The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Collective Transport of Robots: Coherent, Minimalist Multi-robot Leader-following

Megha Gupta, Jnaneshwar Das, Marcos A. M. Vieira,
Hordur Heidarsson, Harshvardhan Vathsangam, Gaurav S. Sukhatme

Abstract— **We study the collective transport of robots (CTR) problem. A large number of commodity mobile robots are to be moved from one location to another by a single operator. Joysticking each one or carrying them physically is impractical. None of the robots are particularly sophisticated in their ability to plan or reason. Prior work on flocking and formation control has addressed the transport of a robot group that maintains its integrity by explicitly controlling coherence. We show how flocking emerges as a consequence of each robot contending for space near the human operator. A coherent flock can be made to follow a leader in this manner thereby solving the CTR problem. We also present the design of a hand-worn IMU-based gesture interface which allows the human operator to issue simple commands to the group. A preliminary experimental evaluation of the system shows robust CTR with different leader behaviors.**

I. Introduction and Related Work

The term flocking [1] is used to describe behavior wherein multiple robots maintain spatial coherence (*i.e.,* robots stay within a certain distance of each other). When there is the added requirement of maintaining the intra-group geometry according to pre-specified constraints, the problem is called *formation control* [2]. Algorithms for robot flocking [3], [4] are often constructed to achieve a reasonable balance between two objectives - `avoidance` and `aggregation`. In turn, algorithms for the former cause robots to move away from each other so as not to collide, and algorithms for the latter cause robots to move toward each other, so as to maintain the coherence of the group. Algorithms for formation control usually rely on robots being able to sense (and control) precise distances and angles from their neighbors or a special member of the group, called the formation leader.

When a number of robots are to be transported from one location to another, an appealing idea is for the human operator to simply 'lead the way' much like the Pied Piper of Hamelin. This **collective transport of robots (CTR)** is what we study here. In this paper the human operator is the (single) *leader* and the robots being transported are the *followers*. In previous approaches, this problem has been addressed in two ways. The first set of approaches are based on flocking, they focus on maintaining flock integrity. Gross mobility of the flock is achieved by allowing some members of the flock to follow the leader causing others to be 'dragged along', thus causing CTR. In the second set of approaches CTR occurs in formation [5]–[7]. The leader acts as the formation leader, and the followers maintain a specified geometric configuration relative to the leader. While we focus on robot transport here, we mention briefly that there is significant prior work on using robots to transport inanimate objects (*e.g.,* herding [8], caging [9], pushing [10], pulling [11]). Of particular note is work that exploits the innate tendency of certain animals/birds to flock and preserve group integrity - a fact exploited elegantly in [12] to design a robot 'sheepdog' to 'transport' a flock of ducks.

In prior work CTR is achieved because the integrity of the robot group (whether as a flock or a formation) is maintained through explicit control. Must this always be the case? Consider the following example from human behavior. A gaggle of press reporters follows a government spokesperson, each hoping to get within soundbite range. The spokesperson (leader) walks briskly, the reporters (followers) keep up, while shouting out questions. The followers may be characterized as competing for a single resource - namely space. This competition is severe, the leader (the term is a misnomer, but we use it for consistency) may in fact act as an evader, and the followers are better characterized as pursuers. Space near the leader is at a premium and members of the press corps have been known to jostle each other out of the way in the course of their pursuit. If a video of such an exemplar 'chase' was anonymized (*e.g.,* all the followers were replaced by blue dots and the leader was replaced with a red dot), the resulting movie could be described as one where the 'group of blue dots follow the red dot'. This is an example of the leader causing collective transport of the followers by exploiting their competition for space. There are three important characteristics of the behavior in the above example: **Spatial coherence** emerges as a side-effect. As illustrated in the example above, it need not be the explicit goal of the leader, nor the followers. **Transport** of the followers need not be the explicit goal of the leader - in the above instance it is not. **Collisions between followers** are commonplace and not catastrophic.

Inspired by this example and others like it, we design a control system for CTR. Spatial coherence of the robot group is not explicitly maintained. Robots track and follow a human leader, they are programmed to do so by preferring to occupy space near the leader if it is available. While we build in simple collision avoidance, our approach unavoidably results

This work was supported in part by NSF under grants CCF-0120778, IIS-0093233 and IIS-0541224. Marcos Vieira was supported in part by Grant 2229/03–0 from CAPES, Brazil.

