Proceedings of the 2005 IEEE
International Conference on Robotics and Automation
Barcelona, Spain, April 2005

Path Planning for Permutation-Invariant Multi-Robot Formations

Stephen Kloder **Seth Hutchinson**

The Beckman Institute for Advanced Science and Technology
University of Illinois at Urbana-Champaign
Urbana, IL 61801
Email: {kloder, seth}@uiuc.edu

***Abstract*— In this paper we demonstrate path planning for our formation space that represents permutation-invariant multi-robot formations. Earlier methods generally pre-assign roles for each individual robot, rely on local planning and behaviors to build emergent behaviors, or give robots implicit constraints to meet. Our method first directly plans the formation as a set, and only afterwards determines which robot takes which role.**

To build our representation of this formation space, we make use of a property of complex polynomials: they are unchanged by permutations of their roots. Thus we build a characteristic polynomial whose roots are the robot locations, and use its coefficients as a representation of the formation. Mappings between work spaces and formation spaces amount to building and solving polynomials.

In this paper, we construct an efficient obstacle collision detector, and use it in a local planner. From this we construct a basic roadmap planner. We thus demonstrate that our polynomial-based representation can be used for effective permutation-invariant formation planning.

I. Introduction

Multi-robot planning is a useful and developing subfield of robot planning. Multiple coordinated robots have been used successfully in a variety of applications including surveillance and monitoring [1], localization and exploration [2], satellite arrangement [3], search and rescue [4], and object manipulation and transportation [5], [6], [7].

Multi-robot motion planning approaches are distinguished by their levels of centralization. In fully centralized methods (e.g. [8], [9], [10]), a single planner plans all robots' motions simultaneously. This often makes use of a joint configuration space that represents the configurations of all robots [11]. Slightly less centralized is the *decoupled* approach [12], [13], [14], where each robot plans its own path, but a single centralized planner coordinates the robots by dictating how each robot follows its path. Significantly less centralized are *emergent behaviors* methods [15], [7], where each robot follows local behaviors that depend on relative locations of nearby robots. In emergent behavior methods, a centralized planner determines the general motion of the robots, and the structure that they maintain. Fully decentralized methods are usually *coverage* methods [16], [17], [18], [19], with goals of maximizing the total region visible by robots' sensors, or of minimizing the probability of passing undetected through a region. In coverage methods, all robots plan independently, but consider each others' locations when moving.

In more centralized methods, robot roles and relationships are usually predefined. The methods that do not predefine roles and relationships are found on the decentralized end of the spectrum, e.g. blanket coverage methods, and a few emergent behavior methods like [20]. However, these methods do not control their formations directly, but set constraints and move until they are met. This usually limits feasible goals to symmetric formations. Our new method plans formations in a direct fashion like the more centralized methods, while still being flexible enough to allow changing roles and relationships like less centralized methods.

We introduced our representation for permutation-invariant formations and a simple planner in [21] for planar robots. Our method uses complex polynomials to represent multi-robot formations. We demonstrated basic planning in obstacle-free environments, and showed some of its properties. In this paper we demonstrate collision detection, and show how it can be used to implement roadmap planning in the formation space.

This paper is laid out as follows: Section II describes the representation and basic planning, Section III describes collision detection, Section IV describes roadmap results, and Section V describes proposed future work.

II. Representation and Basic Planning

A. A Permutation-invariant representation of formations

The configuration space of a labeled multirobot formation is generally an ordered list of configurations, one for each robot. However, we are building a configuration space for *unlabeled* formations, where exchanging two robots does not change the configuration. We call this a *formation space*, and write $FS[X]^n = X^n/S_n$, where n is the number of robots, X is the configuration space of one robot, and S_n is the symmetric group of permutations of n elements. In this quotient space, $(x_1, x_2, \ldots, x_n)$ and $(x'_1, x'_2, \ldots, x'_n)$ are identified iff they are permutations of each other. We shall sometimes refer to $FS[X]^n$ as FS^n for short if X is known.

In this paper we focus on translating planar robots, whose configurations can be represented as $(x, y) \in \mathbb{R}^2$. We represent the configuration of each robot as $z = x + yi \in \mathbb{C}$, so we can construct $FS[\mathbb{C}]^n$ with complex polynomials. Therefore the robots' *workspace* $\mathcal{W}$, the space in which the individual robots move, is represented by the complex plane $\mathbb{C}$. For n robots at locations $z_1, \ldots, z_n$, we define the polynomial

$$\begin{aligned} P(\lambda) &= (\lambda - z_1)(\lambda - z_2)\dots(\lambda - z_n) \\ &= \lambda^n + a_1\lambda^{n-1} + \dots + a_{n-1}\lambda + a_n \end{aligned} \quad (1)$$

The complex coefficients $a_1, \dots, a_n$ can be used to define a permutation invariant formation, or simply a *formation*. For any configuration $Z = \{z_1, \dots, z_n\}$ of the multirobot system, (1) defines a unique formation $\mathbf{a} = (a_1, \dots, a_n)$. Throughout this paper, we will refer to sets of robot configurations, like Z, as *multirobot configurations*, while the n-entry vectors like $\mathbf{a}$ that specify these multirobot configurations will be referred to as *formations*.

