Motion Planning for Formations of Mobile Robots

T D Barfoot
Controls and Analysis Group
MD Robotics

C M Clark
Department of Aeronautics and Astronautics
Stanford University

<tim.barfoot@utoronto.ca>
41 Old Mill Drive
Toronto, Ontario, Canada
M6S 4J8
T (416)-766-0886

Submitted to
Robotics and Autonomous Systems
– 15 January, 2003 –

Motion Planning for Formations of Mobile Robots

T D Barfoot
Controls and Analysis Group
MD Robotics
9445 Airport Road
Brampton, Ontario, Canada L6S 4J3
<tim.barfoot@utoronto.ca>

C M Clark
Department of Aeronautics and Astronautics
Stanford University
Durand Building, 496 Lomita Mall
Stanford, CA, USA 94305-4035
<chrisc@sun-valley.stanford.edu>

Keywords

Motion Planning, Robot Formations, Mobile Robots, Nonholonomic Constraint

Abstract

This paper is concerned with planning the motion of mobile robots in *formation*, which means certain geometrical constraints are imposed on the relative positions and orientations of the robots throughout their travel. Specifically, a method of planning motion for formations of mobile robots with nonholonomic constraints is presented. The kinematic equations developed allow a certain class of formations to be maintained while the group as a whole exhibits motion. The work was validated using the Stanford Micro-Autonomous Rovers Testbed.

1 Introduction

Networks of mobile robots for planetary exploration [5] and terrestrial applications are currently under much investigation. A key element in the control of such networks is motion planning [18], wherein trajectories are computed to avoid costly collisions. The ability to maintain a specific formation of robots is of particular importance in space exploration where data may need to be collected simultaneously from a distributed array of instruments (e.g., seismology, meteorology, three dimensional vision). The very nature of exploration dictates that such arrays of instruments may often need to be relocated in the work environment. This paper investigates the kinematics of formations of mobile robots, that is, how a certain class of geometrical configurations of mobile robots can be maintained during motion. Once such a geometrical configuration of the formation is defined, the equations of this paper allow the network of robots to be treated as a single large robot. Existing single-robot motion planners, controllers or a tele-operator can be used to specify the reference trajectory of the group as a whole. Given this reference trajectory each robot in the formation is able to compute its individual trajectory. This may have particular relevance for tele-operation of such a network from Earth where communication bandwidth is low and latencies are high.

In this approach we explicitly plan the trajectories of every robot in the formation [6]. A controller is implemented for each robot to ensure it tracks its planned trajectory, in contrast to maintaining some fixed distance to a leader. This allows us to specifically alter the formation's shape while turning [8] to accommodate the nonholonomic constraint of the robots. This is accomplished by setting up a curvilinear coordinate system and ensuring that the "distances" between robots are enforced in this curvilinear coordinate system rather than in the usual rectilinear coordinate system. The curvature of the curvilinear coordinate system is precisely the instantaneous curvature of the reference trajectory. When a formation is turning, this method quite naturally plans for robots on the "outside" to speed up and robots on the "inside" to slow down which is what one would naturally expect.

An overview of our approach is as follows. A reference trajectory is provided by a supervisory entity (e.g., a motion planner or tele-operator). Offsets from this reference trajectory (in curvilinear coordinates) are provided for each robot. Based on these offsets and the reference trajectory, each robot computes its own trajectory around which it closes a control loop.

This paper is organized as follows. We first review the literature and classify our approach using various characteristics. We then discuss static formations, those which we simply would like to maintain during motion.

This is followed by dynamic formations, those which we would like to alter during motion. An example simulation is provided, followed by experimental results from the MARS Testbed, discussion, and conclusions.

2 Review

There have been several approaches to maintaining formations of mobile robots including behaviour-based methods [2], potential field methods [3, 25], virtual structures [23], leader-follower [24, 15], and neural networks [17]. This paper is most similar to the leader-follower work (e.g., [12]) in that we use a reference trajectory and define the motion of each individual robot relative to this trajectory. We do not advocate any particular feedback control method to enable each robot to actually track its planned trajectory (there are many possibilities including [21]). Michaud et al. [20] provide a detailed set of characteristics to classify robot formation work, which we will adopt to place our motion planning approach in the context of other methods:

Perceptual Characteristics: We do not assume one robot in the formation can sense the locations of other robots [11, 16, 20] but rather that each robot can sense its own location relative to a common global reference frame [4, 5]. In our experimental work this is accomplished by means of an overhead camera positioning system [10, 9]. However, our approach does not preclude a robot sensing the relative locations of other robots to improve the estimate of its own global position. Since we are assuming each robot can determine its own global position, inter-robot communication is not necessary but the approach does not prohibit the use of communication in a practical implementation to improve position estimates. Communication is, however, necessary to communicate the planned motion of a reference trajectory to each of the robots. This can be thought of as a hierarchical assumption about communication in that a supervisory entity is able to communicate directly with each of the individual robots.

Formation Characteristics: Our approach is able to support an arbitrary geometrical configuration of robots. As we will see, this shape will be slightly altered when the formation is turning. As the geometrical distances between robots vary slightly with time, our approach can be classified as *flexible* rather than *rigid.* This has implications on the candidate applications of our formations. For example, it may not be appropriate for cooperative carrying situations where each robot holds a corner of a large rigid object. As mentioned above, our motion planning approach uses global positioning rather than relative positioning but we do plan the individual robot trajectories relative to the trajectory of an arbitrary common reference point, which makes our formations *point-referenced* rather than *neighbour-referenced* [11, 20](where each robot maintains position relative to one or more neighbours). A further refinement on the classification scheme of Michaud et al. [20] might be to distinguish between the methods used for state estimation and state control. Our approach uses global positioning but the motion planning is relative to a common reference trajectory.

Control Characteristics: Our architecture is best classified as hierarchical rather than purely decentralized. A reference trajectory is planned which is communicated to each of the individual robots who determine their own individual trajectories. The approach is essentially a geometric one [23, 6, 7, 8]. An individual feedback control loop is built for each robot to ensure it follows its planned trajectory [15, 21]. The approach is *homogeneous* in that each robot can use the exact same algorithm except for changes in parameters (e.g., different offsets from a common reference point).

