 2 of the algorithm \(SVA\).

A.1 Analysis

The following results show the approximation factor for this algorithm. First, the distance travelled by the vehicle using algorithm \(S\) is bounded with respect to the Euclidean distance between the vehicle and the target positions. Let the vehicle be located at the origin \(O\) at any angle of orientation. Let the coordinates of the target be \((x, y)\). Assume that the vehicle and the target location \(T\) are separated by a Euclidean distance greater than equal to \(2r\). Let \(D(x, y)\) be the distance travelled by the vehicle using algorithm \(S\) to the target from \(O\).

**Lemma III.3:** Without loss of generality, let the vehicle be position at \((0, 0, \frac{\pi}{2})\) and the target be positionated at \((x, y)\) with \(\sqrt{x^2 + y^2} \geq 2r\). The path \(RS\) is optimum if \(x > 0\) and the path \(LS\) is optimum if \(x < 0\). Both \(RS\) and \(LS\) are optimal if \(x = 0\).
Hence, if the figure 4. Also, consider any other point $P = (x, y)$.

Proof: Consider the point $Q$ with coordinates $(0, y)$. The distance using the path $LS$ to $P'$ is less than using $RS$. Likewise, the distance using the path $RS$ is optimal to reach the location $P$. Both the distances of the paths $RS$ and $LS$ are equal if the points lie on $x = 0$.

Lemma III.4: Consider the set $S = \{(x, y) : \sqrt{(x^2 + y^2)} = R\}$. Let $R \geq 2r$. Then $D(x, y)$ for $(x, y) \in S$ is maximized when $x = 0, y = -R$.

Proof: Basically since the distances are symmetric about $x = 0$ axis, without loss of generality, only $RS$ paths need be considered to targets in the first and the fourth quadrant. Let $S' = \{(x, y) : x \geq 0, \sqrt{(x^2 + y^2)} = R\}$. Consider the point $Q$ with coordinates $(0, -R)$ as shown in the figure 4. Also, consider any other point $P = (x, y) \in S'$. The distances of the path $RS$ to the point $Q$ is greater than or equal to the distance to the point $P$. Therefore, $D(x, y)$ for $(x, y) \in S$ is maximized when $x = 0, y = -R$.

Lemma III.5: Let $R = \sqrt{(x^2 + y^2)} \geq 2r$. The ratio $\frac{D(x, y)}{\sqrt{(x^2 + y^2)}}$ is maximized when $x = 0$ and $y = -2r$. The ratio is $\pi + 1 - \tan^{-1}(0.5)$. Hence, it is enough to consider the maximization of the ratio on the set of points given by $\{(0, -R) : R \geq 2r\}$. Then from figure 4, $D(0, -R) = 2\pi r - 2\gamma r + R = (2\pi - 2\tan^{-1}\frac{R}{\gamma})r + R$. Hence, if $R \geq 2r$, the ratio $\frac{D(x, y)}{\sqrt{(x^2 + y^2)}}$ for $x = 0$ and $y = -R$ is $\frac{D(0, -R)}{R} = (2\pi - 2\tan^{-1}\frac{R}{\gamma})\frac{R}{\gamma} + 1$ which is maximized at $R = 2r$.

Once the distances between the individual points are bounded, it can be combined with the Christofides result to get a approximation for the single vehicle problem.

Theorem III.1: Algorithm(SVA) solves the single vehicle problem with an approximation factor $\beta(\text{RAS}(1), \text{SVA}) = \frac{3}{2}(\pi + 1 - \tan^{-1}(0.5)) \approx 4.64$.

Proof: The Christofides algorithm applied to the Euclidean TSP has an approximation factor of $\frac{3}{2}$. By Lemma 2.1, the maximum ratio of the distance of the path constructed using algorithm $S$ to the euclidian distance is $\pi + 1 - \tan^{-1}(0.5)$. Combining these two results, $\beta(\text{RAS}(1), \text{SVA}) = \frac{3}{2}(\pi + 1 - \tan^{-1}(0.5)) \approx 4.64$, i.e. the algorithm SVA has an approximation factor $\frac{3}{2}(\pi + 1 - \tan^{-1}(0.5))$.

B. Multiple Vehicle Algorithm (MVA)

The algorithm MVA for the multi vehicle path planning problem is as follows:

1. Construct a complete graph with nodes being all the vehicle and target positions. Assign the Euclidean distance as the cost to each edge that joins a vehicle to a target and a target to a target. Assign zero cost to an edge that joins any two vehicles.

2. Find the minimum spanning tree of the graph using Prims algorithm [20]. This minimum spanning tree will contain exactly $n - 1$ zero cost edges where $n$ is the number of vehicles (figure 5).

