Virtual Structures for High-Precision Cooperative Mobile Robotic Control*

Kar-Han Tan
tankh@herd.cs.ucla.edu

M. Anthony Lewis
tlewis@cs.ucla.edu

The Commotion Lab
Computer Science Department
University of California, Los Angeles
Los Angeles, CA 90095-1596

Abstract- *A key problem in cooperative robotics is the maintenance of a geometric configuration during movement. To address this problem, the concept of a* ***virtual structure*** *is introduced. Control methods are developed to force an ensemble of robots to behave as if they were particles embedded in a rigid structure. The method was tested both using simulation and experimentation with a set of 3 robots. Results are presented which demonstrate that this approach is capable of achieving high precision movement which is fault tolerant and exhibits graceful degradation of performance. In addition, this algorithm does not require leader selection as in other cooperative robotic strategies. Finally, the method is highly flexible in the kinds of geometric formations that can be maintained.*

1.0 Introduction

One of the most important and fundamental problems in cooperative robotic systems is that of coordinated motion control. One kind of coordinated motion control involves the maintenance of a geometric configuration during robot movement.

For example, in a box pushing task two or more mobile robots must maintain a certain geometric relationship with the box in order to push it in a desired direction cooperatively. If geometric constraints are violated, the robots will not be able to move cooperatively.

Another example is a spacecraft application. Certain tasks, such as laser interferometry, require that several instruments, spaced up to a kilometer apart, maintain a fixed geometry within 1cm [7].

One way of approaching these problems is through the use of the concept of *virtual structures.* In rigid body motion of a physical object, all points in the object maintain a fixed geometric relationship via a system of physical constraints. No disturbance can be made to one particle that is not propagated to all particles.

* This research was supported by NSF CDA-9303148 and matching funds from the UCLA School of Engineering and Applied Science.

It would be desirable if a robotic system could behave in a similar fashion. We show that this is possible using relatively simple control strategies. Despite the simplicity, the control algorithm achieves high precision performance, is tolerant to failure or degradation of performance of any component robot, and does not require leader selection as in other cooperative robotic strategies.

2.0 Previous Work

There has been a lot of interest in cooperative mobile robotic systems lately. Some emergent themes are traffic control [1], pattern formation and cooperative box pushing/load transportation [2]. Closely related to the problem of motion control with geometric constraints is the problem of forming patterns with multiple mobile robots [4,12,14]. It has been shown that multiple mobile robots can stably converge to a prescribed pattern in time [14]. Cooperative box pushing and load transportation have also been studied [6,10,15]. These two topics are different from the problem of precision motion control. The first is mainly concerned with the final configuration, and the intermediate positions are not necessarily in conformance with the geometric constraints. Box pushing and load transportation are also different because the mobile agents are usually attached to the load, or they exploit some properties of the load for motion control. Studies of formation control have also been carried out [3,6,9,11,13]. Typically the strategy is to have a leader guiding multiple followers. This has the problem of a single point of failure; if the leader fails, a new leader must be selected for continued progress. In addition, if a follower fails, it will be left behind and the formation will be broken.

Often, a robot will not fail completely. For example a wheel may slow down, but not stop completely, or a problem may only be transient. In critical applications, it is desirable to maintain formation even under these extreme conditions. Furthermore, it would be desirable to do this without an excessively complicated control

architecture.

This paper presents an elegant solution that addresses the multiple problems of moving in formation.

The method developed for motion control of multiple robots using the idea that points in space maintaining fixed geometric relationships are actually behaving in the same way as points on a rigid body moving through space. If robots behaved in this way, they would be moving inside of a virtual structure. It is this concept that is explored in detail below.

3.0 Using Virtual Structures and Bi-Directional Control

Multiple robots can be used to form virtual structures. In sensing and manipulation applications, a physical structure is created to maintain a fixed geometric relationship between points on the structure. For example, consider a robotic hand capable of dextrous manipulation. Contact with the object is typically with the tips of the fingers only. The structure of the hand serves to enforce kinematic and force constraints of the tips of these fingers. Another example comes from the area of sensing. In an optical array application, the position of sensor and mirrors may be of importance, and a physical structure may be typically employed to maintain the appropriate geometric constraints.