Our approach allows a static formation to be maintained while moving as described above. It also supports a dynamic formation [14, 12, 19] which might include a change from one shape to another shape. This could be useful if, for example, distributed data needs to be collected at different resolutions and thus the formation must change scale. Or, the formation may need to be nominally very wide (to collect data) but must become narrower to fit between two obstacles. A clear advantage of using motion planning for dynamic formations is that maneuvers can be planned ahead of time to avoid costly collisions [10, 9]. This may also be useful for tele-operation where the operator simply wants the formation to become larger but does not want to manually program each robot nor worry about collisions.

We address the feasibility [1, 22] of particular formations to a certain degree; by planning motion it is possible to know whether a particular formation can follow a particular trajectory without violating certain physical constraints imposed by the robot's structure (e.g., actuator limits, finite robot size). We do not tackle the issue of assembling formations from random initial conditions [20]. Obstacle avoidance [14, 13, 20, 15] is also not addressed but we note that this could be handled by the reference trajectory planner.

3 Static Formations

A common model for a mobile robot with a nonholonomic constraint in Cartesian inertial space is the following:

$$\dot{x}_c = v_c \cos\theta_c, \quad \dot{y}_c = v_c \sin\theta_c, \quad \dot{\theta}_c = v_c K_c \tag{1}$$

where $\begin{bmatrix} x_c & y_c & \theta_c \end{bmatrix}^T$ are the coordinates of the robot, v_c is the current speed, and K_c is the curvature of the trajectory the robot is following (which is a monotonic function of the steering angle of the robot). The nonholonomic constraint of this system, implied in (1), is

$$-\dot{x}_c \sin\theta_c + \dot{y}_c \cos\theta_c = 0 \tag{2}$$

In general the speed and curvature of the robot are limited so we have the additional constraints

$$|K_c| \leq K_{c,max}, \qquad |v_c| \leq v_{c,max} \tag{3}$$

Note that for an ideal differentially driven robot the curvature, K, is not limited (it can turn on the spot). The first thing we must note about a geometrical formation of mobile robots is that the limits of equation (3) on the group as a whole are different than those on the individual robots. For example, consider figure 1 depicting four differentially driven mobile robots in a square. Simply because the individual robots can turn on the spot does not imply that the square formation can turn on the spot (while maintaining the square). In fact, to maintain a perfect square (including the orientation of the robots) the only possible motion is in a straight line.

In order to turn the formation, a concession must be made. We suggest that one approach is to maintain the formation in curvilinear coordinates rather than in the original rectilinear coordinate system. This takes advantage of the nonholonomic constraint imposed on each robot. It also ensures that, for a static formation which does not turn sharper than a threshold curvature, the individual robot trajectories will not collide. This approach could be advantageous, for example, in a data collection task where a particular left-right offset between robots is required to ensure accurate data. Figure 1 shows what the formation would look like while turning. By taking this approach we may steer the formation and thus trace any trajectory (subject to the nonholonomic constraint and the constraints on speed and curvature). We introduce an arbitrary reference point, C, within the formation whose coordinates serve as a single set of reference coordinates for the group. This point could be the center of the formation, one of the robots in the formation, or any other point. All robots in the formation will be described relative to this reference point (but in curved space for reasons described above).

The kinematics equations for the reference point are now given by equations (1)-(3) where $K_{c,max}$ and $v_{c,max}$ are functions of the specific formation. For the remainder of this paper we assume that the trajectory (and control inputs) of the reference point, C, are known. That is, we assume that

$$\begin{bmatrix} x_c(t) & y_c(t) & \theta_c(t) \end{bmatrix}^T, \qquad \begin{bmatrix} v_c(t) & K_c(t) \end{bmatrix}^T \tag{4}$$

are given (t is time). These may be obtained by an existing motion planning algorithm or controller. For convenience, the control inputs can be written as a function of distance, d_c. Using equation (4),

$$v_c(d_c), \quad K_c(d_c), \quad d_c(t) = \int_0^t v_c(\tau) d\tau \tag{5}$$

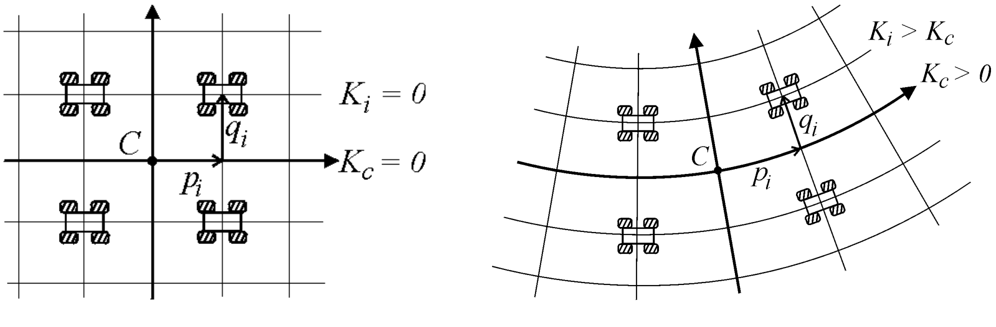

Figure 1: (left) Square formation. (right) Square formation while turning.

