Proceedings of the 2007 American Control Conference
Marriott Marquis Hotel at Times Square
New York City, USA, July 11-13, 2007

A second generation micro-vehicle testbed for cooperative control and sensing strategies

Kevin K. Leung[1], Chung H. Hsieh[2], Yuan R. Huang[2], Abhijeet Joshi[2], Vlad Voroninski[3], and Andrea L. Bertozzi[3]

Abstract— **This paper describes the second generation of an economical cooperative control testbed described in [C. H. Hsieh *et al* Proc. Am. Cont. Conf., 2006]. The original car-based vehicle is improved with on-board range sensing, limited on board computing, and wireless communication, while maintaining economic feasibility and scale. A second, tank-based platform, uses a flexible caterpillar-belt drive and the same modular sensing and communication components. We demonstrate practical use of the testbed for algorithm validation by implementing a recently proposed cooperative steering law involving obstacle avoidance.**

I. Introduction

Computer simulation, as a tool to validate cooperative control algorithms, is only as good as the accuracy of the model as a reflection of real-world parameters. A real vehicle testbed is an important step forward to verify algorithm effectiveness. As the interest in autonomous multi-vehicle motion continues to rise, the testbed remains as an invaluable learning tool to observe theory in real world action. A 1:1 scale multi-vehicle testbed is impractical for both cost and space constraints. Even with a scaled-down approach, many testbeds involve 10-50cm size vehicles that can not maneuver well in an indoor environment. To this end, a miniaturized, economical micro-car testbed was developed in [1], using 1/64 size vehicles in an integrated system with overhead camera positioning and off-board motion planning. The platform demonstrated the possibility of a multi-function cooperative testbed arena in a cost-effective design (less than $4,000 for all material and computing costs). This paper describes the second generation of this facility, which has many new features while preserving the overall cost and scale of the original design. The first generation vehicle is based on a microsizer car chassis, which has three discrete steering states, a single speed, no on-board processing, and a slow one-way wireless communication rate (13Hz). The second generation vehicles communicate two-way at 30Hz and possess on-board processing and on-board range sensing. Two different chassis designs are implemented, one based on a car platform and a second based on a tank with a caterpillar-style drive, allowing for a negligible turning radius. Hardware is divided into multiple sub-modules which ease future expansion and upgrade. Updated camera positioning software allows for better overhead tracking. The path planning software is now implemented both on and off-board depending on the application. In this paper we demonstrate an application utilizing the dynamic coordinated control laws in [14] and a cumulative sum algorithm for barrier detection inspired from [19]. All the motion planning is done on-board and the off-board computer is only used for communication of overhead camera information and sensor data from the vehicles. Our new ground vehicles have comparable functionality to those in [2-11] , while keeping material cost at the order of $160 per vehicle, on a sub-palm-sized chassis. A pocket size transmitter connects to a laptop with a serial cable, thereby making the entire platform very portable. In some scenarios we are also interested in real time obstacle detection in which the on-board IR sensors play a role. The position tracking system update rate and accuracy are improved.

This paper is organized as follows. In Section II we present the overall system structure, tracking system, the vehicle hardware, local motion control, and the physics model. Section III presents mathematical models for the motion of the vehicles. Section IV describes a suite of steering control laws for different tasks. Section V presents the implementation of the control laws for the tasks of circle following, splitting and merging of a group, and point to point motion of a group with dynamic barrier avoidance.

II. MULTIPLE MICRO-VEHICLE TESTBED

A. Vehicle Platform

Figure 1 shows a system diagram of the platform.

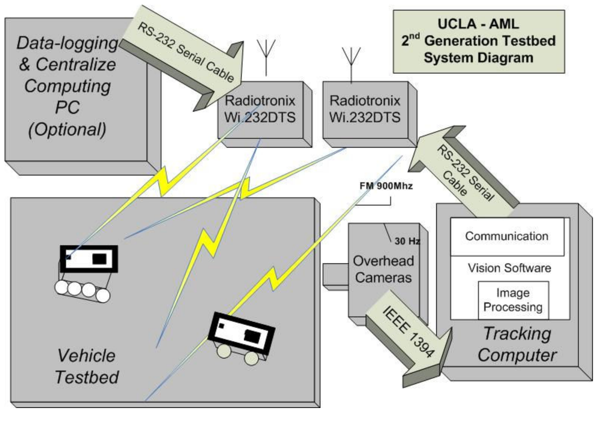

Fig. 1. Testbed system diagram

[1] Dept. of Electrical and Computer Engineering, University of California, Davis 95616 {kevinlg@icedepot.net}
[2]Dept. of Electrical Engineering, University of California Los Angeles, Los Angeles, CA 90095 { chung_hsieh@ieee.org, yuanh@seas.ucla.edu,abhijeet@ucla.edu}
[3]Department of Mathematics, University of California Los Angeles, Los Angeles, CA 90095 {vladvoro@ucla.edu,bertozzi@math.ucla.edu}