The authors are with the Robotic Embedded Systems Laboratory (`http://robotics.usc.edu/resl`), University of Southern California, Los Angeles, CA 90089, USA (`meghagup, jnaneshd, mvieira, heidarss, vathsang, gaurav@usc.edu`)

in collisions, thus we explicitly program collision recovery. Our approach is based on simple sensors, the Wii sensor-rich game remote, and increasing commoditization of small mobile robots (we use the iRobot Create). The primary contributions of this paper are 1. to show an existence proof that CTR is possible without explicitly maintaining the coherence of the group - we do this by giving an algorithm and implementing it on a system of robots, and 2. to experimentally evaluate the robustness and performance of the system. The secondary contribution of the paper is an intuitive command interface for the leader using which high-level commands may be issued to the robots using hand gestures. Gestures are detected using a wrist-mounted, bluetooth-enabled, Inertial Measurement Unit (IMU). A Hidden Markov Model (HMM)-based recognition approach matches time-series IMU data to a dictionary of gestures.

II. System Infrastructure

Fig. 1 shows the components and the overall dataflow in our system.

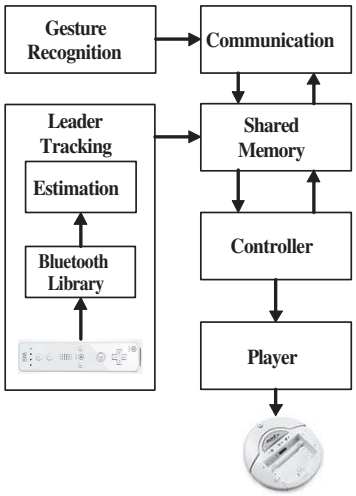


Fig. 1. Robot System Components

The hardware testbed for the experiments consists of 1. a group of mobile robots customized with commodity game devices for leader-tracking, 2. a wearable harness with IR sources facilitating leader detection, and 3. a wrist-mounted IMU for gesture recognition. The robots (Figure 2(a)) are iRobot Creates with an embedded computer running Player [13] for sensing and control (other groups [14]–[16] have recently made a similar choice in different contexts). Each robot uses a front left and front right bumper to detect collisions. A conscious effort was made to choose minimal commodity devices for the tracking hardware. The Nintendo Wii Remote (Wiimote) was chosen for its low cost and reliable multi-object IR tracking engine. The camera has a resolution of 1024x768 pixels, 100Hz refresh rate, and a 45° horizontal and vertical field of view [17] (as a comparision, webcams usually only provide 640x480 tracking at 30Hz and require CPU power to perform real-time tracking). To enhance the horizontal field of view, an assembly with two Wiimotes arranged as shown in Fig. 2(c) was designed and mounted on each robot. The combined wide frame Wiimote arrangement gives a 88° field of view. The robots communicate with the Wiimote through a wireless Bluetooth connection.

A wearable IR-harness was designed and equipped with a vertical assembly of two Nintendo Wii Sensorbars (normally used as IR sources for gameplay with Wiimotes). This assembly provides four equispaced vertical IR sources which are tracked by the Wiimotes on the robots to estimate the distance and orientation of the leader (Section III-A).

A 6DoF V4 IMU with Bluetooth from Sparkfun Electronics [18] was packaged to be worn on the wrist of the leader (Figure 2(b)). A microswitch was attached to the assembly and used by the leader to signal the start and stop of each gesture. In the experiments described here, IMU data was transferred to a computer running the gesture recognition software. This module generates commands for the robots (*e.g.,* follow, stop, *etc.*).

Software development for this work was largely focused on implementing the control system described in the following section. An exception is the inter-robot communication module that provides a channel for all communication. A simple dissemination protocol was used. Each robot periodically broadcasts its position and angle relative to the leader. When a packet is received, a robot checks the timestamp and updates the leader's location with respect to other robots in its shared memory. In our experiments we used UDP with a broadcast rate of 1Hz.

III. Algorithms

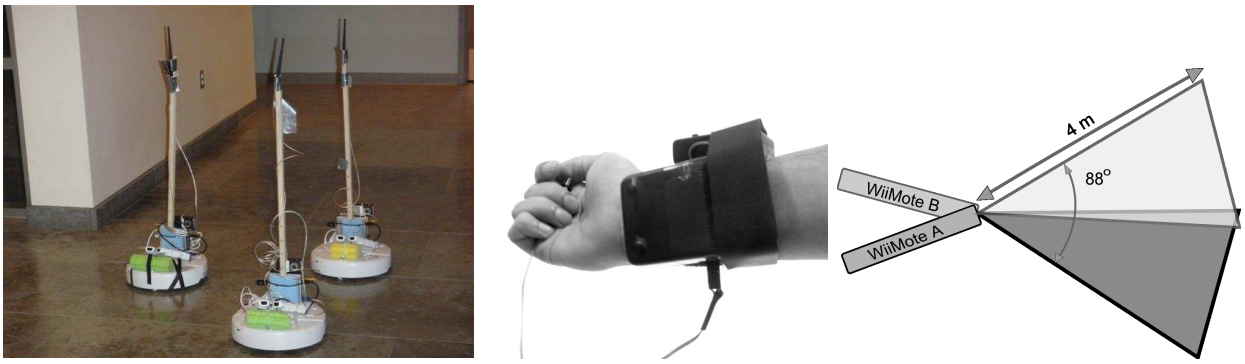
Our system is an implementation of three sets of algorithms used respectively for leader tracking, robot control, and gesture recognition. Leader tracking and robot control are implemented on the robots. Gesture recognition is implemented on a desktop which accepts data from the leader-worn IMU, performs gesture recognition and sends the appropriate command to the robots over WiFi.