3.1 Virtual Rigid Bodies

Consider a rigid body where the relative positions of points on the body are fixed. When the body moves in space with six degrees of freedom, points on the body continue to maintain their relative positions, although their positions in space change. We can think of the rigid body as being stationary with respect to a frame of reference. As the frame of reference moves in space, the points keep their positions with respect to the reference frame. Now consider mobile robots in space. If the robots always maintain their relative positions with respect to some reference frame the same way that points on a rigid body do, they can be thought of as forming a rigid structure which does not physically exist. We call this a Virtual Structure.

> **Definition: Virtual Structure (VS)**- A virtual structure is a collection of elements,e.g. robots, which maintain a (semi) rigid geometric relationship to each other and to a frame of reference.

Note that while the robots form a virtual structure with their positions in space, each robot still has some degrees of freedom in varying their respective orientations.

3.2 Moving In Formation

Given a number of mobile robots, a solution to the problem of moving in formation must simultaneously satisfy two goals: making progress in a given direction and maintaining geometric compliance to the virtual structure imposed at all time. The geometric relationship

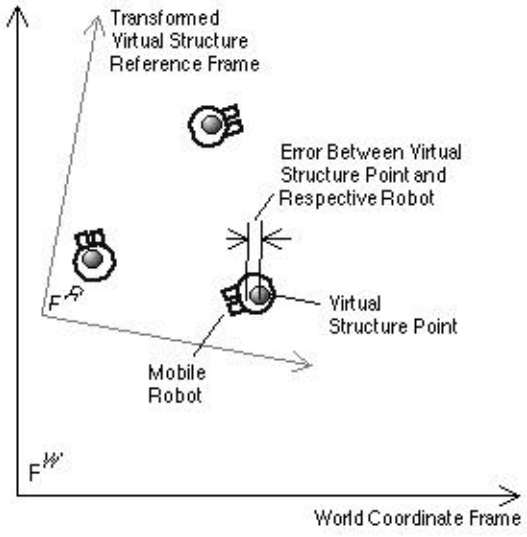

Figure 1: Geometric relationships between mobile robots and the imposed virtual structure

is depicted in Figure 1. We pose the problem formally as follows:

> **Problem: Moving In Formation**
>
> Given n robots labelled $1, \ldots, n$ whose positions in the world coordinate frame is represented by vectors $r_1^w \ldots r_n^w$,
>
> Impose a virtual structure consists of n points, whose positions in its reference coordinate frame is represented by vectors $p_1^r \ldots p_n^r$.
>
> Let I_r^w be a transformation that maps $p_1^r \ldots p_n^r$ to $p_1^w \ldots p_n^w$, the positions of the virtual structure points in the world coordinate frame.
>
> Say that the robots are moving in formation if the robots are moving such that at each point in time

they are also maintaining geometric compliance to the virtual structure imposed.

The problem is thus to design a control algorithm that is capable of making the n robots move in formation in a prescribed direction.

3.3 Solution

The rest of the paper describes a solution for moving in formation.

We adopt a *non-hierarchical* solution. That is, there is not a strict flow of control from top to bottom, as in typical control systems but a *bi-directional* flow of control from top-to-bottom *and* bottom-to-top. This concept is illustrated in Figure 2. In the diagram, we see that the virtual structure's position is controlled by the positions of the robots. Simultaneously, the positions of the robots are controlled by the position of the virtual structure.

If the virtual structure is acted on by an external force, the virtual structure will begin to displace. The robots, trying to match the virtual structure will move in formation with the virtual structure.

If one or more of the robots experiences a fault or a partial loss of performance, the robot will not be able to match the virtual structure perfectly and the virtual structures position will be altered. These two elements, matching the desired position of the virtual structure and modifying the virtual structure's position based on robot performance are the key elements of the algorithm.

We now look at the algorithm in more detail:

---

1) Align the VS with the current robots- VS alignment is via minimization of error between the virtual structure and the current robot position.

2) Move the VS by a Δx and $\Delta\theta$. This increment is determined by a local or global trajectory generator.