Equation (1) defines the mapping $f : \mathbb{C}^n/S_n \to \mathbb{C}^n$ to be $Z \longmapsto \mathbf{a}$. Since the roots of a polynomial vary continuously as a function of the coefficients, f^{-1}, which is $\mathbf{a} \longmapsto Z$, is a continuous mapping from a formation to a multirobot configuration. Since f is a bijection (Fundamental Theorem of Algebra), and both f and f^{-1} are continuous (see [22]), f is a homeomorphism, so $FS[\mathbb{C}]^n \cong \mathbb{C}^n$. We thus define our formation space as $FS[\mathbb{C}]^n = \mathbb{C}^n$, the set of all possible n-tuples of complex polynomial coefficients.

In this paper, we will use $\mathbf{a}$ and $\mathbf{b}$ to represent formations; each formation is a vector of complex coefficients of a complex polynomial $P(\lambda)$, as in (1).

B. Straight-Line Planning

For local planning, we use a "straight-line" planner in the formation space. To plan from $\mathbf{a} = (a_1, \dots, a_n)$ to $\mathbf{b} = (b_1, \dots, b_n)$, follow the path $\ell(t)$ in the formation space, where

$$\ell(t) = (1-t)(a_1, \dots, a_n) + t(b_1, \dots, b_n) \quad (2)$$

in which $t \in [0, 1]$ parameterizes the path. Therefore, at time t, the robots will be at the roots of the polynomial

$$\lambda^n + \sum_{k=1}^{n} [(1-t)a_k + tb_k]\lambda^{n-k} = 0. \quad (3)$$

In [21], we showed that if both the initial and goal configurations translate, scale, rotate, and reflect by the same amount, the resulting path will be transformed accordingly. This means that the path, relative to its endpoints, is independent of scale, orientation, origin location, or handedness.

For example, Figure 1 illustrates two straight-line paths in the formation space The figures show the workspace paths of the robots. The paths on the right were generated by taking the start and goal points from the figure on the left; translating, scaling, rotating, and reflecting the endpoints; and moving from the new start to the new goal using straight-line planning. The resulting paths are the same paths as if the original paths were also translated, scaled, rotated, and reflected the same way as the endpoints.

III. Obstacles

For permutation-invariant motion planning to be useful, it must be able to test for collisions with obstacles. To do this, we will define the *swept volume* of a path $V(\mathbf{a}, \mathbf{b})$ to be the locus of points in the workspace traversed by the robots while following the formation space path $\ell(t)$ from equation (2). Formally:

$$V(\mathbf{a}, \mathbf{b}) = \left\{ z \in \mathbb{C} \,\middle|\, z \in f^{-1}((1-t)\mathbf{a} + t\mathbf{b}), t \in [0,1] \right\},$$

i.e. $V(\mathbf{a}, \mathbf{b})$ is the set of all roots of $\ell(t)$ for all values of $t \in [0, 1]$. Given polynomial configurations $\mathbf{a}$ and $\mathbf{b}$, and an obstacle region $\mathcal{O} \subset \mathcal{W}$, the straight-line path in the formation space from formation $\mathbf{a}$ to formation $\mathbf{b}$ generates a collision iff $V(\mathbf{a}, \mathbf{b}) \cap \mathcal{O} \neq \emptyset$. In this section, we will first show how to check collisions for point obstacles. We will then extend this to line segment obstacles. Once we can check for collisions with line segments, we can check for collisions with any polygonal obstacle.

A. Point Obstacles

Consider a point obstacle with coordinates z_c. For a given formation $\mathbf{a}$, a collision occurs if z_c is a root of (1), i.e. if some robot is coincident with the point obstacle. This can be determined simply by evaluating $P(z_c)$. There is a collision iff $P(z_c) = 0$. If we define the $\hat{}$ operator such that:

$$\hat{\mathbf{z}} = \left(z^{n-1}, z^{n-2}, \dots, z, 1\right), \quad (4)$$

we can write $P(z_c) = 0$ concisely as

$$\mathbf{a} \cdot \hat{\mathbf{z}}_\mathbf{c} = -z_c^n, \quad (5)$$

Since (5) is linear in the formation coefficients, the obstacle point (or more particularly, the locus of formations that correspond to at least one robot at z_c) corresponds to a $2n-2$ dimension hyperplane in the formation space.