A. Leader Tracking

Frames are obtained from each Wiimote at 20Hz. Each frame may have up to four points, corresponding to one of the four IR sources on the user-worn IR harness. Frames are merged and the position of the leader with respect to the robot is computed as follows. From the true distance h between the two groups of IR sources, the observed distance y between the IR sources on the harness, and the focal length f of the Wiimote (Figure 3), the robot computes the distance l between itself and the leader as given in Equation 1. The leader's bearing θ relative to the robot's body frame is computed using Equation 2 (K_{w0} and K_{w1} are constants determined through an offline calibration process, x_{cc} is the image frame center, and x_{IR} is the x coordinate for the projected center of the IR sources length).

$$l/h = f/y \implies l = (f/y) * h \quad (1)$$

$$\theta = K_{w0} + K_{w1}(x_{cc} - x_{IR}) \quad (2)$$



(a) (b) (c)

Fig. 2. System Hardware (a) The team of iRobot Creates, (b) IMU worn on wrist, and (c) Wearable IR harness with two Wii Sensorbars

Consider n image points. $n = 0$ is interpreted as 'leader out of view' and $n = 1$ means that the leader is either too far away or too close to the robot. In these two cases, only the bearing angle to the leader is computed (not the distance to the leader). For $n > 1$, the tracker chooses a pair of neighboring points and uses the distance between them to calculate the distance and bearing of the leader relative to the robot using (1) and (2).

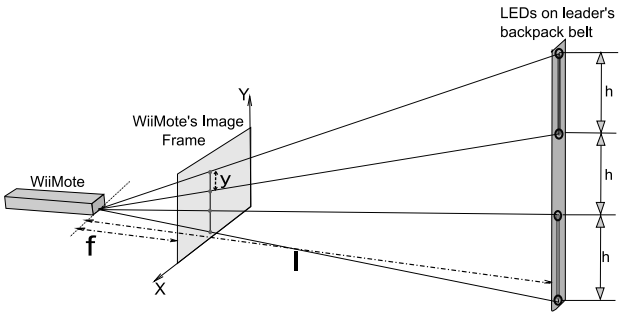


Fig. 3. The camera model

B. Robot Control

Each robot runs the same control program. We describe each of the behaviors below.

Safe-Zone Selection: This behavior attempts to move the robot so that it maintains a specific distance l and bearing ϕ to the leader. An (l, ϕ) pair defines a *safe-zone*. $S1$ and $S2$ are two such safe-zones in Fig. 4 with (l_{S1}, ϕ_{S1}) and (l_{S2}, ϕ_{S2}) being the corresponding parameters. A safe-zone is not unique to a particular robot. A robot can move from one safe-zone to another in the course of the CTR task. How are safe-zones determined and assigned? One approach is to initialize robots at locations near the leader at comfortable inter-robot distances. Each robot uses its Wiimotes to measure (l, ϕ) and broadcasts this value as a safe-zone value to be shared across all robots in the group for them to use later in the course of the CTR task. For a small group of robots, safe-zone values may even be predefined by the leader. The distance of a robot A to a safe-zone $S1$, x_{S1}, can be calculated using (l_{S1}, ϕ_{S1}) and the robot's own distance and bearing to the leader, (l_A, ϕ) as given in (3) (See Fig. 4). Each robot then chooses a safe-zone that is unoccupied and closest to its current position, declares the safe-zone as *occupied*, and moves into it. Each robot maintains this information in an *occupation table*. For a small group of robots, one may assume that there are no race conditions while choosing a safe-zone. However, for a larger group, token scheduling can be used, where a token is passed among the robots based on their IDs. Only the robot with the token can choose a safe-zone from the list of *unoccupied* safe-zones.

$$\begin{aligned} \theta &= \phi_{S1} + \phi \\ {x_{S1}}^2 &= {l_{S1}}^2 + {l_A}^2 - 2 * l_{S1} * l_A * cos(\theta) \end{aligned} \quad (3)$$

The safe-zone behavior causes robots to vie for space near the leader. Instead of anarchy, it reduces interference while the robots compete for space. If each robot selects an exclusive safe-zone, collisions with other robots could be avoided. However, as we indicated in the introduction, our goal in this work is not to obsess over collision avoidance. The heuristic nature of this behavior clearly does not guarantee a collision-free system, but rather a reasonable effort to reduce collisions and (as we see below), a behavior to recover from them.

