Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

Improving the Efficiency of Rapidly-exploring Random Trees Using a Potential Function Planner

Ian Garcia and Jonathan P. How

Aerospace Controls Laboratory
Massachusetts Institute of Technology
{ianmga, jhow}@mit.edu

Abstract— **This paper presents an improvement of the standard randomized path planning algorithm, and uses this new approach to design reconfiguration maneuvers for large formations of spacecraft. A spacecraft reconfiguration maneuver is a difficult 6 DOF trajectory design problem with nonlinear attitude dynamics and non-convex constraints. In addition, some of the constraints couple the positions and attitudes of the spacecraft, which makes the problems high-dimensional. A key step in the overall design is to find a feasible trajectory using a randomized path planner, which typically requires a large fraction of the total computation time. This paper presents an improvement to the path planner that greatly reduces the solution time and enables the solution of very large problems. The primary modification is to change the function that connects two points, which is a key component of the randomized planner. The standard approach is to use a simple and fast function, which is changed here to one that is slower, but much more effective in linking two points. The new connection function finds a link between the points by minimizing a distance function to the target point with a feasible optimizer that accounts for the constraints. The examples presented demonstrate the significant speed improvement of the new planner and that the algorithm can solve problems with up to 16 spacecraft with numerous constraints.**

I. Introduction

Formation flying of multiple spacecraft is an enabling technology for many planned space science missions [1]. To achieve the full science objectives, these missions will typically require a complex initialization maneuver and/or formation reconfiguration maneuvers to reorient the fleet. These maneuvers consist of moving and rotating a group of N spacecraft to a desired target configuration while satisfying an arbitrary number of constraints. These constraints may consist of collision and plume impingement avoidance, and restrictions on the regions of space that certain spacecraft instruments must (and must not) point. Of course, it is also desirable to perform these maneuvers with the lowest possible fuel expenditure. This problem is particularly difficult due to the nonlinearity of the attitude dynamics, the non-convexity of some of the constraints, and the coupling between the positions and attitudes of all spacecraft. The non-convex constraints place the problem in the same class of generalized path planning problems that are well known to be NP-hard [2]. Additionally, the coupling between the positions and attitudes of the entire fleet in some of the pointing constraints requires that the trajectory design be solved as a single $6N$ DOF problem instead of N separate 6 DOF problems.

The spacecraft translation and attitude problems with constraints have each been the subject of extensive research with successful results [3], [4], [5], [6]. The combined translation and attitude problems have attracted less attention. Due to the complexity of the problem, existing solutions have been limited in the number of spacecraft and the complexity of the constraints [8], [9], [7]. For multiple spacecraft and coupling constraints, the problem is NP-Hard with $6N$ DOF, which would be impossible to solve by standard nonlinear optimization methods for reasonably sized problems.

Ref. [7] presented a solution for the spacecraft reconfiguration problem using randomized algorithms that have been successfully applied to robot path planning problems. The method in [7] is comprised of two stages – the first (*path planner*) generates a feasible solution to the spacecraft reconfiguration problem, and the second (*smoother*) improves this solution while maintaining feasibility. The path planning stage uses a variant of the Rapidly-exploring Random Tree developed by Lavalle [10]. The smoothing stage then uses a feasible nonlinear optimization algorithm to improve the cost of this initial solution (see Ref. [7] for further details).

The RRT used to solve the spacecraft reconfiguration problem is a variant of a recent algorithm that has been shown to be among the fastest path planners available [11]. Special care was taken to simplify the formulation of the path planning problem as much as possible and efficiently code the algorithm in C++. The experiments showed that the planner found solutions for problems with up to four spacecraft in approximately forty minutes. These are good results compared with previous solutions to similar problems, but better performance is desirable for real-time applications. For example, the smoothing step consists of a nonlinear optimization which is polynomial in nature and solutions are typically found in ten seconds in the worst case.

This paper presents an extension of the planner that decreases the solution times by approximately two orders of magnitude, thus enabling the solution of problems with up to sixteen spacecraft. The extension consists of replacing the function that generates a *simple trajectory* between two points in the RRT by a function that is more complex but has better performance. Most randomized planners use a simple

0-7803-9568-9/05/$20.00 ©2005 IEEE

function that generates a straight line or a simple rotation that stops after reaching the second point or hitting an obstacle, with the intention of generating as many connections and growing the random trees as fast as possible. However, there is also value in spending more time choosing better connections and thus generating a more efficient search tree. This trade-off between complexity and effectiveness appears in many problems (e.g., other path planning literature [15], [16], and basic optimization, such as first-order steepest descent versus the second order Newton optimization methods). Previous work on improving randomized path planners concentrated mostly in generating good random points. Instead, this paper concentrates on generating efficient connections between points for RRTs.