Testing a single formation for collision is straightforward; testing for a path in the formation space requires a little more work. To check $V(\mathbf{a}, \mathbf{b})$ for collision with z_c, we must essentially evaluate (5) at all formations along the path.

Theorem 1: For any $\mathbf{a}, \mathbf{b} \in \mathbb{C}^n$, define function

$$\tau(z_c) = \frac{z_c^n + \mathbf{a} \cdot \hat{\mathbf{z}}_\mathbf{c}}{(\mathbf{a} - \mathbf{b}) \cdot \hat{\mathbf{z}}_\mathbf{c}}. \quad (6)$$

$V(\mathbf{a}, \mathbf{b})$ collides with a point obstacle at $z_c \in \mathbb{C}$ iff $\tau(z_c) \in \mathbb{R}$ and $\tau(z_c) \in [0, 1]$.

Proof: Using (2) to give formations as a function of t, we write (5) as

$$\ell(t) \cdot \hat{\mathbf{z}}_\mathbf{c} = -z_c^n \quad (7)$$

$$(1-t)\mathbf{a} \cdot \hat{\mathbf{z}}_\mathbf{c} + t\mathbf{b} \cdot \hat{\mathbf{z}}_\mathbf{c} = -z_c^n. \quad (8)$$

If (8) is satisfied for any $t \in [0, 1]$, then at least one robot collides with the obstacle at z_c. Solving for t we obtain

$$t = \frac{z_c^n + \mathbf{a} \cdot \hat{\mathbf{z}}_\mathbf{c}}{(\mathbf{a} - \mathbf{b}) \cdot \hat{\mathbf{z}}_\mathbf{c}} = \tau(z_c). \quad (9)$$

In order for there to be a collision with z_c, the t given by (9) must be real with $0 \leq t \leq 1$. ■

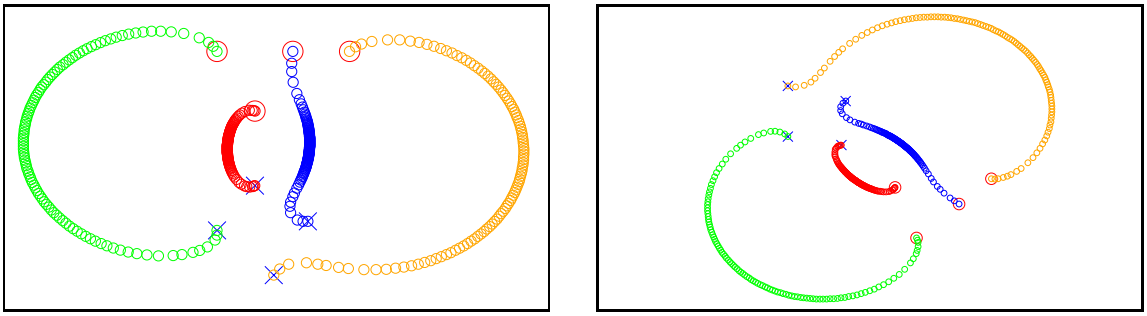

Fig. 1. Before and after: translation, scaling, rotation, and reflection

B. Horizontal Segment Obstacles

We showed in [21] that if the start and goal are translated and rotated, the paths transform accordingly. Since any edge can be rotated and translated onto the real axis, we can rotate and translate $\mathbf{a}$ and $\mathbf{b}$ the same way. If $V(\mathbf{a},\mathbf{b})$ collides with an edge, then the transformed path will collide with a subset of the real axis. This means we can test collision with any line segment if we can test for collision with the real axis.

Theorem 2: For any $\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{s} \in \mathbb{R}^n$, $V(\mathbf{p}+\mathbf{q}i,\mathbf{r}+\mathbf{s}i)$ collides with the real axis iff there exists an $x \in \mathbb{R}$ such that x is a root of the polynomial $\Omega(x) = \sum_{m=1}^{2n} \omega_m x^{2n-m}$, where ω_m is given in Figure 2, and $\tau(x) \in [0,1]$.