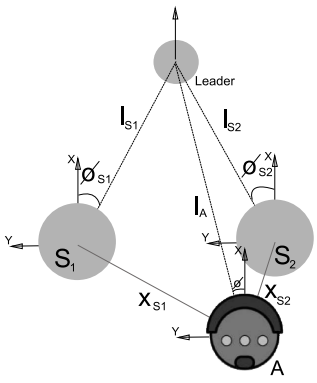


Fig. 4. Showing how a robot chooses a *safe-zone* for itself. $S1$ and $S2$ are two *unoccupied* safe-zones available to robot A. It uses equation 3 to calculate which safe-zone is nearest and then occupies it.

Safe-zones, as we define them, do not enforce a rigid shape on the robot group and so, this behavior is different from formation control.

Leader-following: This behavior attempts to keep a robot in its safe-zone while causing it to follow the leader. It uses (4) and (5) to determine the velocity of the robot based on its distance and bearing with respect to the leader. (l_{sz}, ϕ_{sz}) represent the chosen safe-zone. (l, ϕ) is the robot's position with respect to the leader. v_a is the angular velocity and v_x the translational velocity of the robot (since the Creates cannot move sideways, v_y is always zero). K_a, K_x, $K_{catchup}$, and K_{sat} are constants chosen by the system designers. The robot determines its velocity based on how far it is from its chosen safe-zone as given by the first term in (5). If it is too far away from the leader, it moves faster to catch up with the group (the second term). The third term in (5) causes the robot to slow down when it is turning.

$$v_a = K_a(\phi_{sz} - \phi) \quad (4)$$

$$v_x = K_x(l_{sz} - l) + K_{catchup} * l - K_{sat} * v_a \quad (5)$$

The robot updates its safe-zone value or moves into a different safe-zone when required. This is explained in the following sections.

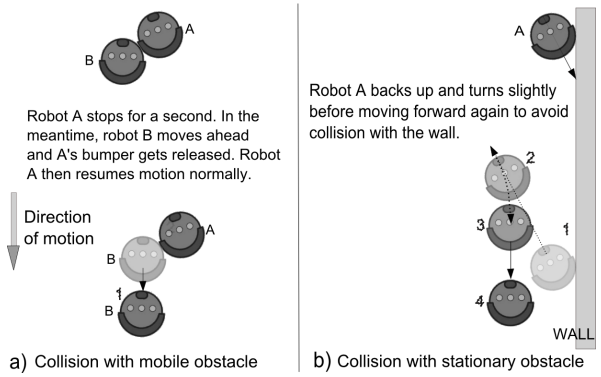


Fig. 5. Collision recovery in the case of (a) a mobile obstacle, and (b) a stationary obstacle.

Collision-recovery: When a robot collides with an obstacle, one or both of its bumpers are activated. If the obstacle is mobile (*e.g.,* another robot), the bumper often tends to be released relatively soon as the mobile obstacle moves away (Fig. 5(a)). This is not the case when a robot collides with a fixed obstacle (*e.g.,* a wall) (Fig. 5(b)). In this behavior, the robot stops for a short duration (1 second in our implementation) to judge if the collision was with a mobile or a fixed obstacle. In the former case, the robot resumes normal motion; in the latter, it backs up, turns, and moves forward. Based on which bumper was hit and its angular velocity prior to the collision, the robot decides the direction in which to turn in order to avoid the obstacle on the retry.

There are some scenarios where a robot incorrectly determines whether the obstacle it collided with is mobile or stationary. Fig. 6 shows an example where robots A and B collide with each other such that bumpers of both robots get activated. Both of them stop for a second, neither bumper is released. Both robots deduce that the obstacle is stationary and back up to avoid it. Even though the deduction about the nature of the obstacle is wrong in this case, subsequent behavior is appropriate because both robots back up and turn away from each other, thus avoiding further collisions with each other on moving forward.

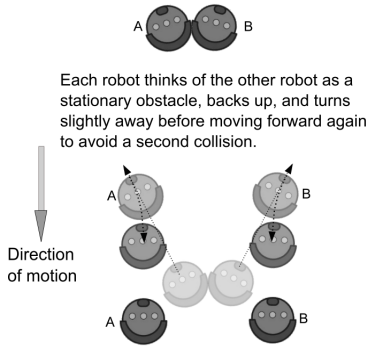


Fig. 6. Erroneously attributing a collision to a static obstacle.