The connection approach used consists of a method similar to a gradient descent on artificial potential functions [12], [13]. Ref. [13] in particular uses potential functions for path planning, aided by a randomized algorithm to leave local minima, which are the methods combined in this article. However, the approach proposed here is the opposite, which is to aid a particular type of randomized path planner with gradient descent on a potential function. Since the obstacles in the spacecraft reconfiguration problem are constraints that can be written in the form $g(\boldsymbol{x}) \leq 0$, the gradient descent is posed as a nonlinear optimization problem and solved with a feasible solver. The intermediate solutions of the solver are then used as the trajectory connecting the points. The results show that this extension significantly improves the solution times from those reported in Ref. [7].

II. The Path Planner

A. Problem Formulation

The general reconfiguration problem consists of finding a trajectory, represented by the state and control inputs of N spacecraft, from time 0 to T. This state consists of

$$\boldsymbol{r}_i(t) \in \mathbb{R}^3, \quad \dot{\boldsymbol{r}}_i(t) \in \mathbb{R}^3 \tag{1}$$

$$\boldsymbol{q}_i(t) \in \mathbb{SO}^3, \quad \boldsymbol{w}_i(t) \in \mathbb{R}^3 \tag{2}$$

where $i \in 1 \ldots N$ indicates the spacecraft. $\boldsymbol{r}_i(t)$ is the position of its center, $\dot{\boldsymbol{r}}_i(t)$ its velocity, and $\boldsymbol{w}_i(t)$ its angular velocity, all measured with respect to a local inertially fixed frame. The attitude quaternion is $\boldsymbol{q}_i(t)$. The input consists of

$$\boldsymbol{T}_i(t) \in \mathbb{R}^3 \text{ and } \boldsymbol{M}_i(t) \in \mathbb{R}^3 \tag{3}$$

where $\boldsymbol{T}_i(t)$ are the forces and $\boldsymbol{M}_i(t)$ the moments. Let $\boldsymbol{x}_i(t)$ be a point of the trajectory of a single spacecraft at time t,

$$\boldsymbol{x}_i(t) = \left[\boldsymbol{r}_i(t), \dot{\boldsymbol{r}}_i(t), \boldsymbol{T}_i(t), \boldsymbol{q}_i(t), \boldsymbol{w}_i(t), \boldsymbol{M}_i(t)\right], \tag{4}$$

for $i \in 1 \ldots N$, and

$$\boldsymbol{x}(t) = [\cdots, \boldsymbol{x}_i(t), \cdots] \tag{5}$$

a point in the composite trajectories of all the spacecraft at time t.

For the deep space missions considered, the translational dynamics are approximated with a double integrator

$$\begin{bmatrix} \dot{\boldsymbol{r}}_i(t) \\ \ddot{\boldsymbol{r}}_i(t) \end{bmatrix} = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{r}_i(t) \\ \dot{\boldsymbol{r}}_i(t) \end{bmatrix} + \begin{bmatrix} 0 \\ I \end{bmatrix} \boldsymbol{T}_i(t) \tag{6}$$

and the attitude dynamics, in quaternion notation, are

$$\dot{\boldsymbol{q}}_i(t) = \frac{1}{2}\Omega_i(t)\boldsymbol{q}_i(t) \tag{7}$$

$$J\dot{\boldsymbol{w}}_i(t) = -\boldsymbol{w}_i(t) \times (J\boldsymbol{w}_i(t)) + \boldsymbol{M}_i(t) \tag{8}$$

where

$$\Omega_i(t) = \begin{bmatrix} 0 & w_{i3}(t) & -w_{i2}(t) & w_{i1}(t) \\ -w_{i3}(t) & 0 & w_{i1}(t) & w_{i2}(t) \\ w_{i2}(t) & -w_{i1}(t) & 0 & w_{i3}(t) \\ -w_{i1}(t) & -w_{i2}(t) & -w_{i3}(t) & 0 \end{bmatrix} \tag{9}$$

and J is the inertia matrix, which is considered to be the same for all spacecraft for simplicity. The collision avoidance constraints are

$$\|\boldsymbol{r}_i(t) - \boldsymbol{r}_j(t)\| \geq R \tag{10}$$

for $i, j \in 1 \ldots N$, $i \neq j$, and R is the minimum distance allowed between the centers of spacecraft. The *absolute stay outside* pointing constraints are given by