This allows the geometry of the reference trajectory to be described independently of the speed along that trajectory. We would now like to compute the trajectory for each of N robots in a formation relative to the reference point C. Figure 1 depicts the $\begin{bmatrix} p_i & q_i \end{bmatrix}^T$ coordinates of robot i relative to the reference point, C. If $\begin{bmatrix} p_i & q_i \end{bmatrix}^T$ is constant then we arrive at the following control inputs for robot i in order to remain in formation

$$\begin{aligned} v_i(s_i) &= v_c(s_i)\Big(1 - q_i K_c(s_i)\Big) \\ K_i(s_i) &= \frac{K_c(s_i)}{1 - q_i K_c(s_i)} \end{aligned} \tag{6}$$

where $s_i(t) = d_c(t) + p_i$ (see Appendix for derivation). Together with

$$\dot{x}_i = v_i \cos\theta_i, \quad \dot{y}_i = v_i \sin\theta_i, \quad \dot{\theta}_i = v_i K_i \tag{7}$$

these equations describe the kinematics of robot i. Note, we must check that

$$|K_i| \leq K_{i,max}, \qquad |v_i| \leq v_{i,max} \tag{8}$$

at all points on the trajectory to ensure $\begin{bmatrix} v_c(d_c) & K_c(d_c) \end{bmatrix}^T$ is a valid reference trajectory. If not, the trajectory cannot be implemented by robot i and this part of the formation cannot be maintained.

An advantage of this approach is that the shape of the formation can be described independently of the reference trajectory. This implies that it is possible to compute the reference trajectory, $\begin{bmatrix} v_c(d_c) & K_c(d_c) \end{bmatrix}^T$, in real-time. To do so, the reference point, C, must be the furthest forward point in the formation (in the direction of travel) or some individual robots will require knowledge about the reference trajectory that has not yet been determined. An equivalent condition is that the p_i must all be opposite in sign to v_c.

4 Dynamic Formations

The more general case is to allow the shape of the formation to change over time. For example, this might be necessary to allow the group to fit through a small opening. This approach handles the general case very nicely by allowing $\begin{bmatrix} p_i & q_i \end{bmatrix}^T$ to be a function of time, or equivalently, distance. To simplify matters we again define the distance travelled by robot i along the reference trajectory as

$$s_i(t) = d_c(t) + p_i(t) \tag{9}$$

where we notice that $p_i(t)$ now changes with time. We also now have $q_i(t)$ as a function of time but we will work instead with $q_i(s_i)$. It becomes necessary to determine all of the following quantities before computing the trajectory of robot i

$$s_i(t), \quad q_i(s_i), \quad \frac{dq_i}{ds_i}(s_i), \quad \frac{d^2 q_i}{ds_i^2}(s_i) \tag{10}$$

Figure 2 interprets these quantities graphically. Once they have been determined, we may compute the local trajectory of robot i as the following

$$\begin{aligned} v_i &= SQv_c \\ K_i &= \frac{S}{Q}\left(K_c + \frac{\Big(1 - q_i K_c\Big)\frac{d^2 q_i}{ds_i^2} + K_c\Big(\frac{dq_i}{ds_i}\Big)^2}{Q^2} \right) \end{aligned} \tag{11}$$

with

$$S = \operatorname{sgn}\Big(1 - q_i K_c\Big) \tag{12}$$

$$Q = \sqrt{\left(\frac{dq_i}{ds_i}\right)^2 + \Big(1 - q_i K_c\Big)^2} \tag{13}$$

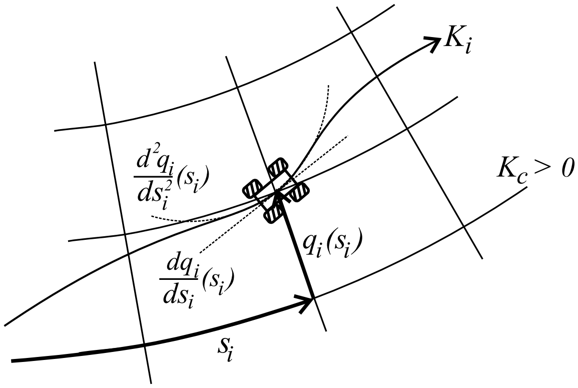

Figure 2: Graphical depiction of dynamic robot trajectory and associated quantities.

where all quantities are evaluated at $s_i(t)$ not $d_c(t)$. Together with equations (7) these describe the kinematics of robot i. We again must ensure the limits in equation (8) are not exceeded at all points on the trajectory. The factor, $\text{sgn}\left(1 - q_i K_c\right)$, in both of the above equations accounts for the fact that a robot might have to move backwards to maintain formation. Notice that when the curvature of the reference trajectory, K_c, is zero, the space is not curved and equation (11b) simplifies to

$$K_i = \frac{\frac{d^2 q_i}{ds_i^2}}{\left(1 + \left(\frac{dq_i}{ds_i}\right)^2\right)^{\frac{3}{2}}} \tag{14}$$

which is a common expression for curvature. Thus we may interpret the two terms of equation (11b) as stemming from the curvature of the reference trajectory (left) and the curvature of the maneuver (right).

A major advantage of this approach is that we may design formation maneuvers independently of the reference trajectory. A library of transformations may be built up and used repeatedly with any reference trajectory (e.g., switch from a square to a line, magnify the current shape, or even permute positions within the same formation).

5 Example Maneuver

It will be useful to give an example of a maneuver within a formation. One possibility is to increase the coordinate $q_{i,o}$ by an amount $\delta q_i = q_{i,f} - q_{i,o}$ over a distance, $\delta s_i = s_{i,f} - s_{i,o}$. To do this we may use the following trajectory

$$q_i(s_i) = \begin{cases} q_{i,o} & 0 \le s_i \le s_{i,o} \\ q_{i,o} + \delta q_i b_i^2 (3 - 2b_i) & s_{i,o} < s_i \le s_{i,f} \\ q_{i,f} & s_{i,f} < s_i \end{cases} \tag{15}$$

where $b_i = \frac{s_i - s_{i,o}}{\delta s_i}$. This in turn allows us to compute

$$\frac{dq_i}{ds_i}(s_i) = \begin{cases} 0 & 0 \le s_i \le s_{i,o} \\ 6\frac{\delta q_i}{\delta s_i} b_i (1 - b_i) & s_{i,o} < s_i \le s_{i,f} \\ 0 & s_{i,f} < s_i \end{cases} \tag{16}$$

$$\frac{d^2 q_i}{ds_i^2}(s_i) = \begin{cases} 0 & 0 \le s_i \le s_{i,o} \\ 6\frac{\delta q_i}{\delta s_i^2} (1 - 2b_i) & s_{i,o} < s_i \le s_{i,f} \\ 0 & s_{i,f} < s_i \end{cases} \tag{17}$$

Figure 3 shows this maneuver carried out while the formation is turning. Now that we have this maneuver worked out, we may use it as many times as we like, for any reference trajectory. We simply need to use equations (15)-(17) in (11) to compute the trajectory of robot i either ahead of time or online.