3) Compute individual robot trajectories to intercept the desired VS point.

4) Adjust wheel velocities to follow the desired trajectories

5) Goto 1

Next we describe how the minimization in step 1, the trajectory generation of step 2, and the trajectory computation in step 3 is accomplished.

3.3.1 Fitting The Virtual Structure

In the beginning of an iteration the VS's is aligned with the current position of the robots. This is equivalent to finding a VS translation and rotation, I_r^w, that minimizes the distances between the positions of each robot and the corresponding virtual structure points.

A number of distance measure (metrics) can be used. For example:

$$L^1(d_1,\ldots,d_n) = \sum_{i=1}^{n} d_i \qquad (1)$$

$$L^2(d_1,\ldots,d_n) = \sqrt{\sum_{i=1}^{n} d_i^2} \qquad (2)$$

$$L^\infty(d_1,\ldots,d_n) = \sup_{i=1\ldots n} d_i \qquad (3)$$

where

$$d_i = dist(r_i^w, p_i^w) = \sqrt{\left[r_i^w - p_i^w\right]^T \left[r_i^w - p_i^w\right]} \qquad (4)$$

Recall that p_i^w is a function of the position of the VS given by I_r^w. Here I_r^w can be represented as a composition of a translation and rotation:

$$I_r^w = T(x) \bullet R(\theta) \qquad (5)$$

where x is the vector of translation and θ is the angle of rotation.

The L^2 and L^∞ measures are used in our simulation and experiments. The L^2 has the interpretation as the sum of error of all robots from their desired VS points. The L^∞ measure has the interpretation as being proportional to the amount of time needed for the robots to move into formation, and works well both in simulation and implementation.

The cost function minimized is of the form:

$$G(I_r^w) = L^n \qquad (6)$$

The goal is to find a I_r^w which minimizes the cost function. In our case we used a direction set method in multidimensional space [16]. This has the advantage that a derivative function does not have to be specified for the algorithm to work.

3.3.2 Trajectory of the Virtual Structure

In the second phase of each iteration, the virtual structure is displaced in the desired direction of movement. The 'pushing' is done by adding a small angular and translational displacement to the current virtual structure position to derive a desired position for the virtual structure in the next timestep. Let $\dot{x}$ be the translational velocity and $\dot{\theta}$ be the rotational velocity. Thus in the next time step $t + dt$ the template position is represented by

$$I_r^w = T(x + \dot{x}dt) \bullet R(\theta + \dot{\theta}dt) \qquad (7)$$

and the desired position for robot i is then

$$r_i'^{w} = T_r^w \cdot p_i^r \qquad (8)$$

In order not to incur any error in complying with the virtual structure, robot i has to be on r'^{w}_i in the next time step.

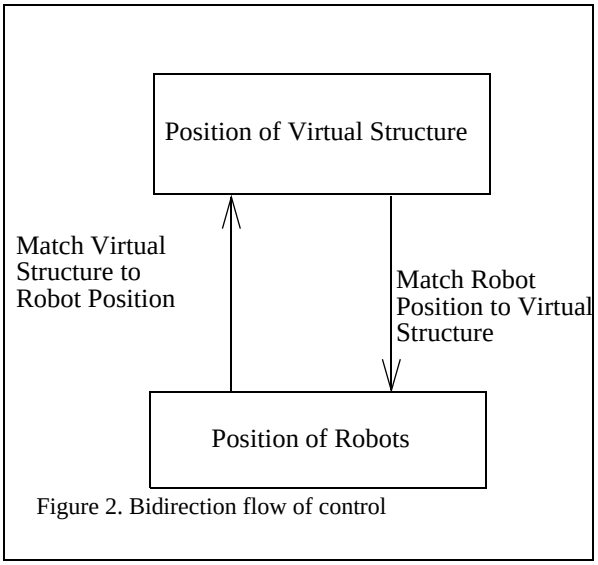

Figure 2. Bidirection flow of control

3.3.3 Trajectory of the robots

The robots used are IS Robotics R3 robots, illustrated in Figure 8. These robots use differential drive. The robot trajectories algorithms were developed for these robots in particular but these results could easily be extended to robots with other kinematic configurations.