$$\boldsymbol{z}_k^T \boldsymbol{y}_k \leq \cos\theta_k \tag{11}$$

where $k \in 1 \ldots N_c$ identifies a constraint. This condition ensures that the spacecraft vector $\boldsymbol{y}_k$ remains at an angle greater than $\theta_k \in [0, \pi]$ from the inertial vector $\boldsymbol{z}_k$. The vector $\boldsymbol{y}_k$ is the representation in the inertial coordinate frame of the body vector $\boldsymbol{y}_{kB}$. The transformation of coordinates is given by

$$\boldsymbol{y}_k = \boldsymbol{y}_{kB} - 2(\boldsymbol{q}_{i0}^T \boldsymbol{q}_{i0})\boldsymbol{y}_{kB} + 2(\boldsymbol{q}_{i0}^T \boldsymbol{y}_{kB})\boldsymbol{q}_{i0} - 2q_{i4}(\boldsymbol{y}_{kB} \times \boldsymbol{q}_{i0}) \tag{12}$$

where $\boldsymbol{q}_{i0}(t)$ and $q_{i4}(t)$ are defined as

$$\boldsymbol{q}_i(t) = [q_{i1}(t), q_{i2}(t), q_{i3}(t), q_{i4}(t)]^T = [\boldsymbol{q}_{i0}(t), q_{i4}(t)]^T$$

The *absolute stay inside* pointing constraints only differ by the sign of the equation

$$\boldsymbol{z}_k^T \boldsymbol{y}_k \geq \cos\theta_k \tag{13}$$

The inter-spacecraft *relative stay outside* pointing constraints are given by

$$\hat{\boldsymbol{r}}_{ij}^T \boldsymbol{y}_k \leq \cos\theta_k \tag{14}$$

where $\boldsymbol{y}_k$ and θ_k are as above, and $\hat{\boldsymbol{r}}_{ij}(t) = (\boldsymbol{r}_j(t) - \boldsymbol{r}_i(t))/(\|\boldsymbol{r}_j(t) - \boldsymbol{r}_i(t)\|)$ is the unit vector that points from spacecraft i to spacecraft j. Similarly, the *relative stay inside* pointing constraints are

$$\hat{\boldsymbol{r}}_{ij}^T \boldsymbol{y}_k \geq \cos\theta_k \tag{15}$$

B. The Planner

The randomized path planning algorithm is reproduced below to aid in the presentation of this paper

```
RRT-BIDIRECTIONAL(x_i, x_f)
 1  T_a.init(x_i); T_b.init(x_f)
 2  for j ← 1 to K
 3       do x_n ← NEAREST(T_a, α(j))
 4          x_s ← CONNECT(x_n, α(j))
 5          if x_s ≠ x_n
 6            then T_a.add-vertex(x_s)
 7                 T_a.add-edge(x_n, x_s)
 8                 x'_n ← NEAREST(T_b, x_s)
 9                 x'_s ← CONNECT(x'_n, x_s)
10                 if x'_s ≠ x'_n
11                   then T_b.add-vertex(x'_s)
12                        T_b.add-edge(x'_n, x'_s)
13                 if x'_s = x_s
14                   then return Solution
15          SWAP(T_a, T_b)
16  return Failure
```

In this algorithm, T_a and T_b represent trees with a composite trajectory point $\boldsymbol{x}$ at each node (Eq. 5). The points $\boldsymbol{x}$ at the nodes are considered at rest, so the only relevant information in these points are positions and attitudes. The two trees T_a and T_b are started from the initial and final points of the desired trajectory. At each iteration, a random point is obtained by $\alpha(j)$, and its NEAREST point is found from the tree. Here NEAREST$(T_a, \alpha(j))$ finds the point in the set of nodes of T_a with minimum distance to the point $\alpha(j)$, and distance is defined as

$$\mathrm{d}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \sum_{i=1}^{N} \|\boldsymbol{r}_{1,i} - \boldsymbol{r}_{2,i}\|^2 + K_a \angle(\boldsymbol{q}_{1,i}, \boldsymbol{q}_{2,i})^2 \quad (16)$$

where $\angle(\boldsymbol{q}_{1,i}, \boldsymbol{q}_{2,i})$ is the angle of an eigen-axis rotation between attitude $\boldsymbol{q}_{1,i}$ and $\boldsymbol{q}_{2,i}$ for spacecraft i, and K_a is a scale factor that relates translation distance and rotation angle. In the scenarios presented later, $K_a = 0.1$.