Proof: Set $\mathbf{a} = \mathbf{p}+\mathbf{q}i$ and $\mathbf{b} = \mathbf{r}+\mathbf{s}i$. We use the results from the previous section to test for collision with the real axis. With point obstacles, z_c was fixed, so we tested z_c in (9) and determined whether t was real and in $[0,1]$. Here, however, z_c can be any point on the edge, so we search for a z_c that makes t real and in $[0,1]$. Substituting $x \in \mathbb{R}$ for z_c in (9), we obtain

$$t = \tau(x) = \frac{x^n + \mathbf{a}\cdot\hat{\mathbf{x}}}{(\mathbf{a}-\mathbf{b})\cdot\hat{\mathbf{x}}}. \tag{10}$$

If there is some x such that $\tau(x)$ is real and between 0 and 1, then $V(\mathbf{a},\mathbf{b})$ intersects the real axis at x.

To make this determination, first find the set of values for x such that $\tau(x)$ is real. Since (10) is a ratio of two complex numbers, we make the denominator real, and focus on the numerator. With $\overline{z}$ defined as the complex conjugate of z, we transform (10) as

$$\begin{aligned}
\tau(x) &= \left(\frac{x^n + \mathbf{a}\cdot\hat{\mathbf{x}}}{(\mathbf{a}-\mathbf{b})\cdot\hat{\mathbf{x}}}\right)\left(\frac{\overline{(\mathbf{a}-\mathbf{b})}\cdot\hat{\mathbf{x}}}{\overline{(\mathbf{a}-\mathbf{b})}\cdot\hat{\mathbf{x}}}\right) && (11)\\
&= \frac{(x^n + \mathbf{a}\cdot\hat{\mathbf{x}})\left[\overline{(\mathbf{a}-\mathbf{b})}\cdot\hat{\mathbf{x}}\right]}{\left|(\mathbf{a}-\mathbf{b})\cdot\hat{\mathbf{x}}\right|^2} && (12)\\
&= \frac{\left(x^n + \sum_{k=1}^{n} a_k x^{n-k}\right)\sum_{j=1}^{n}\left(\overline{a}_j - \overline{b}_j\right)x^{n-j}}{\left|(\mathbf{a}-\mathbf{b})\cdot\hat{\mathbf{x}}\right|^2} && (13)
\end{aligned}$$

$$\begin{aligned}
&= \frac{\sum_{j=1}^{n}\left(\overline{a}_j - \overline{b}_j\right)x^{2n-j}}{\left|(\mathbf{a}-\mathbf{b})\cdot\hat{\mathbf{x}}\right|^2}\\
&\quad + \frac{\sum_{j=1}^{n}\sum_{k=1}^{n} a_k\left(\overline{a}_j - \overline{b}_j\right)x^{2n-j-k}}{\left|(\mathbf{a}-\mathbf{b})\cdot\hat{\mathbf{x}}\right|^2}. && (14)
\end{aligned}$$

Since the denominator is real, $\tau(x)$ is real iff the imaginary component of the numerator is 0. We rearrange terms on the right half of (14) by substituting $m = j+k$ and removing k:

$$\begin{aligned}
&\sum_{j=1}^{n}\sum_{k=1}^{n} a_k\left(\overline{a}_j - \overline{b}_j\right)x^{2n-j-k}\\
&\quad = \sum_{m=2}^{2n} x^{2n-m}\sum_{j=1}^{n} a_{m-j}\left(\overline{a}_j - \overline{b}_j\right). && (15)
\end{aligned}$$

However, the terms on the RHS are only valid if $1 \le m-j \le n$, i.e. $m-n \le j \le m-1$, while still satisfying $1 \le j \le n$. Therefore, we write (15) as

$$\begin{aligned}
&\sum_{j=1}^{n}\sum_{k=1}^{n} a_k\left(\overline{a}_j - \overline{b}_j\right)x^{2n-j-k}\\
&\quad = \sum_{m=2}^{2n} x^{2n-m}\sum_{j=\max(1,m-n)}^{\min(n,m-1)} a_{m-j}\left(\overline{a}_j - \overline{b}_j\right).
\end{aligned}$$

Using this, we can rewrite the numerator of (14) as a polynomial $\sum_{m=1}^{2n} w_m x^{2n-m}$, where

$$w_m = \begin{cases}
\overline{a}_m - \overline{b}_m & m = 1\\
\overline{a}_m - \overline{b}_m + \sum_{j=1}^{m-1} a_{m-j}\left(\overline{a}_j - \overline{b}_j\right) & 2 \le m \le n\\
\sum_{j=m-n}^{n} a_{m-j}\left(\overline{a}_j - \overline{b}_j\right) & n+1 \le m,\\
 & m \le 2n
\end{cases} \tag{16}$$