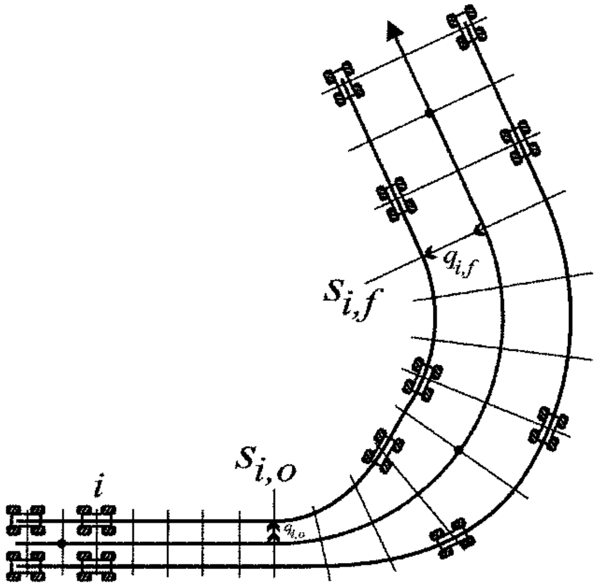

Figure 3: An example maneuver. Robot i moves further away from the reference point during a turn.

Notice that in figure 3 not only does the q_i increase (in magnitude) for each robot but so does the p_i coordinate. This was not explained above but was accomplished by slowing down (or speeding up) each robot while still tracing out the same trajectory. Figure 4 depicts the formation during extended maneuvers at various times.

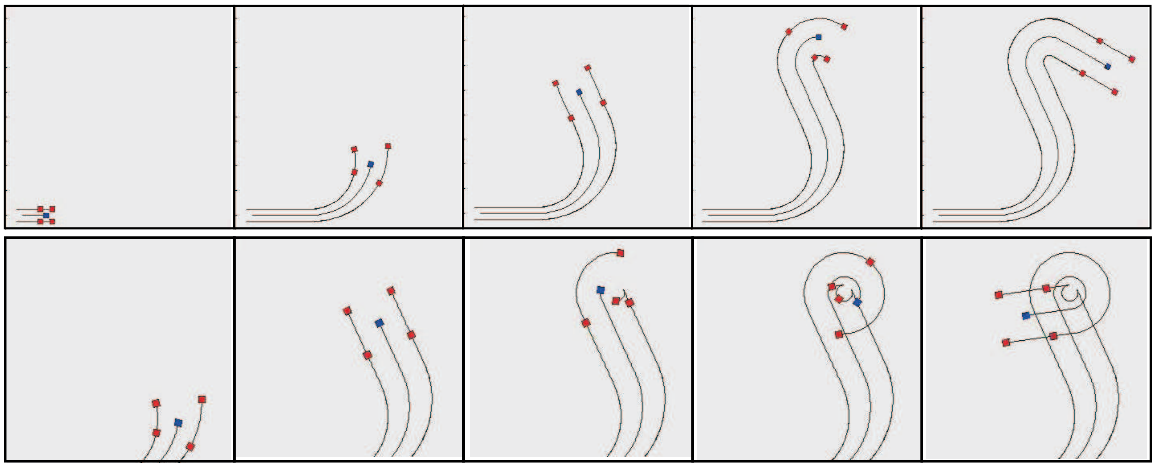

Figure 4: Snapshots of two extended formation maneuvers. In the bottom one the two rovers on the right are required to go backwards to stay in formation in the curvilinear coordinates.

6 Experiment

The Micro-Autonomous RoverS (MARS) Testbed (Figure 5) at Stanford University models rovers in a two-dimensional workspace and was used to carry out the formation planning experiments. The platform consists of a large 3m x 2m flat, granite table with six autonomous robots that move about the table's surface.

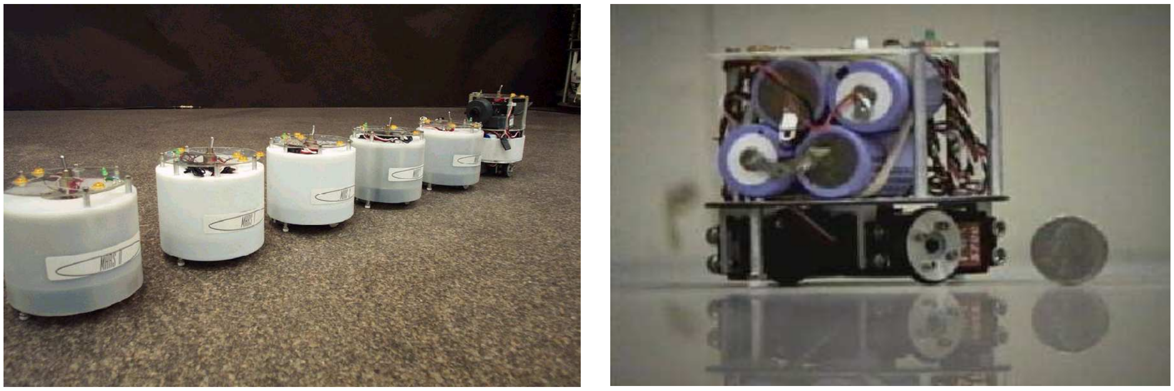

Figure 5: The Micro-Autonomous RoverS (MARS) Testbed.

The robots are cylindrical in shape and use two independently driven wheels that allow them to rotate on the spot, but inhibit lateral movement so as to induce the nonholonomic constraint of equation (2). Each robot has its own motion planner located off-board. Control signal processing is also done off-board, and the control signals are sent to the individual robots via a wireless RC signal.