Safe-zone-switching: This behavior attempts to reposition a robot relative to the leader because its collision rate is high. The behavior adjusts the robot's bearing to the leader accordingly, as shown in Fig. 7. If multiple collisions happen with mobile obstacles, the robot is most probably too close to other robot(s) and it is best to move away from it by increasing the bearing maintained with the leader (Fig. 7(a)). This results in an expansion or a bulge. If a high collision rate is due to static obstacles, the robot moves away from the obstacle by reducing the angle to the leader (Fig. 7(b)). This results in the boundary encompassing the group of robots to compress and causes the group shape to elongate. When the robots move from an open space to a corridor-like environment, many collisions are likely with walls - this behavior attempts to reshape the group to fit better within the confines of a corridor. On the other hand, when the group moves from a corridor-like environment to an open space, collisions would mostly happen among robots themselves and this behavior causes the group to expand to avoid this.

In addition to repositioning a robot relative to the leader, we would like to reduce the chance that the robots will cross each other's paths in order to maintain the chosen distances and bearings to the leader. This behavior uses an algorithm to dynamically alter these parameters. The algorithm is executed on each individual robot at random time intervals to avoid race conditions and fluctuations in the parameters. At random intervals each robot uses a cost function to calculate the cost of moving to some other safe-zone (or staying in its currently allocated safe-zone) and chooses the safe-zone with the minimum cost (Algorithm 1).

If robot A finds a slot that has lower cost than its current slot, A takes that slot. The swap is trivial if the slot is unoccupied, involving just an update of the occupation table. If it is occupied by another robot, B, then A swaps the ids for A and B in the occupation table. As each robot is reading

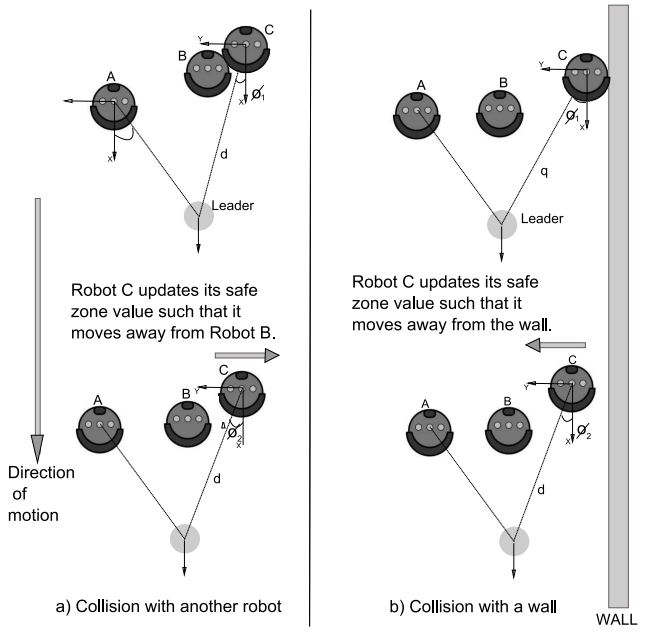


Fig. 7. Safe-zone angle adaptation due to multiple collisions in a short time: (a) multiple collisions with another robot cause robot C to move away from B by increasing its safe-zone angle from ϕ_1 to ϕ_2, (b) multiple collisions with a wall cause robot C to move away from the wall by reducing its safe-zone angle from ϕ_1 to ϕ_2.

the allocations repeatedly, B will update its selected slot (as A's former slot) on its next update. This method depends on the communication system to share states of the robots.

Leader-relocation: This behavior is triggered on a robot if the leader disappears from view. Algorithm 2 describes the relocation strategy.

C. Gesture Recognition

The final piece is the gesture recognizer modeled as an HMM which receives IMU data from the leader and parses it to recognize gesture from a known vocabulary. In our prototype system three gestures were used for the three commands: follow, stop, spin. Each IMU data packet (an observation) is a 6-element tuple (X, Y, Z accelerations, pitch, roll and yaw angles). An HMM model is defined by the parameters λ={A,B,π}. The state transition probabilities, A=$\{a_{ij}\}$ represent the probability of a transition from state S_i to state S_j. These are used to propagate the model at each stage. An observation probability, B=$\{b_j(k)\}$, represents the probability of observing symbol O_k given the system was in state j. π represents a-priori probabilities of the states before the observation sets start arriving. The observation model is represented by a lookup table that contains the probabilities of observing different sensor values given that each value belongs to a state S_i. Recognizing a gesture is equivalent to identifying which state has the highest probability at the end of a gesture sequence. The motivation for the gesture recognition system is to form a intuitive framework for interaction between human leaders and the robots being transported. We hope to use a fully-developed version of this interface in our future work.