Continuing with the algorithm, CONNECT$(\boldsymbol{x}_n, \alpha(j))$ finds the last valid configuration in the "direct motion" from $\boldsymbol{x}_n$ to $\alpha(j)$. In our implementation, a *direct motion* is a rest-to-rest straight line translation and eigen-axis rotation of each spacecraft

$$\left.\begin{array}{r} \mathrm{trans}(\boldsymbol{r}_{1,i}, \boldsymbol{r}_{2,i}; t) = \boldsymbol{r}_{1,i} + t(\boldsymbol{r}_{2,i} - \boldsymbol{r}_{1,i}) \\ \mathrm{rot}(\boldsymbol{q}_{1,i}, \boldsymbol{q}_{2,i}; t) = \boldsymbol{q}_{1,i} \cdot (\boldsymbol{q}_{1,i}^{-1} \cdot \boldsymbol{q}_{2,i})^t \end{array}\right\} \begin{array}{c} \forall i \\ t \in [0,1] \end{array} \quad (17)$$

where rot is a quaternion interpolation.

If a new node is successfully found by the direct motion, then a branch to this node is added to the tree, and a similar attempt is made to connect the opposite tree to the new node. If the attempt succeeds the algorithm stops with a solution.

The output of the algorithms consists of a sequence of points, from the initial point $\boldsymbol{x}_i$ to the desired target point $\boldsymbol{x}_f$, where each spacecraft is at rest, their states are given by position and attitude values, and each point can reach the next by a direct motion that is guaranteed to satisfy all the constraints.

III. THE CONNECTION FUNCTION

The solution method discussed previously represents a standard application of randomized algorithms for path planning. Randomized algorithms perform well in general, but they have been shown to perform poorly for problems that contain difficult features such as "narrow passages" (i.e., the free space is long and narrow). Because the randomized algorithms only generate a few sample point in those areas, finding a path that connects them can be very difficult [14]. These difficult features have generated a large amount of interest, and as a result, numerous methods have been developed with many successful results, including the bridge test and Gaussian sampling, among many others [15], [16]. The common idea behind these methods is that they spend more effort in generating the random sample points. Instead of just growing a large tree by generating many blind samples, information about the environment is used to focus the effort on generating samples in the difficult areas, leading to a more efficient tree growth. These methods of finding better samples do not work well with RRT's because a key characteristic of RRT's is that the samples must be distributed evenly over the solution space [10].

This paper uses a similar philosophy applied specifically to Rapidly-exploring Random Trees algorithms for the spacecraft reconfiguration problem. In particular this paper presents a new technique that exerts more computational effort in the step of *connecting* two points of the tree. This step is a basic component of most randomized path planners, including RRT's. The standard implementation is the CONNECT function presented in the previous section, which consists of a simple function that tries to connect two points directly, but stops as soon as it finds an obstacle. The main idea behind using a simple connection function is that the strength of the algorithm comes from its ability to generate as many connections as possible. However, with a slight increase in computational effort, this connection function can be extended to use some information about the obstacles to increase the chance of reaching the target point.

This paper proposes using artificial potential functions as one way of including information about the obstacles. A connection function based on potential functions is more flexible than a direct connection because it takes into account the obstacles, but it is still fast compared to a full path planner. The artificial potential function consists of a continuous function that decreases monotonically in the direction of the goal, (e.g., a *distance* function). Thus, following the negative gradient of the function generates a path to the goal. Obstacles are then introduced as high *peaks* in the potential function, so that they effectively repel any search that follows the gradient. Ideally it should be possible to find a path descending to the goal while avoiding obstacles from every point in space. In practice however the gradient descent method does not always work, because the sum of the distance function plus the high peaks create local minima in the potential field [17]. Another difficulty with this method is that some path planning problems do not

have an obvious formulation with potential functions, or have functions with too many singular points or local minima. However, for a path planning problem without differential constraints and with a moderate number of obstacles that are easy to formulate as potential sources (e.g., constraints such as $g(\boldsymbol{x}) \leq 0$), the new method proposed in this paper significantly improves the performance of the Rapidly-exploring Random Trees.