To determine $\Im(w_m)$, the imaginary component of w_m, we use the fact that $\mathbf{a} = \mathbf{p}+\mathbf{q}i$, $\mathbf{b} = \mathbf{r}+\mathbf{s}i$, and $\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{s} \in \mathbb{R}^n$. Therefore

$$\begin{aligned}
\overline{a}_j - \overline{b}_j &= (p_j - r_j) - i(s_j - q_j) && (17)\\
a_j\left(\overline{a}_k - \overline{b}_k\right) &= (p_j + iq_j)\left((p_k - r_k) - i(s_k - q_k)\right)\\
&= \left(p_j(p_k - r_k) + q_j(q_k - s_k)\right)\\
&\quad + i\left(p_j(s_k - q_k) + q_j(p_k - r_k)\right) && (18)
\end{aligned}$$

$$\omega_m = \begin{cases} s_m - q_m & m = 1 \\ s_m - q_m + \sum_{j=1}^{m-1} p_{m-j}(s_j - q_j) + q_{m-j}(p_j - r_j) & 2 \le m \le n \\ \sum_{j=m-n}^{n} p_{m-j}(s_j - q_j) + q_{m-j}(p_j - r_j) & n+1 \le m \le 2n \end{cases}$$

Fig. 2. Formulae for coefficients of $\Omega(x)$

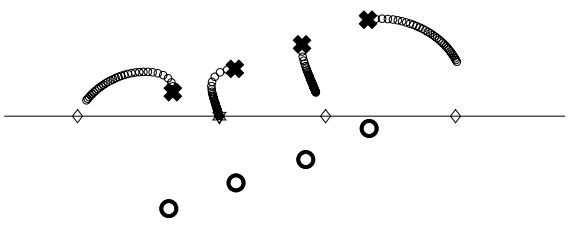

Fig. 3. Collision with real axis

We substitute (17) and (18) into (16) to construct $\omega_m = \Im(w_m)$, producing the coefficients in Figure 2. The real zeroes of the degree $2n-1$ real polynomial $\Omega(x) = \sum_{m=1}^{2n} \omega_m x^{2n-m}$ are the values for x such that $\tau(x) \in \mathbb{R}$. If $0 < \tau(x_j) < 1$, then $V(\mathbf{a}, \mathbf{b})$ intersects the real axis at x_j. ∎

This method was applied to collisions with the real axis. But it can also be used to test for collisions with a segment of the real axis, e.g. a line segment on the real axis from x_{min} to x_{max} (where $x_{min} < x_{max}$). This segment can be checked for collision by rejecting any $x_j \notin [x_{min}, x_{max}]$ without calculating $\tau(x_j)$.

Fig. 3 shows an example of collisions with the real axis. The robots are moving from the X's towards the O's, and the diamonds mark the points of collision with the real axis, as returned by the method described here. The robot paths are given up to the earliest collision, i.e. the x_j among the roots of $\Omega(x)$ with minimum $\tau(x_j)$.

C. Arbitrary Segment Obstacles

Now that we can test for collision with segments on the real axis, we can test for collisions with any segment from $u \in \mathbb{C}$ to $v \in \mathbb{C}$. To do so we make use of the transformations in [21].

Translation: In the complex plane, translation is the same as addition. Apply this addition to all items uniformly.

$$T_c(\{z_1, \ldots, z_n\}) = \{z_1 + c, z_2 + c, \ldots, z_n + c\} \tag{19}$$

Scale & Rotation: In the complex plane, scaling and rotation are actually the same operation: multiplication. Every $c \in \mathbb{C}$ can be written as $c = m \cdot d$, where m is an integer and d has magnitude 1 (set $m = |c|$, and $d = c/m$). Multiplying by m is a scale operation, and multiplying by d is a rotation. Therefore multiplying by c does both.

$$SR_c(\{z_1, \ldots, z_n\}) = \{z_1 c, z_2 c, \ldots, z_n c\} \tag{20}$$

Any line segment can be rotated, scaled, and translated to the real axis via T_{-u} and $SR_{\overline{v}-\overline{u}}$:

$$u' = SR_{\overline{v}-\overline{u}}(T_{-u}(u)) = 0 \tag{21}$$

$$v' = SR_{\overline{v}-\overline{u}}(T_{-u}(v)) = |v-u|^2 \tag{22}$$

In [21], we also defined lifted operators $\widetilde{T}_c$ and $\widetilde{SR_c}$, which show how the transformational operators affect the polynomials. $\widetilde{T}_c = f T_c f^{-1}$, which means that for any $Z \subset \mathbb{C}$ and any $c \in \mathbb{C}$, if $\mathbf{a} = f(Z)$, $Z' = T_c(Z)$, and $\mathbf{a}' = f(Z')$, then $\mathbf{a}' = \widetilde{T}_c(\mathbf{a})$. Similarly, $\widetilde{SR}_c = f SR_c f^{-1}$.