All communication within the MARS platform is accomplished with Real Time Innovation's Network Data Delivery Service (NDDS) software. NDDS is based on a publish/subscribe architecture. An overhead vision system is used to provide position and velocity sensing. Three cameras with infrared filters are used to detect LEDs mounted on the top surface of robots and obstacles. Each robot/obstacle has a distinct pattern of LEDs to distinguish it from other robots/obstacles. The vision system updates the robot's position and velocity at a rate of 15Hz.

The testbed features a Graphical User Interface (GUI) designed in Java/Swing. It provides a top-down view of the table including graphical representations of robots and obstacles (see Figure 6). Setting robot goal locations is accomplished with a drag and drop system. New goal locations are sent to the respective motion planner so trajectories can be constructed. For further details on the MARS Testbed, see [9].

In order to construct the reference trajectory that the formation will follow, a Probabilistic Road Map (PRM) planner similar to that described in [9] and [10] was implemented. The planner provides real-time generation of trajectories that satisfy the necessary dynamic and kinematic constraints (e.g. the nonholonomic constraint).

For real-time formation planning experiments, a single "Leader" robot is used to construct the reference trajectory. The reference point, C, coincides with this Leader. A collision-free reference trajectory for the Leader robot is constructed using the PRM planner. When checking for collisions, the planner assumes that the Leader is a circular robot with an effective radius equal to the largest distance from the Leader to any other robot in the formation (Euclidean distance). For a sample maneuver refer to figure 6.

After the Leader constructs the reference trajectory, it communicates this trajectory to all other robots in the formation. Assuming robots have knowledge of the Leader's and their own initial position, they can calculate their offsets from the Leader's initial location. This provides the static offset values p_i and q_i depicted in Figure 1. Knowing the Leader/reference trajectory and their own formation offsets, each robot can plan its trajectory according to equations (6) through (8).

Figure 6 depicts a typical static formation maneuver involving three robots and three stationary obstacles. Photos of the maneuver are presented with their corresponding screen-shots of the GUI below. The three robots, located in the bottom left corner of the image in Figure 6 (left column), begin in a triangle formation. Once the start command is given, the Leader robot constructs a collision-free trajectory to the desired goal location.

In Figure 6 (middle two columns), the formation is progressing successfully towards the goal. Before reaching the goal, the user has moved the desired goal location of the formation to illustrate the real-time planning. In Figure 6 (right column), the formation is continuing along its new trajectory to the goal. Note that the goal

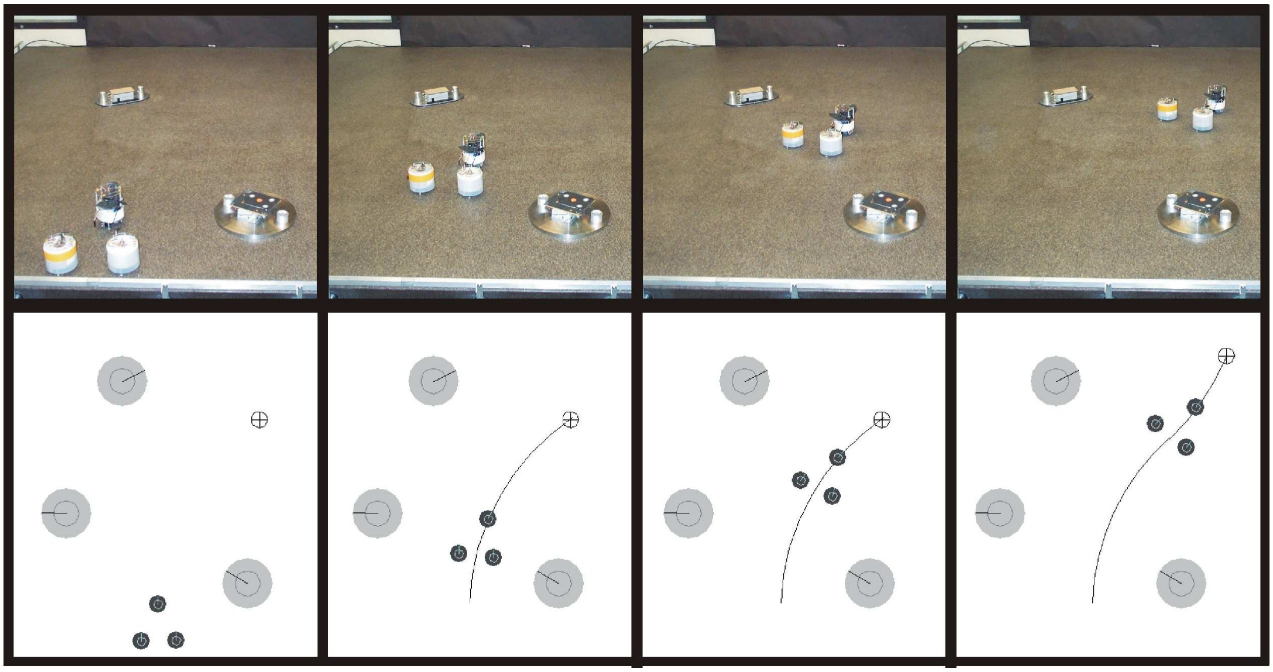

Figure 6: A typical static formation maneuver. Top row shows time progression of hardware formation. Bottom row shows corresponding graphical display. The large circles represent obstacles and the small circles represent robots. The symbol, $\oplus$, indicates the goal which is dynamic. Note, the reference trajectory has been altered in real-time in the right most column by moving the goal location up and to the right.

relocation occurred far enough in advance to allow all robots in the formation to know the new reference trajectory in a timely fashion.

7 Discussion

The approach we have taken in this paper is to explicitly plan the motion for each robot in the formation. We believe this provides an advantage over previous methods, particularly for the case of dynamic formations, as we may plan to avoid collisions during complex formation maneuvers.