The method consists of replacing the CONNECT function in the path planner by a potential function based on a distance metric $d(\boldsymbol{x}_1, \boldsymbol{x}_2)$ with the obstacles and other constraints represented by inequality and equality constraints of the form

$$g(\boldsymbol{x}) \leq 0 \tag{18}$$

$$h(\boldsymbol{x}) = 0 \tag{19}$$

For the spacecraft reconfiguration problem, the distance metric is the same as Eq. 16, with the collision avoidance and pointing constraints described by Eqs. 10 to 15. The search attempts to find a sequence of feasible points that decreases the distance to the target point. This search is posed as a nonlinear optimization problem, and is solved with a feasible sequential optimization method. Since the optimization method is feasible, the intermediate solutions provide a valid trajectory. Each new intermediate solution should also be restricted to be within a certain distance of the previous one, to ensure that the trajectory between them is also valid. The potential connection function then becomes

POTENTIAL-CONNECT$(\boldsymbol{x}_i, \boldsymbol{x}_f)$
1 **for** $j \leftarrow 1$ **to** K
2 **do** Solve nonlinear program:
$\min_{d\boldsymbol{x}} d(\boldsymbol{x} + d\boldsymbol{x}, \boldsymbol{x}_f)$
subject to
$h(\boldsymbol{x} + d\boldsymbol{x}) = 0$
$g(\boldsymbol{x} + d\boldsymbol{x}) \leq 0$
$\|d\boldsymbol{x}\| \leq \epsilon$
▷ End of nonlinear program
3 $\boldsymbol{x} \leftarrow \boldsymbol{x} + d\boldsymbol{x}$ ▷ Update new point
4 **return** $\boldsymbol{x}$

In this function $\boldsymbol{x}_i$ and $\boldsymbol{x}_f$ are initial and goal points, $\boldsymbol{x}$ is the temporary point that moves towards the goal by $d\boldsymbol{x}$ increments. The norm of the $d\boldsymbol{x}$ increments is limited to be less than ϵ to ensure the feasibility of the trajectory between two adjacent points.

The numerical experiments in this paper were done with a custom sequential linear programming method [7]. This method is based on solving a sequence of linear programs with linearized constraints, but there are several feasible sequential optimization methods that could be used here [18].

IV. SIMULATION RESULTS

This section presents examples of different complexity. The first example illustrates the difference between the direct connection function versus the new connection. The following examples compare the performance of the new connection connection function against results from a previous paper [7]. The last set of examples shows the performance of the algorithm handling larger formations of 8 and 16 spacecraft.

A. Simple 2D Path Planning Example

Figures 1 to 4 illustrate the difference between the approaches for a simple example. This is a 2D path planning problem with ellipsoidal obstacles. The distance metric is

$$d(\boldsymbol{r}_1, \boldsymbol{r}_2) = \|\boldsymbol{r}_1 - \boldsymbol{r}_2\|^2 \tag{20}$$

and the ellipsoidal obstacles are introduced as the inequalities

$$(\boldsymbol{r}_x - e_{i,x})^2/A_i^2 + (\boldsymbol{r}_y - e_{i,y})^2/B_i^2 \geq 1 \tag{21}$$

where $e_{i,x}$ and $e_{i,y}$ are the coordinates of the center of ellipse i, and A_i and B_i the lengths of its x and y axis. This problem has no equality constraints of type $h(\boldsymbol{x}) = 0$.

The figures show two different configurations. Configuration A includes many obstacles and configuration B shows an example of a "narrow passage". Figures 1 and 2 show the bidirectional random trees created using the direct connection function. Note in figure 2 that the tree must create several zigzagging branches to clear the narrow passage. Figures 3 and 4 show the bidirectional random trees using the new connection function. These figures show that the RRT's only need one or a few iterations to solve the problem, as was the case with all the experiments. The main characteristic of this new connection function is that it borders the obstacles while still decreasing the distance function, which can be seen in the figures.

Although these examples highlight the benefits of the potential function approach, it is possible to create examples where this type of function does not perform as well as shown here. However, in general this method performs better than using a simple direct connection, and also performs well for the spacecraft reconfiguration problem, as shown in the next section.

B. Comparison with Previous Results

The following examples compare the performance of the path planner proposed in this paper with the one proposed in a previous paper [7]. These examples have 3–4 spacecraft and relative pointing constraints. Full descriptions of the example are in Ref. [7].

1) *Three spacecraft:* Spacecraft 1 and 2 are required to switch positions. Relative pointing constraints couple the attitudes and positions of all the spacecraft, and an absolute pointing constraint is added to better show this coupling.

2) *Change of formation shape with 4 spacecraft:* A square formation moves to a tetrahedral shape. Spacecraft 1 to 3 must point at each other and toward Spacecraft 4.

3) *Formation rotation with 4 spacecraft:* The same tetrahedral formation of four spacecraft is required to rotate, with the same constraints.

4) *Formation reflection with 4 spacecraft:* Starting with the same tetrahedral configuration and constraints as before, Spacecraft 4 is required to cross the plane of the other three spacecraft and switch to the other side.