$\widetilde{T}_c$ and $\widetilde{SR}_c$ can be calculated explicitly, without f or f^{-1}:

$$\widetilde{T}_c(a_1, \ldots, a_n) = (a'_1, \ldots, a'_n), \tag{23}$$

where

$$a'_k = \binom{n}{k}(-c)^k + \sum_{j=1}^{k} a_j \binom{n-j}{k-j}(-c)^{k-j}, \tag{24}$$

and

$$\widetilde{SR_c}(a_1, a_2, \ldots, a_n) = \left(a_1 c, a_2 c^2, \ldots, a_n c^n\right). \tag{25}$$

See [21] for derivations of these expressions.

Now, we can take the transformational operators we applied to u and v, and apply the corresponding lifted transformation to the polynomials:

$$\mathbf{a}' = \widetilde{SR}_{\overline{v}-\overline{u}}\left(\widetilde{T}_{-u}(\mathbf{a})\right) \tag{26}$$

$$\mathbf{b}' = \widetilde{SR}_{\overline{v}-\overline{u}}\left(\widetilde{T}_{-u}(\mathbf{b})\right). \tag{27}$$

If $V(\mathbf{a}, \mathbf{b})$ collide with uv, then $V(\mathbf{a}', \mathbf{b}')$ will also collide with the real axis between u' and v'. We can therefore apply the method in III-B to $\mathbf{a}'$, $\mathbf{b}'$, u', and v' to determine if the original path collides with uv.

Fig. 4 shows an example of collisions with arbitrary segments. The robots and edge in this example can be rotated and scaled to match Fig. 3 exactly, and the resulting collision information transforms accordingly. Therefore we can use the collision information from Fig. 3 to find the collisions in Fig. 4.

D. General Obstacle Collisions

We can use the edge collision checker to check collision with any obstacle region. A polygonal region can be checked by checking for collision with each edge, as a robot cannot enter the polygon without crossing an edge. Any non-polygonal region can be approximated by polygons. Therefore we can check for collision with any obstacle region.

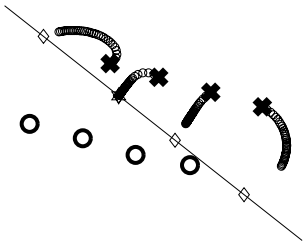

Fig. 4. Collision with segment

IV. Roadmap Planning

The straight-line planner, combined with the obstacle collision checker, form an effective local planner. We use it to apply simple Probabilistic Roadmap (PRM) [23] methods to permutation-invariant formations. To generate each node in the roadmap, we generated n random samples $z_1, \ldots, z_n$ from the collision-free workspace, and constructed a node at $\mathbf{a} = f(\{z_1, \ldots, z_n\})$. Configuration nodes $\mathbf{a}$ and $\mathbf{b}$ were connected iff $V(\mathbf{a}, \mathbf{b})$ did not collide with any edges. To determine routes, we used A* with a metric of

$$d(\mathbf{a}, \mathbf{b}) = \sum_{j=1}^{n} |a_j - b_j|^{\frac{n}{j}}. \tag{28}$$

The exponents applied to the distances are to offset the tendencies of the later exponents to grow much faster than the earlier ones. Compare to (25) above.

Figures 6 and 9 show typical results. In both cases, the border is treated as an obstacle; robots are forbidden from leaving the visible region. Figure 6 uses 50 nodes for 4 robots, while figure 9 uses 200 nodes for 8 robots. Both show PRM planning is feasible for FS^n with obstacles. We can smooth the paths by iteratively averaging the points along the path, while avoiding robot-robot and robot-obstacle collisions. Applying this path smoothing to Figures 6 and 9 produces paths in Figures 7 and 10 respectively. By contrast, figures 5 and 8 show the same start and goal configurations as 6 and 9 respectively, but planned without any obstacle checking or PRM.

V. Conclusion

In this paper we demonstrated obstacle collision detection for permutation-invariant formations, and used it to implement effective roadmap planning that avoids obstacles. The collision checking is efficient – as it can be executed in polynomial time – and is not dependent on resolution or path deconstruction.

There remains much more to explore on this subject. Robot-to-robot collisions, while rare in experiments, are still possible, and remain unchecked. The sampling method is based on selecting individual robot configurations; we would like to find a good formation sampling scheme. The roadmap method used is not optimal; we do not know which metrics, connecting methods, and enhancing mechanisms are most appropriate for this representation. We would also like to look at other types of robots, other levels of centralization and uncertainty, and other types of paths.