Another novelty of our method is that we allow the distances between robots to change (by design) when the formation turns. This was done to incorporate the nonholonomic constraint associated with the robots. This feature may provide a disadvantage for applications where the robots are trying to cooperatively carry large objects. For these applications, methods that attempt to maintain constant distances between robots may prevail [23]. The advantage of this feature is that we may very simply compute the trajectories of every robot in the formation given the reference trajectory. In fact, we may consider the entire formation to be one robot with the same form of nonholonomic constraint as the individual robots. This not only provides a nice symmetry but allows a great many single robot methods to be recycled and applied at the formation level. We must reiterate that the limits on speed and curvature of the entire formation (e.g., equation (3)) are not identical for a formation and its constituent robots.

The equations provided allow a precomputed (or real-time) formation maneuver to be combined with a precomputed (or real-time) reference trajectory at run-time which makes the method very flexible and modular. This advantage was clearly demonstrated on the MARS Testbed as an existing Probabilistic Road Map planner [9, 10] was used to compute the reference trajectory for a static formation. This trajectory was altered in real-time and the formation was able to compensate easily. It should be stressed, however, that any motion planner (or controller) for a single robot with a nonholonomic constraint could be used to compute the reference trajectory.

In the static formation case, our approach provides a method to automatically plan the trajectories of all the robots, given the reference trajectory and the constant offsets from it (in the curvilinear coordinate system). Other methods, such as input-output feedback linearization [11], have also been successfully used to track a point slightly displaced from a reference trajectory (in the inertial, Cartesian coordinate system). Thus, for the static case, our approach may not provide a large advantage over existing methods, in that we have not reduced the complexity of the problem. In essence, we are capitalizing on the knowledge of the individual robots' capabilities to generate open loop commands that acknowledge the nonholonomic constraint. We also make the observation that under our approach, the reference trajectory must also be subject to the same nonholonomic constraint, an advantage when considering motion planning of the entire formation.

For the dynamic formation case, we feel this approach is particularly advantageous in that we can explicitly plan the paths of each robot (within the formation). Again, other methods such as input-output feedback linearization can be used to change formation shape [11]. In the simplest approach, one could discretely change the offsets (in the inertial, Cartesian coordinate system), and allow the control laws to take care of the transient behaviour; but this may lead to undesirable collisions. Consider the case where two robots must simply switch places within a formation which is travelling in a straight line. To use input-output feedback linearization (where the offsets are instantaneously swapped) will most likely result in a collision (as both the robots will nominally travel along the straight line that joins them, to their new assigned location). To avoid the collision, an intermediate set of offsets could be used (e.g., one robot moves to the side) before the final desired offsets are commanded. However, this requires planning the formation change, something addressed by the planner presented in this paper. Our approach allows generic formation plans to be worked out in advance and applied while following an arbitrary reference path. We should again point out that we have not reduced the complexity of the problem as the individual robot's trajectories must be individually planned in the dynamic case. However, if a library of all the desired formation maneuvers is worked out in advance, then at runtime it remains only to incorporate the reference trajectory information into the plans.

Our experimental validation of this technique has to date involved only static formations. We plan to continue this validation for dynamic formations on the MARS Testbed. There are several extensions to this work which we hope to investigate. An example of a dynamic formation maneuver that was designed by hand was provided. One extension is to use a multirobot planner such as in [9, 10] to plan the formation maneuvers and then combine these with a reference trajectory (which could be generated by the same planning method). This would fully automate the process of motion planning for dynamic formations and would accommodate knowledge about the environments (e.g., obstacles). We would further like to demonstrate the technique using limited visual capabilities (e.g., directional on-board cameras) rather than an overhead camera for robot localization.

8 Conclusions

A motion planning approach to formations of mobile robots with a nonholonomic constraint was presented. It was shown that a formation could be treated in the same way a single robot is treated thus making a great deal of single robot work relevant at the formation level. The method allows the geometry (both static and dynamic) of a formation to be considered separately from the formation's overall trajectory by providing a way to combine these two components. The method was validated on the Stanford MARS Testbed and found to be very flexible in that an existing trajectory planner was used to generate the reference trajectory for the formation in real-time.

Acknowledgments

The research was funded in part by the Natural Sciences and Engineering Research Council of Canada.

References

[1] H Ando, I Suzuki, and M Yamashita. Formation and agreement problems for synchronous mobile robots with limited visibility. In *Proceedings of the IEEE International Symposium on Intelligent Control*, pages 453–460, Monterey, CA, August 1995.

[2] T Balch and R C Arkin. Behaviour-based formation control for multi-robot teams. *IEEE Transactions on Robotics and Automation*, 14(6), 1998.

[3] T Balch and M Hybinette. Social potentials for scalable multi-robot formations. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, San Francisco, USA, April 2000.

[4] T D Barfoot, C M Clark, S M Rock, and G M T D'Eleuterio. Path-planning for formations of mobile robots with a nonholonomic constraint. In *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2819–2824, Lausanne, Switzerland, September 2002.

[5] T D Barfoot, E J P Earon, and G M T D'Eleuterio. A new breed: Development of a network of mobile robots for space exploration. Montréal, Canada, June 19-21 2001. The 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS).

[6] Calin Belta and Vijay Kumar. Motion generation for formation of robots: A geometric approach. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, Seoul, Korea, May 21-26 2001.

[7] David Brock, David Montana, and Andrew Ceranowicz. Coordination and control of multiple autonomous vehicles. In *Proceedings of the 1992 IEEE International Conference on Robotics and Automation*, Nice, France, May 1992.

[8] Qin Chen and J Y S Luh. Distributed motion coordination of multiple robots. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, volume 3, pages 2315–2320), 1994.

[9] Christopher M Clark, Tim Bretl, and Stephen M Rock. Applying kinodynamic randomized motion planning with a dynamic priority system to multi-robot space systems. Big Sky, Montana, USA, March 9-16 2002. IEEE Aerospace Conference.

[10] Christopher M Clark and Stephen M Rock. Randomized motion planning for groups of nonholonomic robots. Montréal, Canada, June 19-21 2001. The 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS).

[11] Aveek Das, Rafael Fierro, Vijay Kumar, Jim Ostrowski, John Spletzer, and Camillo Taylor. A framework for vision based formation control. *Multi-Robot Systems: A Special Issue of IEEE Transactions on Robotics and Automation*, October 2002.