Table I shows that with the change in the connection function, the path planner solves the problems more than 100 times faster. These problems were carefully chosen for

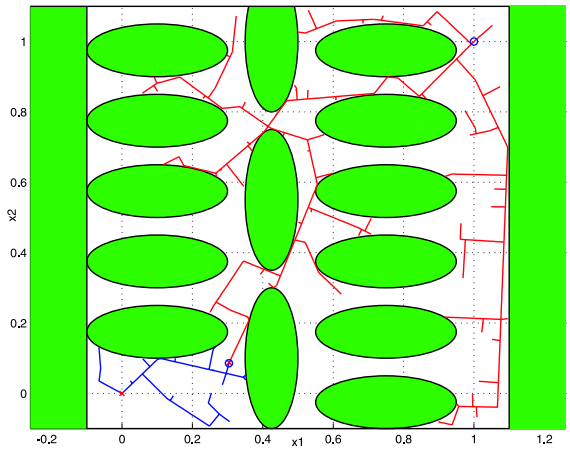

Fig. 1. Example A: RRT with direct connection on obstacle field

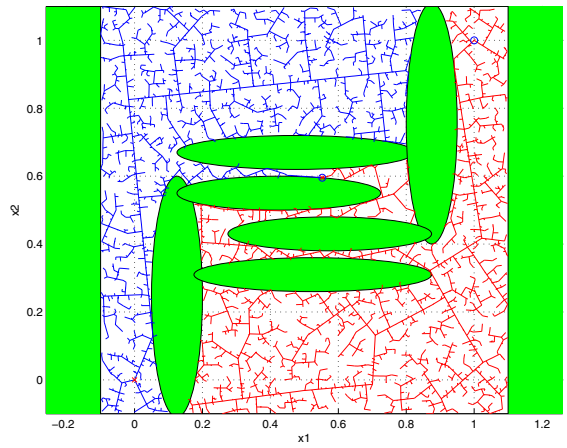

Fig. 2. Example B: RRT with direct connection on narrow passage

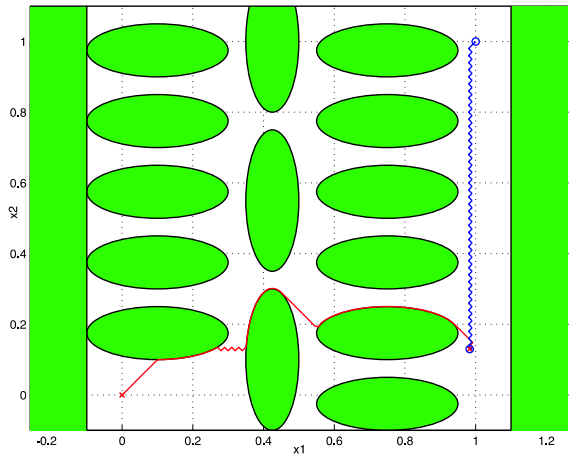

Fig. 3. Example A: RRT with potential connection on obstacle field

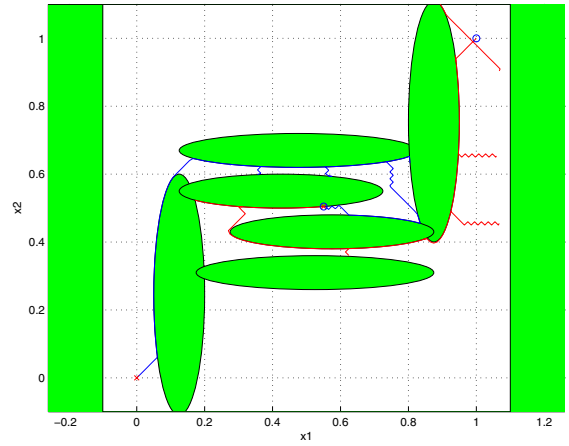

Fig. 4. Example B: RRT with potential connection on narrow passage

TABLE I. Comparison with previous results for small configurations.

Example	Previous Approach[7] Iterations	Time (s)	New Approach Iterations	Time (s)
3 s/c	336	652	2	3
4 s/c shape change	386	902	1	3
4 s/c rotation	51	27	2	2
4 s/c reflection	309	492	1	3

the previous paper [7] because they are difficult examples of reconfiguration maneuvers that might be required for future formation flying missions.

C. New Results with Larger Configurations

The following examples were tested with configurations of 8 and 16 spacecraft. They include relative pointing constraints coupling the degrees of freedom of all the spacecraft.
1) *Crossing at the center:* The spacecraft start at the corners of an imaginary cube (or two concentric cubes for the case of 16 spacecraft) facing each other in pairs, and each pair is then required to switch places. Meanwhile, they must point their X body vector at each other to within $40°$. This maneuver is chosen so that all the spacecraft would coincide at the origin if they were to move in straight lines. The result is a coordinated movement of spacecraft avoiding each other while pointing at their respective pairs, which can be seen in figures 5 and 6. The planned trajectories have been smoothed for these figures.