Algorithm 1: Dynamic allocation of safe-zones

Input: Robots $\{R_1, ..., R_N\}$, safe-zones $\{S_1, ..., S_M\}$
Output: Safe-zones, $\{A_1, ..., A_N\}$, chosen by the robots
foreach i = *1 to* N **do**
    X_i = current position of R_i
    A_i = currently allocated safe-zone to R_i
    $MinCost_i = dist(X_i, A_i)$
    $B_i = A_i$
    foreach j = *1 to* M *where* $S_j \neq A_i$ **do**
        if S_j *not allocated* **then**
            $Cost_i = dist(X_i, S_j)$
        else
            k = index of robot to which S_j is allocated
            $Cost_i = dist(X_i, S_j) + dist(X_k, A_i)$
        end
        if $Cost_i < MinCost_i$ **then**
            $MinCost_i = Cost_i$
            $B_i = S_j$
        end
    endfch
    $A_i = B_i$
endfch

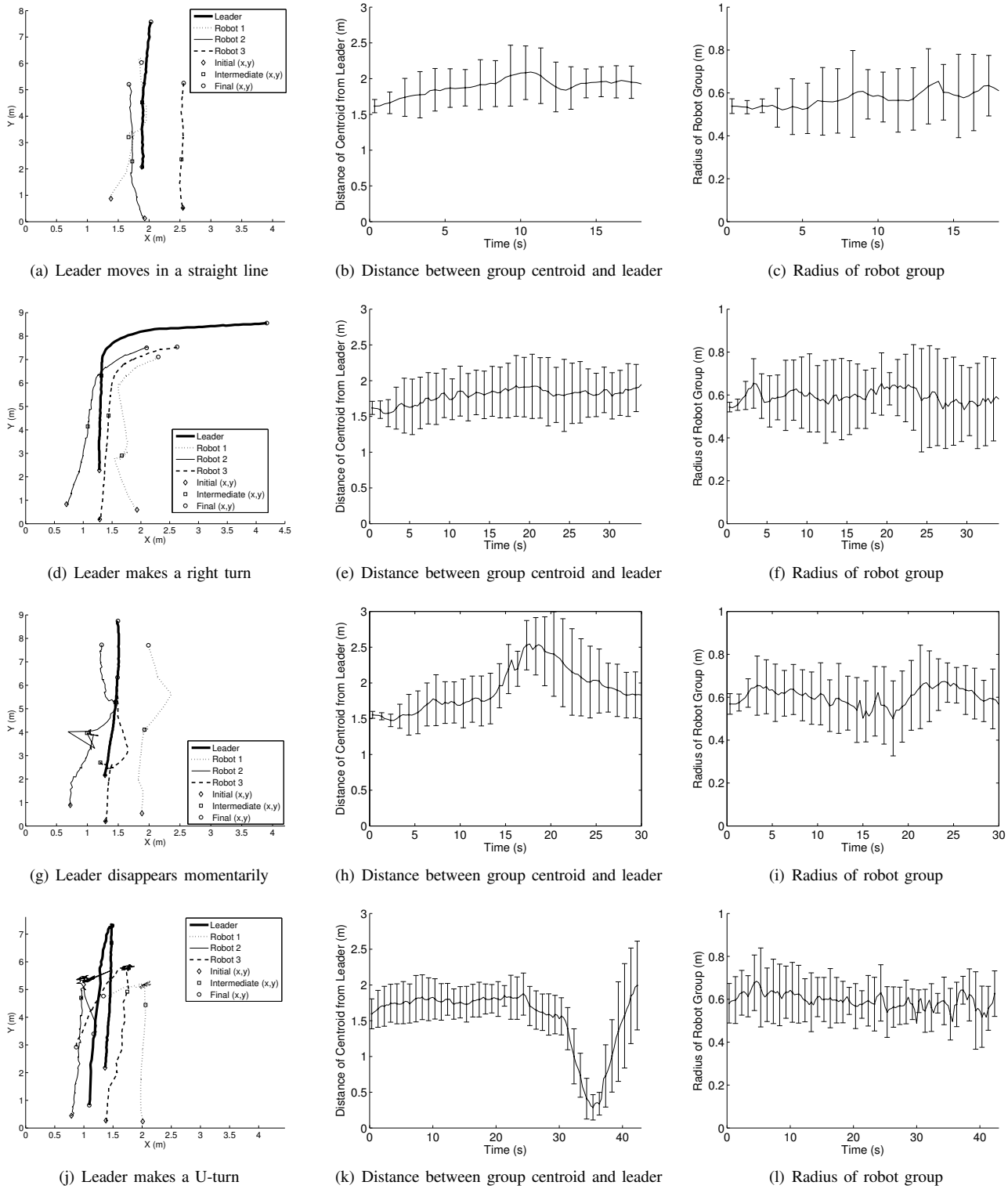
Algorithm 2: Relocating the leader

if *leader is out of view* **then**
    Stop and spin to relocate leader.
    Stop if completed a full rotation.
    if *leader still out of view* **then**
        Go to leader's last seen position using odometry.
        Stop and spin to relocate leader.
        if *leader still out of view* **then** output failure.
    end
end