We used two approaches to determine the trajectories of the robots. The simplest equation acts to turn the robot toward the desired position. Once the robot is close to the appropriate heading, the robot begins to move smoothly toward the target. The motion of the robot, using this approach, is given by the following algorithm:

$$v_l = \theta_{err} \cdot K + MIN(d_{max}, dist) \cdot (H_c \bullet H_d) \quad (9)$$

$$v_r = -\theta_{err} \cdot K + MIN(d_{max}, dist) \cdot (H_c \bullet H_d) \quad (10)$$

where

$$\theta_{err} = \text{atan}\left(\frac{H_c \times H_d}{H_c \bullet H_d}\right) \qquad (11)$$

and H_c is the current robot heading and H_d is the desired robot heading. The "$\bullet$"and the "$\times$"are the scalar and the vector products, respectively. $MIN(\cdot,\cdot)$ computes the minimum of its two arguments.

The second approach explicitly takes into account the kinematics of the robot and uses this knowledge to plan an optimal move. This approach is described below.

A main cause of robot deviation from the virtual structure is the non-holonomic nature of the mobile robots used in our simulations and experiments. Using our kinematic model of the R3, we visualized the region around the R3 that can be reached within a given amount of time, Δt, and constraints on the maximum acceleration for each wheel. One of the results is shown in Figure 3. The chart shows that it is less expensive to move to regions to the front and back of the robots than to move to regions to the side, because that would require high accelerations to steer the robot.

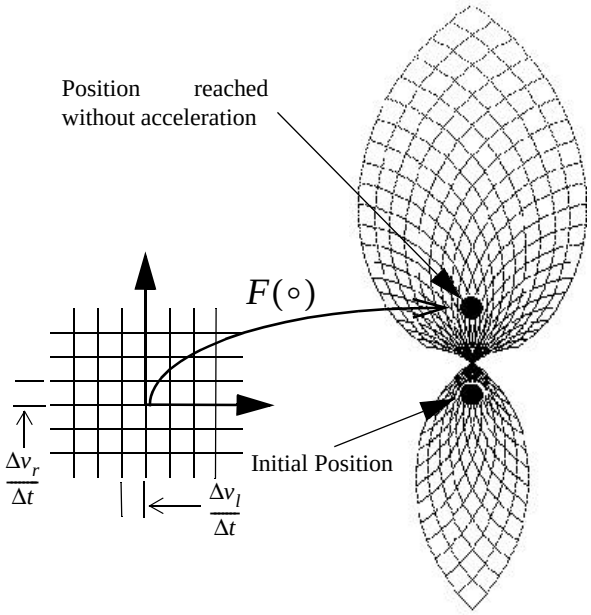

Figure 3: Chart of the function $F(\circ)$ which maps changes in wheel velocity, $(\Delta v)/(\Delta t)$, to changes in the robot's position in the next time unit. This chart was produced using the following initial conditions: initial velocity = 1cm/s; maximum acceleration=1cm/s^2; time given=8s; The dark spot below is the initial position and that on top is the position reached by the robot if it applies no acceleration.

The virtual structure control can be made smoother by incorporating this idea. When the control algorithm 'pushes' the virtual structure, it can take into consideration the costs involved for each robot to reach their destinations. Using the kinematic equations for the robots we can derive an inverse operator. That is, given the next desired position, compute the velocities required. From

$$\Delta x = R\sin\theta \qquad (12)$$

$$\Delta y = R(1-\cos\theta) \qquad (13)$$

we can derive the angle of turning θ and the turning radius R.

$$\theta = \text{asin}\left(\frac{1-\left(\frac{\Delta y}{\Delta x}\right)^2}{1+\left(\frac{\Delta y}{\Delta x}\right)^2}\right) \qquad (14)$$

$$R = \frac{\Delta x}{\sin\theta} \qquad (15)$$

and thus we arrive at two velocities

$$v = (R \pm r)\frac{\theta}{dt} \qquad (16)$$

that are then easily assigned to the left and right wheels depending on whether the desired destination is on the 'left' or 'right' side of the robot. Using this we can formulate a cost function based on changes in velocities such that by minimizing the cost we arrive at a good way of 'pushing' the virtual structure such that it does not require commands outside of the robots performance envelope. In this way errors are minimized.