3. Remove the zero cost edges to get a tree for each vehicle.

4. For each tree corresponding to a vehicle, double its edges to construct a Eulerian graph (figure 6). Then construct a tour for each vehicle based on the Eulerian graph. A tour for each vehicle is a sequence of targets for it to visit (figure 7). (This step is similar to Tarjan’s algorithm for a single vehicle Euclidean TSP [20]).

5. Use the sequence derived from the previous step for the each vehicle and construct paths using algorithm $S$ between any two consecutive targets as in the single vehicle case (figure 8).

B.1 Analysis

First we show algorithm MVA, without step 5, has an approximation factor of 2. Then as in the single vehicle case, each edge is replaced with a path that satisfies the minimum curvature constraints yielding a bound similar to the single vehicle problem.

Let $G(V, E)$ be a graph with vertices $V = \{v_1, v_2, \ldots, v_n, d_1, d_2, \ldots, d_m\}$. The graph is complete, that is, there is a edge $e_{ab}$ between every pair of vertices $a$ and
b. Each edge is assigned a cost $C : E \rightarrow R^+$ such that $C(e_{ab}) = \|a - b\|_2$ if $a, b \notin \{v_1, v_2, ..., v_n\}$ and $C(e_{ab}) = 0$ if $\{a, b\} \subseteq \{v_1, v_2, ..., v_n\}$. Note if $\{a, b\} \not\subseteq \{v_1, v_2, ..., v_n\}$, $\|a - b\|_2 > 0$.

Lemma III.6: The minimum spanning tree $MST$ of the graph $G$ computed using the Prims algorithm has $n - 1$ zero cost edges.

Proof: Prims algorithm can be started at any arbitrary vertex in the graph $G$. Since, the Prims algorithm is greedy, once a vehicle vertex is reached, it will add further $n - 1$ zero costs edges before reaching any target vertex. The algorithm cannot add more than $n - 1$ zero cost edges, because it would form a cycle otherwise. Hence there will be exactly $n - 1$ zero cost edges. □

Now the following theorem gives a bound for visiting all the targets based on the Euclidean distances. Let $MST$ be the minimum spanning tree from step 2 of algorithm $MVA$.

Lemma III.7: Step 4 of algorithm $MVA$ produces a sequence of targets for each vehicle to visit and has an approximation ratio of 2.

Proof: Consider the optimal tours for all the vehicles for the graph $G(V, E)$ based on the cost function $C$. From each tour, remove one of the two edges that connect to the vehicle vertex to yield a tree for each vehicle as shown in the figure 9. Now, add an appropriate set of $n - 1$ zero cost edges to join all the trees connected to the vehicles to make a joined tree, say $T'$ (figure 10). Clearly the cost of $T'$ must be ≥ to the cost of $MST$. Hence the cost of the tour constructed using Step 4 of the algorithm $MVA$ must be lower bounded by $Cost(MST)$. $Cost(MST)$ is the sum of the distances of all the edges in $MST$.

Let $MSD_i$ represent the tree for the $i^{th}$ vehicle after removing the zero cost edges from the minimum spanning tree. Each tree $MSD_i$ must be a minimum spanning tree for the subset of targets connected to the corresponding vehicle. Hence doubling the edges results in a Eulerian graph for the corresponding subset of vertices with a cost $= 2Cost(MSD_i)$. As given in [20], a TSP tour can be constructed for each vehicle with a total cost upper bounded by $2\sum_i Cost(MSD_i)$ or $\leq 2Cost(MST)$. Hence the tour constructed has a bound of 2. □

Now, the following is the result for the approximation factor of the algorithm $MVA$.

Theorem III.2: Algorithm($MVA$) solves the multiple vehicle problem with an approximation factor $\beta(RAS(n), MVA) = 2(\pi + 1 - \tan^{-1}(0.5)) \approx 6.08$ in $O((n + m)^2)$ steps.

Proof: Follows from lemma III.5 and III.7. The time complexity is basically determined by three steps: finding the minimum spanning tree which takes $O((n + m)^2)$; finding the Eulerian tour which also takes $O((n + m)^2)$ steps; finding the $RS, LS$ approximations also take
Add zero cost edges between the vehicles to form a tree

Fig. 9. Remove one of the edges incident on each vehicle in the optimal tour

Add zero cost edges between the vehicles to form a tree

Fig. 10. Adding the zero cost edges to form a tree.

$O((n + m)^2)$ steps.

IV. DIRECTIONS FOR FUTURE WORK

This paper presented a constant factor approximation algorithm for multi-vehicle systems with non-holonomic constraints. The basic assumption was that the target points are at least a distance of 2 apart. The vehicle was modeled as a simple unicycle model with yaw rate constraints. Even if the dynamics of the vehicle is included, as long as the distances travelled satisfy the triangular inequality constraints, the results given in this paper can be generalized.

There are many future directions for this work. The issues that can addressed are targets with ordering constraints; targets requiring multiple visits; and stochastic uncertainty.

REFERENCES