IV. Experimental Setup and Results

We performed a qualitative evaluation of the complete system with the IR-harness and gesture recognition module worn by a user. As a quantitative evaluation, the system was tested in the controlled setting of a lab space with an overhead camera (Fig. 9). Four markers were pasted on the floor for calibration. Each robot was dressed with a unique color marker. A blob filter tracks each robot's position over time, providing ground-truth. The IR-harness is designed so that it can be mounted on a Create at approximately the same height as a on human leader. This Create serves as a pseudo-leader (no gestures are used). The leader is programmed to move along a pre-planned trajectory. By using a robot as a leader, we can study our system response systematically.

We performed four experiments (5 trials each) with the leader traveling in a straight line, making a right-turn, disappearing from follower view momentarily, and reversing its direction of travel. We caused the momentary disappearance



(a) Leader moves in a straight line (b) Distance between group centroid and leader (c) Radius of robot group

(d) Leader makes a right turn (e) Distance between group centroid and leader (f) Radius of robot group

(g) Leader disappears momentarily (h) Distance between group centroid and leader (i) Radius of robot group

(j) Leader makes a U-turn (k) Distance between group centroid and leader (l) Radius of robot group

Fig. 8. Robot and leader trajectories, distance d of the robot group centroid from the leader, and the radius of the robot group for four leader movement scenarios (straight line, right turn, blocked-leader, U-turn) .

by manually blocking the LEDs on the IR-harness for a few seconds. When making a U-turn, the IR-harness is also not in the field of view of the followers for a brief period (see attached video).

We report on two metrics. The first is the distance, d, between the centroid of the robot team (x_c, y_c) and the leader, (x_L, y_L) (7). The coordinates of the centroid are obtained as shown in (6). The second is the robot group

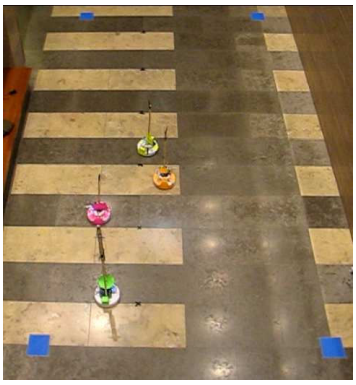


Fig. 9. Overhead camera view in the labspace

radius, r, defined as the average distance of the robots from their centroid (8). For each experiment, we plot the time evolution of the leader and robot trajectories (looking down on the experiment from above) (Fig. 8(a,d,g,j)), the radius r of the group, and the distance d from the leader to the group's centroid (Fig. 8). Each r and d plot is generated by averaging over 5 trials. Each plot also shows the respective standard deviation. The bold line represents the leader and the different kinds of broken lines represent each of the three followers. We label the initial, intermediate and final points of the trajectories.

$$x_c = \frac{1}{n}\sum_{n}^{i=1} x_i; \qquad y_c = \frac{1}{n}\sum_{n}^{i=1} y_i \tag{6}$$

$$d = \sqrt{((x_L - x_c)^2 + ((y_L - y_c)^2))} \tag{7}$$

$$r = \frac{1}{n}\sum_{n}^{i=1} \sqrt{((x_i - x_c)^2 + ((y_i - y_c)^2))} \tag{8}$$

Fig. 8(b,e,h,k) depicts the group distance d to group's centroid. The distance d varies by a small amount over time on the straight line and right angle turn experiments. In the lost leader experiments, the distance d increases a little ($<$1m) at Fig. 8(h) as expected. When we blocked the IR-harness, the leader was still moving away from the followers. When the IR-harness was unblocked, the followers were able to find the leader and the group was able to recover. In the U-turn experiments, the distance d sharply increases ($>$1.5m) at Fig. 8(k) as expected. When the leader executed the U-turn, the IR-harness was briefly out of followers' field of view. After finishing the U-turn, the leader moved toward the followers, decreasing the distance to the group. When the leader had moved enough to be in the followers' field of view, the followers recovered and were able to track the leader again, as shown in Fig. 8. Overall, the experiments show that the system is able to track the leader, is robust and has a reasonable recovery time($<$5s).