4.0 Experiments and Results

Design and testing of the control algorithm was carried out in two stages: simulation and real implementation. While simulation was indispensable in developing the algorithm, real implementation proved the power of the control strategy.

4.1 Simulation

Graphical simulation and rendering were carried out with Silicon Graphics Incorporated, Indigo-2 workstation (see Figure 7). The motion of the R3s were simulated with the following equations, whose derivation is illustrated in Figure 5. At time $t = 0$, if the left and

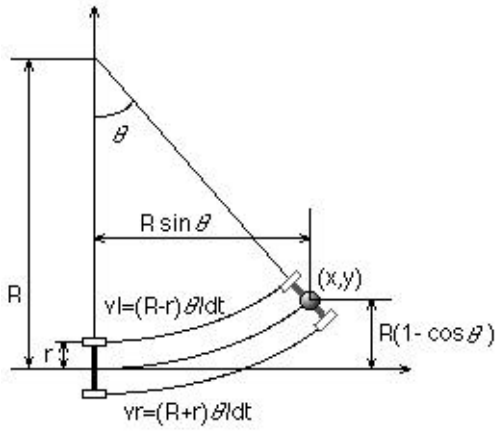

Figure 5: Derivation of wheel equations.

right wheels are given the velocities vl and vr, then the robot moves along a circular arc with radius R, which can be derived from the desired translation along and perpendicular to the robot's axis:

$$\theta = \frac{(vl - vr)}{-2r}dt \qquad (17)$$

$$R = \frac{(vl + vr)}{2\theta}dt \qquad (18)$$

The circular arc result is based on the assumption that within the timestep the robot moves with constant wheel velocities. The case of moving along a straight line can be thought of as moving along a circular arc of infinite radius. This approximation scheme is computationally inexpensive and thus lends itself to real time control and simulation. Simulation results are shown in Figure 6.

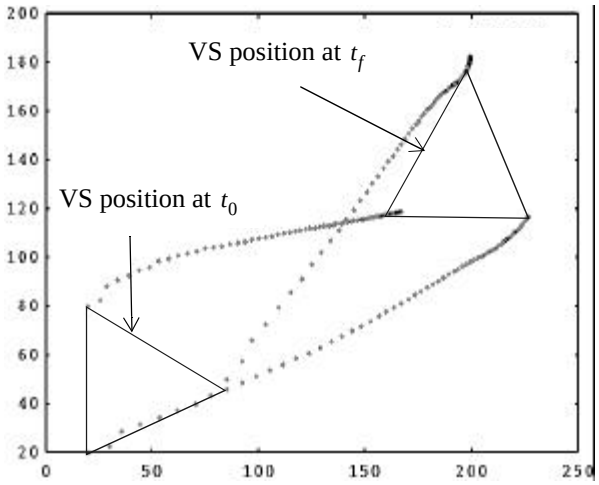

Figure 6: Smooth trajectories followed by robots when nonholonomic constraints are taken into account.

An interesting property of the method can be demonstrated both in simulation and real implementation: when a robot fails and thus halts, the virtual structure is maintained even while the rest of the robots do not know of the failure. Figure 9 illustrates the behavior of 3 simulated robots during a 'failure.'

The mobile robots were initially positioned near the lower left hand corner of the plot. In both of the experiments the virtual structure was given the goal of moving to a target position from the same initial position. In the case where the robot fails the virtual structure initially moves forward until the virtual structure fitting error causes both of the other robots to swerve in order to make progress towards the target. The virtual structure then comes to a halt. The significant point here is that the virtual structure does not disintegrate with robot failures, as leader-follower systems do. This enables higher-level planning processes to recognize that there