2) *Rotation of star-like configuration:* A configuration consisting of a spacecraft in the center and the remainder in concentric rings (two rings for eight spacecraft, three rings for sixteen spacecraft) must rotate $90°$. The spacecraft must always point the X body axis (within $40°$) towards the spacecraft in the ring that follows counter-clockwise. It should also point a body vector towards the spacecraft in the center (within $40°$).
3) *Random diagonal configuration to star configuration:* The spacecraft start close to a diagonal configuration, pointing the X body axis at the spacecraft that goes next in the star configuration (similar to example 2), and they are required to move to the star configuration. To remove the symmetry in the example, the initial positions and attitudes were randomly perturbed while still satisfying the constraints.

The large examples were also chosen so that the pointing constraints couple the position and attitudes of all the spacecraft. As a result, the path planning problems with 8 and 16 spacecraft have 48 DOF and 96 DOF respectively. These are large difficult problems in which the constraints are active over most of the trajectories. Nevertheless, Table II shows that the new RRT-based path planner with the potential connection function solves these large examples in reasonable times, which was not possible with the basic RRT [7]. The fast solution times for the examples with 8

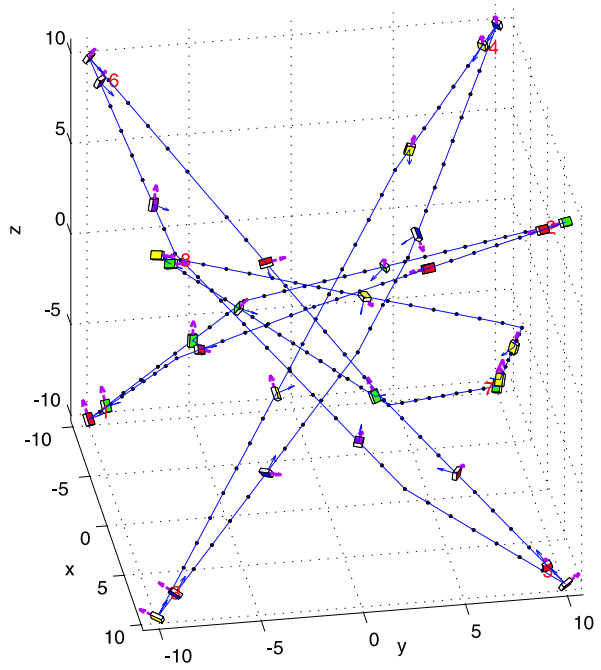

Fig. 5. 8 spacecraft start at the corners of a cube and switch places

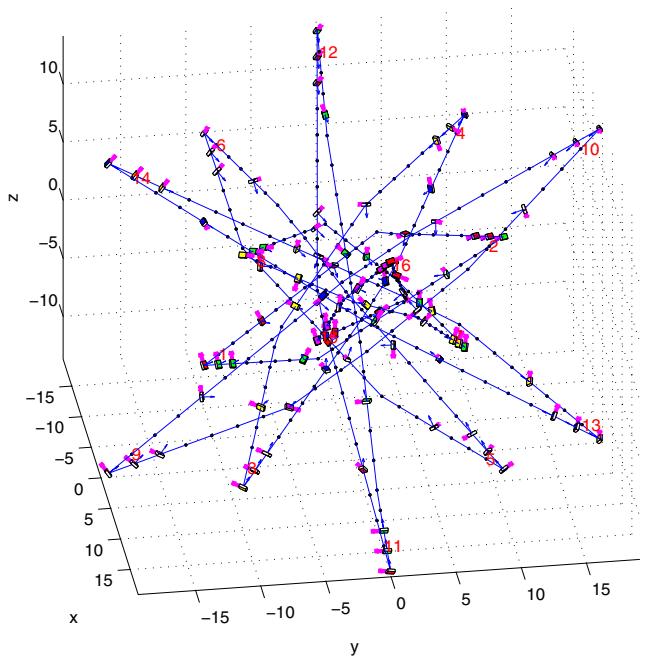

Fig. 6. 16 spacecraft start at the corners of 2 cubes and switch places

TABLE II. New results for larger configurations.

Problem	8 Spacecraft Iter.	8 Spacecraft Time (sec)	16 Spacecraft Iter.	16 Spacecraft Time (sec)
Crossing at Center	3	29	1	307
Rotation of Star	3	9	23	349
Diagonal to Star	1	17	3	388

spacecraft or less shows the clear improvement in speed, and also suggest that the planner can be used online to solve spacecraft reconfiguration problems, which was not possible with the basic RRT [7].