[12] Jaydev P Desai, Vijay Kumar, and James P Ostrowski. Control of changes in formation for a team of mobile robots. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, Detroit, Michigan, May 1999.

[13] Jaydev P Desai, James P Ostrowski, and Vijay Kumar. Controlling formations of multiple mobile robots. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, Leuven, Belgium, May 1998.

[14] Jaydev P Desai, James P Ostrowski, and Vijay Kumar. Modeling and control of formations of nonholonomic mobile robots. *IEEE Transactions on Robotics and Automation*, 17(6):905–908, December 2001.

[15] R Fierro, A K Das, V Kumar, and J P Ostrowski. Hybrid control of formations of robots. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, Seoul, Korea, May 2001.

[16] J Fredslund and M J Mataric. A general, local algorithm for robot formations. *Multi-Robot Systems: A Special Issue of IEEE Transactions on Robotics and Automation*, October 2002.

[17] K Hirota, T Kuwabara, K Ishida, A Miyanohara, H Ohdachi, T Ohsawa, W Takeuchi, N Yubazaki, and M Ohtani. Robots moving in formation by using neural network and radial basis functions. In *Proceedings of the 1995 International Conference on Fuzzy Systems*, volume 5, pages 91–94, 1995.

[18] Jean-Claude Latombe. *Robot Motion Planning*. Kluwer, 1991.

[19] Jonathan R Lawton, Brett J Young, and Randal W Beard. A decentralized approach to elementary formation maneuvers. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, San Francisco, USA, April 2000.

[20] François Michaud, Dominic Létourneau, Matthieu Guilbert, and Jean-Marc Valin. Distributed motion coordination of multiple robots. In *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2740–2745, Lausanne, Switzerland, September 2002.

[21] C Samson and K Ait-Abderrahim. Feedback control of a nonholonmic wheeled cart in cartesian space. In *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, Sacramento, USA, April 1991.

[22] P Tabuada, G J Pappas, and P Lima. Feasible formations of multi-agent systems. In *Proceedings of the American Control Conference*, 2001.

[23] Kar-Han Tan and M Anthony Lewis. Virtual structures for high-precision cooperative mobile robotics control. *Autonomous Robots*, 4:387–403, October 1997.

[24] P K C Wang. Navigation strategies for multiple autonomous mobile robots moving in formation. In *Proceedings of the 1989 IEEE International Workshop on Intelligent Robotics and Systems (IROS)*, pages 486–493, Tsukuba, Japan, September 4-6 1989.

[25] H Yamaguchi and T Arai. Distributed autonomous control method for generating shape of multiple mobile robot group. In *Proceedings of the 1994 IEEE International Workshop on Intelligent Robotics and Systems (IROS)*, volume 2, pages 800–807, 1994.

Appendix: Derivations

In this section we provide the derivation of equations (11), including the simplifying case of equations (6), which describe what local speed, v_i, and curvature, K_i, robot i must follow when the overall formation is travelling with speed, v_c, and curvature, K_c. We note from the onset that these derivations rely on the fact that the curvature of the reference trajectory is piecewise-constant such that our derivation considers K_c to be a constant. To expand to smoothly varying K_c (i.e., C^2 continuous) is certainly possible, but is beyond our scope of this paper.

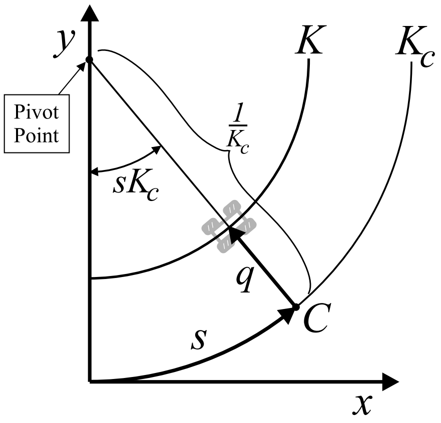

Figure 7: Relationship of Cartesian coordinates, (x, y), to curvilinear coordinates, (s, q). Note that the distance from the reference point C to the pivot point is the *radius of curvature*, $1/K_c$. Also note that the s coordinate for robot i is always measured along the reference trajectory rather than the robot's own trajectory.

The derivation relies on our ability to relate the Cartesian coordinates, (x_i, y_i), to the curvilinear coordinates, (s_i, q_i). To simplify notation we will drop the subscript, i, but bear in mind each robot will have a different set of local coordinates. Referring to figure 7 we find that

$$\begin{aligned} x &= \left(\frac{1}{K_c} - q\right)\sin(sK_c) \\ y &= \frac{1}{K_c} - \left(\frac{1}{K_c} - q\right)\cos(sK_c) \end{aligned} \tag{18}$$

For the first derivatives (in time) of the cartesian quantities we have

$$\begin{aligned} \dot{x} &= -\dot{q}\sin(sK_c) + \dot{s}\,(1 - qK_c)\cos(sK_c) \\ \dot{y} &= \quad \dot{q}\cos(sK_c) + \dot{s}\,(1 - qK_c)\sin(sK_c) \end{aligned} \tag{19}$$

and for the second derivatives (in time) of the cartesian quantities we have

$$\begin{aligned} \ddot{x} &= -\left(\ddot{q} + \dot{s}^2 K_c(1 - qK_c)\right)\sin(sK_c) + \left(\ddot{s}(1 - qK_c) - 2\dot{s}\dot{q}K_c\right)\cos(sK_c) \\ \ddot{y} &= \quad \left(\ddot{q} + \dot{s}^2 K_c(1 - qK_c)\right)\cos(sK_c) + \left(\ddot{s}(1 - qK_c) - 2\dot{s}\dot{q}K_c\right)\sin(sK_c) \end{aligned} \tag{20}$$