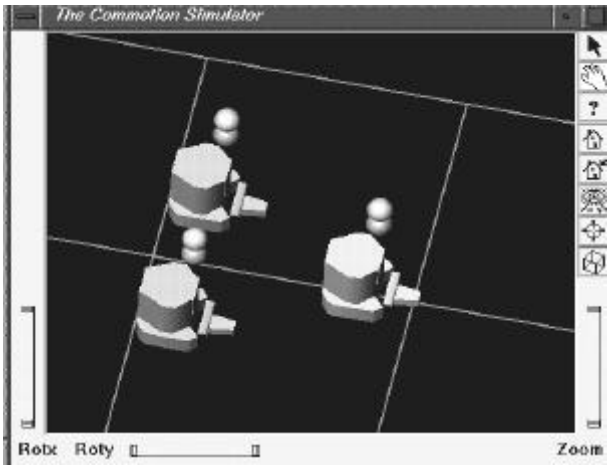

Figure 7. Rendering of the simulated R3s. The upper sphere represent the current position of the VS and the lower spheres represent the VS after position increment. See text for details.

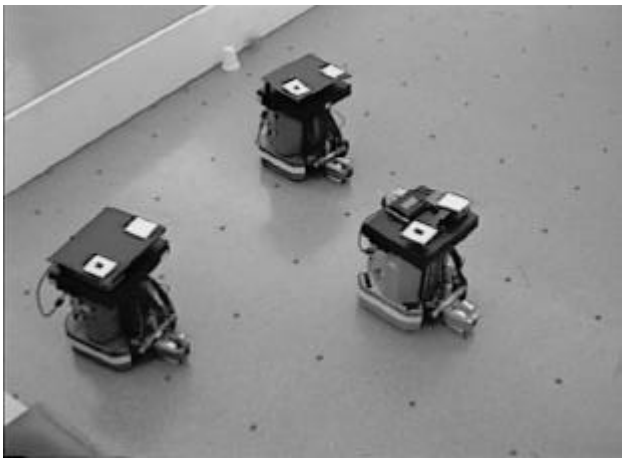

Figure 8. The experimental setup. Shown are three R3 Robots with LINUX boxes and tracking features.

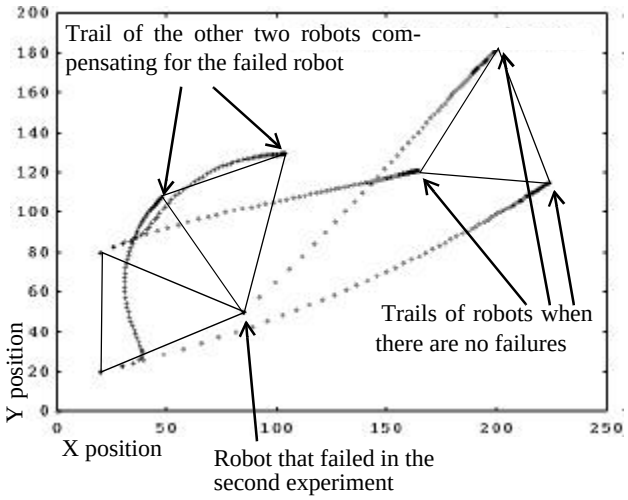

Figure 9: Trajectory of three robots in two experiments, one with a robot failure.

is a system failure and take appropriate action.

This "higher level" is not dealt with in the current work since the current work is focused on the concept of the Virtual Structure. The higher level process could be used to sacrifice a robot by taking it out of the virtual structure or by replacing the robot with a spare. But the key idea here is that the geometric structure is maintained as long as possible (i.e. until two or more robots fail).

4.2 Results with Real Mobile Robots

Experiments with real robots were performed with the R3Net mobile robotic testbed at the UCLA Commotion Laboratory. The setup is illustrated in figures 8 and 11. The mobile robots used are differential drive R3s from IS Robotics. These robots originally had to be programmed with a LISP-like language. Small computers with DX4/100 processors were placed on top of the R3's and a C++ API (called R3++) was implemented. The reason for the modification of the R3 was so that they could accessible via Internet connections. The position of each robot is monitored by a vision-based tracking system (called VGPS) that we developed. The computer executing the control algorithm obtains the positioning data from VGPS, computes the appropriate velocities for the wheels of each robot and sends the commands via R3++ calls, which route the commands to the robots with an AT & T WaveLan wireless net-