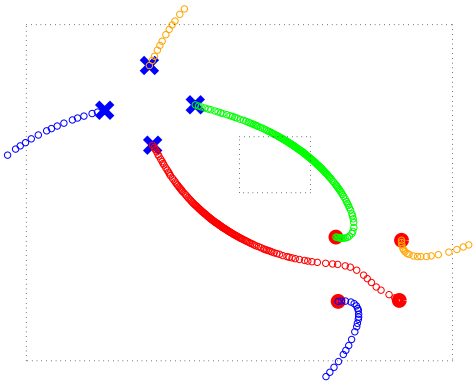

Fig. 5. Four robots, without obstacle checking

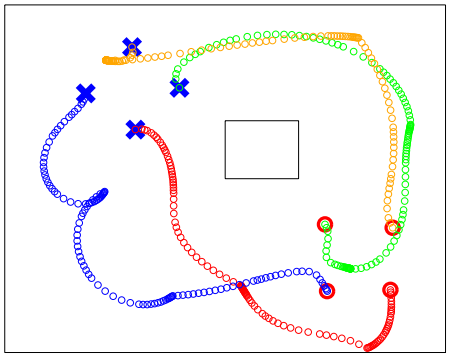

Fig. 6. Four robots, with obstacle checking

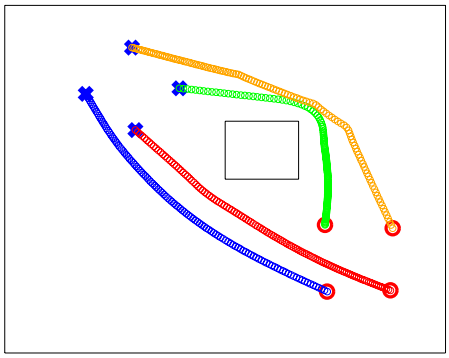

Fig. 7. Four robots, with path smoothing

References

[1] J. Feddema and D. Schoenwald, "Decentralized control of cooperative robotic vehicles," *IEEE Trans. Robot. Automat.*, vol. 18, no. 5, pp. 852–864, Oct. 2002.

[2] D. Fox, W. Burgard, H. Kruppa, and S. Thrun, "A probabilistic approach to collaborative multi-robot localization," *Autonomous Robots*, vol. 8, no. 3, pp. 325–344, June 2000. [Online]. Available: citeseer.nj.nec.com/fox00probabilistic.html

[3] W. Kang, N. Xi, and A. Sparks, "Formation control of autonomous agents in 3d workspace," in *Proc. IEEE ICRA '00. Int. Conf. Robotics and Automation 2000*, vol. 2, San Francisco, CA, Apr. 2000, pp. 1755–1760.

[4] J. S. Jennings, G. Whelan, and W. F. Evans, "Cooperative search and rescue with a team of mobile robots," in *Proc. IEEE ICAR Int. Conf. Advanced Robotics, 1997*, July7-9 1997, pp. 193–200.

[5] D. Rus, B. Donald, and J. Jennings, "Moving furniture with teams

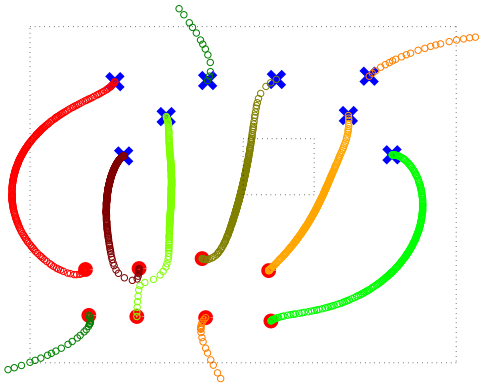

Fig. 8. Eight robots, without obstacle checking

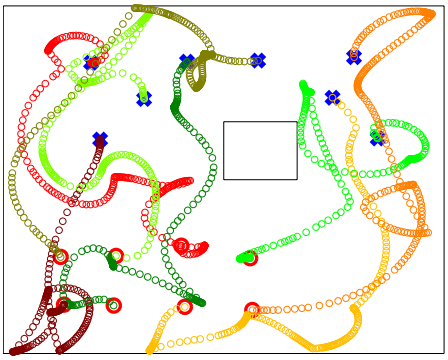

Fig. 9. Eight robots, with obstacle checking

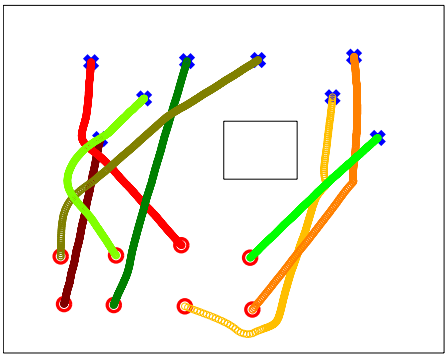

Fig. 10. Eight robots, with path smoothing

of autonomous robots," in *Proc. IEEE IEEE/RSJ Int. Conf. Intelligent Robots and Systems, 1995*, Pittsburgh, PA, Aug. 1995, pp. 235–242.