Fig. 8(c,f,i,l) show the group radius r over time for the four experiments. The experiments show that although our system does not maintain a rigid formation, it maintains reasonable spatial coherence. It is worthwhile to note that without explicit control of group coherence, r for all the experiments is almost the same, even when collisions occurred.

V. Conclusions and Future Work

We have defined the **collective transport of robots (CTR)** problem wherein a single human operator acts as a *leader* and the robots being transported are the *followers*. In contrast to previous approaches based on explicit flocking or formation control, we design a control system for CTR where spatial coherence of the robot group is not explicitly maintained. Our approach is based on simple sensors, game devices and commodity robots. This paper is an existence proof that CTR is possible without precise control of group coherence. Initial experimental evaluation of the system and the allied gestural interface for command and control shows promise.

Acknowledgments

This work was done as part of a class project for CS 547: Sensing and Planning in Robotics at the University of Southern California.

References

[1] M. Zavlanos, A. Jadbabaie, and G. Pappas, "Flocking while Preserving Network Connectivity," *Decision and Control, 2007 46th IEEE Conference on*, pp. 2919–2924, Dec. 2007.

[2] Z. Lin, M. Broucke, and B. Francis, "Local Control Strategies for Groups of Mobile Autonomous Agents," *Automatic Control, IEEE Transactions on*, vol. 49, no. 4, pp. 622–629, April 2004.

[3] C. W. Reynolds, "Flocks, Herds, and Schools: A Distributed Behavioral Model," *Computer Graphics*, vol. 21, no. 4, pp. 25–34, 1987.

[4] R. Vidal, O. Shakernia, and S. Sastry, "Following the Flock: Distributed Formation Control with Omnidirectional Vision-Based Motion Segmentation and Visual Servoing," *Robotics & Automation Magazine, IEEE*, vol. 11, no. 4, pp. 14–20, Dec. 2004.

[5] S. Monteiro and E. Bicho, "Robot Formations: Robots Allocation and Leader-Follower Pairs," in *ICRA*, 2008, pp. 3769–3775.

[6] Z. Wang and D. Gu, "A Local Sensor Based Leader-Follower Flocking System," in *ICRA*, 2008, pp. 3790–3795.

[7] J. Fredslund and M. Matarić, "A General Algorithm for Robot Formations using Local Sensing and Minimal Communication," *Robotics and Automation, IEEE Transactions on*, vol. 18, no. 5, pp. 837–846, Oct 2002.

[8] M. J. Matarić, "Interaction and Intelligent Behavior," Ph.D. dissertation, Massachusetts Institute of Technology, 1994.

[9] M. F. M. Campos, "Decentralized algorithms for multi-robot manipulation via caging," *International Journal of Robotics Research*, no. 23, pp. 783–795, 2004.

[10] M. J. Matarić, M. Nilsson, and K. T. Simsarian, "Cooperative multi-robot box-pushing," 1995, pp. 556–561.

[11] A. J. Ijspeert, A. Martinoli, A. Billard, and L. M. Gambardella, "Collaboration Through the Exploitation of Local Interactions in Autonomous Collective Robotics: The Stick Pulling Experiment," *Auton. Robots*, vol. 11, no. 2, pp. 149–171, 2001.

[12] R. T. Vaughan, "Experiments in Automatic Flock Control," Ph.D. dissertation, University of Oxford, 1999.

[13] B. P. Gerkey, R. T. Vaughan, K. Støy, A. Howard, G. S. Sukhatme, and M. J. Mataric, "Most Valuable Player: a Robot Device Server for Distributed Control," in *Intl. Conf. On Intelligent Robots and Systems*, vol. 3, Maui, HI, USA, 2001, pp. 1226–1231.

[14] J. Reich, V. Misra, and D. Rubenstein, "Roomba MADNeT: a Mobile Ad-hoc Delay Tolerant Network Testbed," in *MC2R: Mobile Computing and Communications Review*. ACM Sigmobile, 2008.

[15] "The SmURV Robotics Platform," 2008. [Online]. Available: http://robotics.cs.brown.edu/projects/smurv/

[16] "Sumo Robot Assembly Instructions," 2008. [Online]. Available: http://msdn2.microsoft.com/en-us/robotics/bb403184.aspx

[17] J. C. Lee, "Hacking the Nintendo Wii Remote," *IEEE Pervasive Computing*, vol. 7, no. 3, pp. 39–45, 2008.

[18] "IMU 6 Degrees of Freedom v4 Data Sheet," Sparkfun Electronics, 6175 Longbow Drive, Suite 200 Boulder, Colorado, Rev1 - 080422, June 2008.