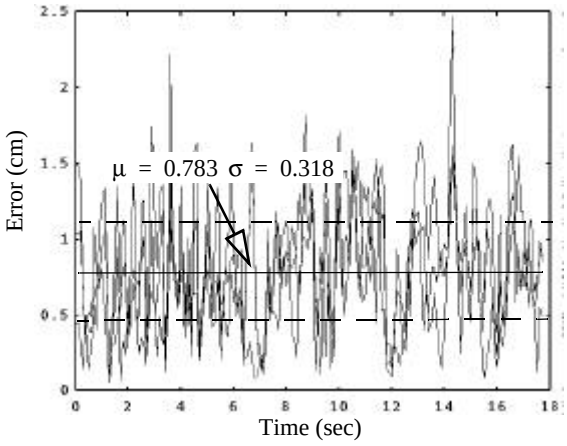

Figure 10: Deviation of each robot from the virtual structure over time when there are no failures.

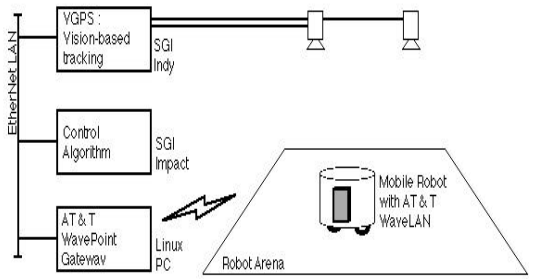

Figure 11. The R3net tracking system. Multiple overhead cameras track markers positioned on top of each R3.

work. Though quite complex, the R3Net has attained a high level of reliability, to the extent that it is capable of supporting remote experiments through a World-Wide Web interface (called W3R3) released recently to the public[17].

The control algorithm is found to be capable of high precision, in spite of the fact that the real robots used differ in performance. In the first experiment, a virtual structure of three mobile robots was given the task of moving towards a target. The deviation from the imposed virtual structure for each robot is plotted in Figure 10 It is apparent that the robots do not deviate more than 2.5 cm from the virtual structure. In this particular experiment, the mean error was $\mu = 0.783$ cm and the standard deviation was $\sigma = 0.318$ cm. This indicates that the robots are moving with a high degree of precision. In the next experiment we test the virtual structure behavior when robot fails. As predicted in simulation, the virtual structure swerves as it attempts to make progress towards the target, but halts soon after. The accumulated error after the simulated failure is shown in Figure 12. As can be seen, while the error does begin to rise, the robots still maintain formation.

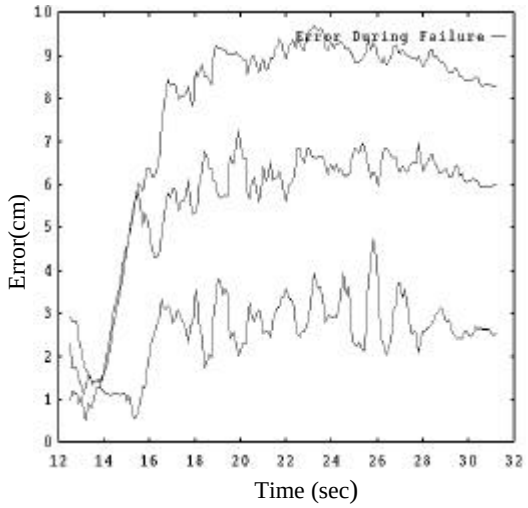

Figure 12: During a robot failure, error increases initially and then stabilizes as the virtual structure adjusts to the failure.

5.0 Conclusions

A novel, extensible and effective method for multiple mobile robot motion control is proposed. Although it has only been tested with robots capable of moving on a plane, there's no inherent limitation to its application to motion control in three dimensional space. There are several directions in which this method can be extended, including flexible/deformable virtual structures, hierarchical virtual structures. Virtual structures can also be applied immediately to many motion control problems.

In conclusion, in this paper we have demonstrated the following merits of our algorithm:

Capable of high-precision control-It is shown by implementation on real robots that the control algorithm can achieve a high level of precision.