For the speed (for robot i) we will require the identity

$$\dot{x}^2 + \dot{y}^2 \triangleq \dot{q}^2 + \dot{p}^2(1 - qK_c)^2 \tag{21}$$

which is straightforward to prove (and is left to the reader). The speed is then given by

$$v = \left(\dot{x}^2 + \dot{y}^2\right)^{\frac{1}{2}} = \left(\dot{q}^2 + \dot{p}^2(1 - qK_c)^2\right)^{\frac{1}{2}} = Qv_c \tag{22}$$

where we note that the speed on the reference trajectory is simply $v_c = \dot{s}$ and we have defined

$$Q = \left(\left(\frac{dq}{ds}\right)^2 + (1 - qK_c)^2\right)^{\frac{1}{2}} \tag{23}$$

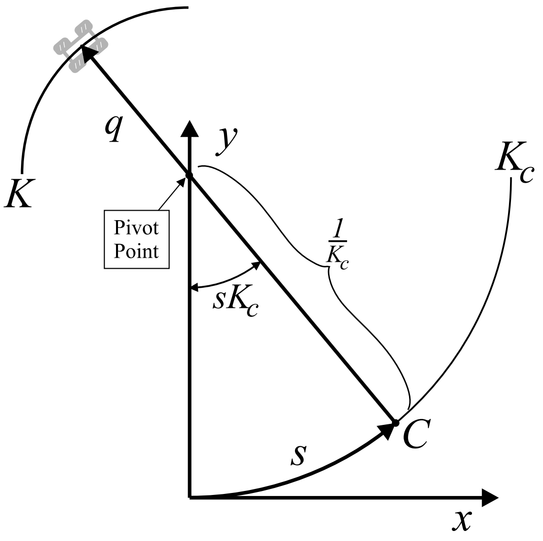

Figure 8: Depiction of when the q coordinate is longer than the radius of curvature, $1/K_c$. In this case both the resulting curvature and speed for robot i are of the opposite sign to the reference trajectory.

where $dq/ds = \dot{q}/\dot{s}$. It is straightforward to plug the above derivatives into the standard expression for curvature (for robot i) in Cartesian coordinates

$$K = \frac{\dot{x}\ddot{y} - \ddot{x}\dot{y}}{(\dot{x}^2 + \dot{y}^2)^{\frac{3}{2}}} \tag{24}$$

to arrive at

$$K = \frac{K_c\dot{s}\dot{q}^2 + (1 - qK_c)(\dot{s}\ddot{q} - \ddot{s}\dot{q}) + K_c\dot{s}\left(\dot{q}^2 + \dot{s}^2(1 - qK_c)^2\right)}{(\dot{q}^2 + \dot{s}^2(1 - qK_c)^2)^{\frac{3}{2}}} \tag{25}$$

If we divide the numerator and denominator by a factor of $\dot{s}^3$ and note that

$$\frac{d^2q}{ds^2} = \frac{\dot{s}\ddot{q} - \ddot{s}\dot{q}}{\dot{s}^3} \tag{26}$$

then we may write

$$K = \frac{1}{Q}\left(K_c + \frac{\left(1 - qK_c\right)\frac{d^2q}{ds^2} + K_c\left(\frac{dq}{ds}\right)^2}{Q^2}\right) \tag{27}$$

Equations (27) and (22) are almost the same as those in equations (11) except that we have neglected to account for the case depicted in figure 8. It is possible to have the q coordinate be larger than the radius of curvature of the reference trajectory, $1/K_c$ whereupon the robot must have curvature and speed opposite in sign to the reference trajectory in order to maintain formation. This is equivalent to saying

$$1 - qK_c < 0 \tag{28}$$

Thus, to arrive at equations (11) from (27) and (22), we multiply these latter two equations by

$$S = \mathrm{sgn}(1 - qK_c) \tag{29}$$

which accounts for the sign of $(1 - qK_c)$.

There are a few simplifying cases to equations (27) and (22) worth mentioning. First, when the q coordinate is constant we have

$$\frac{d^2q}{ds^2} = \frac{dq}{ds} = 0 \tag{30}$$

such that

$$K = \frac{K_c}{1 - qK_c}, \qquad v = (1 - qK_c)v_c \tag{31}$$

which are equations (6). Note that we do not have to worry about including S to account for the situation when $1 - qK_c < 0$ as these expressions already take care of that case.

Another simplifying case of interest is when the reference trajectory is a straight line in Cartesian space such that $K_c = 0$. In this case the (x, y) coordinates and the (s, q) coordinates are exactly the same and so we have

$$K = \frac{\frac{d^2q}{ds^2}}{\left(1 + \left(\frac{dq}{ds}\right)^2\right)^{\frac{3}{2}}}, \qquad v = \left(\left(\frac{dq}{ds}\right)^2 + 1\right)^{\frac{1}{2}} v_c \tag{32}$$

Finally, a case that has not been mentioned explicitly is the case when $1 - qK_c = 0$. Here the robot is positioned exactly at the pivot point of the reference trajectory whereupon

$$K \rightarrow \infty, \qquad v \rightarrow 0 \tag{33}$$

Infinite curvature implies that the robot must turn on the spot to maintain formation which is certainly possible for several common styles of mobile robot (including those used in the experiment section of this paper) but not possible for others.

Timothy D Barfoot is currently working in the Controls and Analysis Group at MD Robotics, Brampton, Canada. Previously, he was an Assistant Professor in the Space Robotics Group at the University of Toronto Institute for Aerospace Studies. He received his B.A.Sc. in the Aerospace Option of Engineering Science at the University of Toronto. His Ph.D. thesis, obtained from the Institute for Aerospace Studies at the University of Toronto, focused on the coordination of multiple mobile robots with application to space exploration. His current research interests lie in the area of experimental mobile robotics.

Christopher M Clark is currently a Ph.D. candidate in the Aerospace Robotics Laboratory, located in the Department of Aeronautics and Astronautics, Stanford University. After finishing a B.Sc. in Engineering Physics at Queen's University, he worked as a Control Systems Designer at Sterner Automation, Toronto, Canada. He received his M.Sc. in Mechanical Engineering at the University of Toronto where he focused on the application of Neural Networks to PID controlled Robots.