[6] M. Mataric, M. Nilsson, and K. Simsarian, "Cooperative multi-robot box pushing," in *Proc. IEEE IEEE/RSJ Int. Conf. Intelligent Robots and Systems, 1995*, Pittsburgh, PA, Aug. 1995, pp. 556–561.

[7] A. Das, R. Fierro, V. Kumar, J. Ostrowski, J. Spletzer, and C. Taylor, "A vision-based formation control framework," *IEEE Trans. Robot. Automat.*, vol. 18, no. 5, pp. 813–825, Oct. 2002.

[8] J. Schwartz and M. Sharir, "On the piano movers' problem: III. coordinating the motion of several independent bodies: The special case of circular bodies moving amidst polygonal barriers," *Robotics Research*, vol. 2, no. 3, pp. 46–75, 1983.

[9] P. Svestka and M. Overmars, "Coordinated path planning for multiple robots," *Robotics and Autonomous Systems*, vol. 23, no. 4, pp. 125–152, 1998.

[10] G. Sanchez and J. Latombe, "Using a prm planner to compare centralized and decoupled planning for multi-robot systems," in *Proc. IEEE ICRA '02. Int. Conf. Robotics and Automation 2002*, vol. 2, May 2002, pp. 2112–2119.

[11] J. Latombe, *Robot Motion Planning*. Norwell, Massachusetts: Kluwer Academic Publishers, 1991.

[12] Y. Guo and L. E. Parker, "A distributed and optimal motion planning approach for multiple mobile robots," in *Proc. IEEE ICRA '02. Int. Conf. Robotics and Automation 2002*, Washington, DC, May 2002, pp. 2612–2619.

[13] T. Simeon, S. Leroy, and J.-P. Laumond, "Path coordination for multiple mobile robots: a resolution complete algorithm," *IEEE Transaction on Robotics and Automation*, vol. 18, no. 1, pp. 42–49, Feb. 2002.

[14] S. M. LaValle and S. A. Hutchinson, "Optimal motion planning for multiple robots having independent goals," *IEEE Trans. on Robotics and Automation*, vol. 14, no. 6, pp. 912–925, Dec. 1998.

[15] J. Fredslund and M. Mataric, "A general algorithm for robot formations using local sensing and minimal communication," *IEEE Trans. Robot. Automat.*, vol. 18, no. 5, pp. 837–846, Oct. 2002.

[16] D. W. Gage, "Command control for many-robot systems," in *AUVS92, the Nineteenth Annual AUVS Technical Symposium*, Hunstville Alabama, USA, June 1992, pp. 22–24.

[17] Q. Du, V. Faber, and M. Gunzburger, "Centroidal voronoi tessellations: Applications and algorithms," *SIAM Rev.*, vol. 41, no. 4, pp. 637–676, 1999.

[18] D. Popa, H. Stephanou, C. Helm, and A. Sanderson, "Robotic deployment of sensor networks using potential fields," in *Proc. IEEE ICRA '04. Int. Conf. Robotics and Automation 2004*, vol. 1, New Orleans, LA, Apr. 2004, pp. 642 – 647.

[19] A. Howard, M. Mataric, and G. Sukhatme, "An incremental deployment algorithm for mobile robot teams," in *IEEE/RSJ International Conference on Intelligent Robots and System*, vol. 3, Oct. 2002, pp. 2849 – 2854.

[20] P. Ogren, E. Fiorelli, and N. Leonard, "Formations with a mission: Stable coordination of vehicle group maneuvers," in *Proceedings of the 15 International Symposium on Mathematical Theory of Networks and Systems*, Notre Dame, IN, Aug. 2002. [Online]. Available: citeseer.nj.nec.com/ogren02formations.html

[21] S. Kloder, S. Bhattacharya, and S. Hutchinson, "A configuration space for permutation-invariant multi-robot formations," in *Proc. IEEE ICRA '04. Int. Conf. Robotics and Automation 2004*, vol. 3, New Orleans, LA, Apr. 2004, pp. 2746–2751.

[22] M. Marden, *The Geometry of the Zeros of A Polynomial in a Complex Variable*. New York: American Mathematics Society, 1949.

[23] L. Kavraki, P. Svestka, J. C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Automat.*, vol. 12, pp. 556 – 580, Aug. 1996.