Inherently fault tolerant-When one or more robots in the system fail, it is important that the rest of the mobile robots do not let the faulty robots fall behind before some higher-level process can detect the failure and decide on the next action to take. It has been shown in both simulation and real implementation the mobile robots collectively maintain compliance to the virtual structure when robots fail.

No leader election required-Unlike many cooperative robotics control strategies, this solution does not rely on the existence of some leader among the robots. This increases the fault tolerance and avoids leader election, which is a hard problem in distributed algorithms.

Reconfigurable for different kinds of virtual structures- As the algorithm do not rely on particular properties of geometric primitives such as circles or straight lines, the same method can be applied to virtual structures of various shapes with no modification.

Can be implemented in a distributed fashion-The nature of the algorithm is such that it can easily be duplicated and executed on each of the mobile robots with no increase in communications from a centralized implementation.

No explicit functional decomposition- An interest-

ing feature of the solution is that there are no complex protocols for communication or decision making. All the above features are implicit properties and emergent behavior from a set of control equations.

Acknowledgments

The authors would like to thank Jian L. Zhen for invaluable programming support and Romeo Elias for help early on in experimentation and Lucia Simo for valuable suggestions and critical reading of the manuscript.

In addition, the authors thank the anonymous reviewers for their helpful comments.

References

[1] L. Aguilar, R. Alami, S. Fleury, M. Herrb, F. Ingrand, F. Robert. Ten autonomous mobile robots (and even more) in a route network like environment. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 260-267, 1995..

[2] D. L. Brock, D. J. Montana, A. Z. Ceranowicz. Coordination and control of multiple autonomous vehicles. In *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pages 2725-2730, 1992.

[3] Y. Cao, A. Fukunaga, A. B. Kahng and F. Meng. Cooperative mobile robotics: antecedents and directions. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 226-234, 1995.

[4] Q. Chen and J. Y. S. Luh. Coordination and control of a group of small mobile robots. In *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pages 2315-2320, 1992

[5] G. Dudek, M. Jenkin, E. Milios and D. Wilkes. Experiments in sensing and communication for robot convoy navigation. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 268-273, 1995.

[6] M. Hashimoto, F. Oba and T. Eguchi. Dynamic control approach for motion coordination of multiple wheeled mobile robots transporting a single object. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1944-1951, 1995.

[7] Kenneth H. Lau, Jet Propulsion Lab, Personal Communication.

[8] K. Ozaki, H. Asama, Y. Ishida, A. Matsumoto, K. Yokota, H. Kaetsu and I. Endo. Synchronized motion by multiple mobile robots using communication. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1164-1170, 1995.

[9] M. Pachter, J. J. D'Azzo and J. L Dargan. Automatic formation flight control. Journal of guidance, control, and dynamics, vol. 17, no. 6, pages 1380-1383, 1994.

[10] D. Rus, B. Donald and J. Jennings. Moving furniture with teams of autonomous robots. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 235-248, 1995.

[11] S. Sheikholeslam and C. A. Desoer. Control of interconnected nonlinear dynamical systems: the platoon problem. IEEE Transactions on Automatic Control, vol. 37, no. 6, pages 806-810, 1992.

[12] K. Sugihara and I. Suzuki. Distributed motion coordination of multiple mobile robots. In *Proceedings 5th IEEE International Symposium on Intelligent Control*, vol.1, pages 138-143, 1990.

[13] P. K. C. Wang. Navigation strategies for multiple autonomous mobile robots moving in formation. Journal of Robotic Systems, vol. 8, no. 2, pages 177-195, 1991.

[14] H. Yamaguchi and T. Arai. Distributed and autonomous control method for generating shape of multiple mobile robot group . In *Proceedings 1994 EEE/RSJ International Conference on Intelligent Robots and Systems*, pages 800-807, 1994.

[15] Y. Yamaguchi, J. K. Tan, S. Ishikawa, K. Kato. On the development of a cooperative work strategy by multiple robots. In *Proceedings 34th SICE Annual Conference 1995*, pages 1453-1456, 1995..

[16] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P . Flannery, *Numerical Reciples in C: The Art of Scientific Computing*, Second Edition, Cambridge University Press, New York, 1995.

[17] URL: HTTP://muster.cs.ucla.edu/w3r3.