V. Conclusions

This paper presented an improved path planner that significantly reduces the solution times for spacecraft reconfiguration maneuver problems. The primary improvement was to change the function that connects two points, a key element of the randomized planner, from a simple and fast function, to one that is slower but more effective in linking two points. The new approach uses a potential function in a feasible optimizer to find a link between the two points while accounting for the constraints. Several examples were presented to show the increase in speed of the new planner and that the algorithm can solve problems with up to sixteen spacecraft with numerous constraints.

References

[1] J. Leitner, F. Bauer, D. Folta, M. Moreau, R. Carpenter, and J. How, "Distributed Spacecraft Systems Develop New GPS Capabilities," in *GPS World: Formation Flight in Space* Feb. 2002.

[2] J. H. Reif, "Complexity of the Mover's Problem and Generalizations," *20th Annual IEEE Symposium on Foundations of Computer Science,* San Juan, Puerto Rico, October 1979, p. 421–427.

[3] A. Richards, T. Schouwenaars, J. P. How, and E. Feron, "Spacecraft Trajectory Planning With Collision and Plume Avoidance Using Mixed-Integer Linear Programming," AIAA *Journal of Guidance, Control, and Dynamics,* Vol. 25, No. 4, p. 755–765, Aug. 2002.

[4] E. Frazzoli, "Quasi-Random Algorithms for Real-Time Spacecraft Motion Planning and Coordination" *International Astronautical Congress*, Houston, TX, 2002.

[5] H. Hablani, "Attitude Commands Avoiding Bright Objects and Maintaining Communication with Ground Station," AIAA *Journal of Guidance, Control, and Dynamics,* Vol. 22, No. 6, p. 759–767.

[6] E. Frazzoli, M. Dahleh, E. Feron, R. Kornfeld, "A Randomized Attitude Slew Planning Algorithm For Autonomous Spacecraft," *AIAA Guidance, Navigation and Control Conf.*, AIAA 2001–4155.

[7] I. M. Garcia and J. P. How, "Trajectory Optimization for Satellite Reconfiguration Maneuvers with Position and Attitude Constraints" to appear in the IEEE *ACC*, June 2005

[8] D. P. Scharf, and S. R. Ploen, F. Y. Hadaegh, J. A. Keim, and L. H. Phan, "Guaranteed Initialization of Distributed Spacecraft Formations," *AIAA Guidance, Navigation and Control Conference*, AIAA 2003–5590, August 2003.

[9] J. Phillips, L. E. Kavraki, and N. Bedrosian , "Probabilistic Optimization Applied to Spacecraft Rendezvous and Docking," In *13th American Astronomical Society/AIAA - Space Flight Mechanics Meeting,*, Puerto Rico, February 2003.

[10] S. M. LaValle and J. J. Kuffner, "Randomized Kinodynamic Planning," *Int. J. of Robotics Research,* Vol. 20, No. 5, p. 378–400, May 2001.

[11] C. M. Clark, *Dynamic Robot Networks: A Coordination Platform for Multi-Robot Systems,* PhD Thesis, June 2004.

[12] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Robotics Research*, (5), no. 1, p. 90–98, 1986.

[13] J. Barraquand, B. Langlois, and J.C. Latombe, "Numerical potential field techniques for robot path planning," IEEE *Transactions on Systems, Man and Cybernetics,* Vol. 22, No. 2, pp.224–241, 1992.

[14] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. Overmars, "Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces," In IEEE *Transactions on Robotics and Automation,* Vol. 12, No. 4, p. 566–580, 1996.

[15] D. Hsu, T. Jian, J. Reif, and Z. Sun, "The Bridge Test for Sampling Narrow Passages with Probabilistic Roadmap Planners," In Proceedings of the IEEE *Int. Conf. on Robotics and Automation,* Taipei, 2003.

[16] V. Boor, M.H. Overmars, and A.F. van der Stappen, "The Gaussian sampling strategy for probabilistic roadmap planners," Proc. of the IEEE *Int. Conf. on Robotics and Automation,* p. 1018–1023, 1999.

[17] J. C. Latombe, "Robot Motion Planning", Kluwer Academic Publishers, Boston, MA, 1991.

[18] C.T. Lawrence and A.L. Tits, "A Computationally Efficient Feasible Sequential Quadratic Programming Algorithm," SIAM J. Optimization, Vol. 11, No. 4, 2001, p. 1092-1118