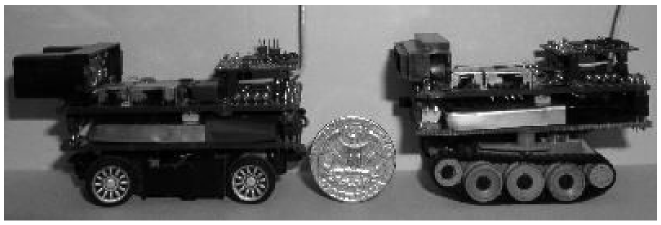

Fig. 2. (left) Car based vehicle. (right) Tank based vehicle.

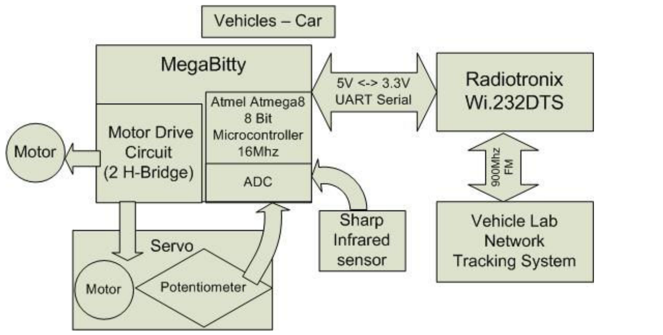

Fig. 3. Diagram of vehicle on-board system. The tank system diagram is the same as above, except that it has another motor instead of the servo.

B. Vehicle Information

We design two vehicles: a car and a tank. The car is built from a ZipZaps micro RC Special Edition car with a 21,500 RPM motor and 12:1 gearing. It has rear wheel drive in either a forward or backward direction. It uses a potentiometer steering gauge that provides feedback to the steering controller. The tank is built from an Ecoman R/C mini tank with 96:1 gearing and the ability to climb a 38° slope. It has two motors, one controlling the left belt and the other the right. Their electronic system infrastructure is identical (see figure 3). For ease of installation and flexibility, the vehicles have four major hardware modules, a processing H-Bridge board (Megabitty), an upper circuit board, a lower deck, and the vehicle chassis. They are interconnected by screws and/or socket pins (see figure 2). This configuration allows future expansions and upgrades. One will only need to modify the lower deck, if a different chassis or type of vehicle is needed. Furthermore, the processing power could also be increased by replacing the Megabitty. Table I shows the physical dimension of the new vehicles compared with the first generation cars in [1]. Note that vehicles equipped with long range IR sensors are 1 cm longer and 2g heavier.

TABLE I
PHYSICAL DIMENSIONS OF THE VEHICLE.

	Dimensions (L x W x H)	Weight (with batteries)
Old Car	$(5.8 \times 3 \times 3.2)$cm	50g
New Car	$(7 \times 3.4 \times 4.3)$cm	68g
Tank	$(7 \times 3.8 \times 4.6)$cm	65g

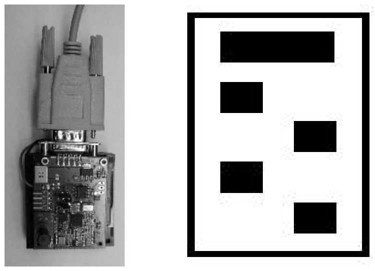

Fig. 4. (left) Wireless communication module. (right) Sample tag for positioning.

C. Vehicle Hardware

The Processing Board. We use the pre-assembled processor board from Junun.org, Megabitty. It has an 8 - bit RISC AVR 16MHz microprocessor and two 500mA rated H-bridges. The module is slightly modified to accommodate the overall hardware structure. This board sits on the upper deck and connects to the wireless module through a 3.3V to 5V level shifter.

The Upper Deck holds the main component of the robot, the communication module, power structure, the infrared sensor, and connectors. The main power reservoir is a single cell 3.7V 740mAh Lithium polymer battery. We utilize the step up regulator to provide an 8V power rail such that a low drop out 5V and 3.3V regulator can maintain stability for the communication unit and the processors module.

The Lower Deck. Screws are bolted tight through the board to the chassis. It holds the mechanics mount between the chassis and the upper deck. It also provides the space to insert the Li - poly battery and as well the on / off switch.

The IR Sensors are 'proximity' model numbers GP 2Y0A02YK and GP 2Y0A21 YK from Sharp. They have a range of 20 to 150 cm and 10 to 80 cm, respectively. The output voltage is higher if an object is closer in their effective range. The sensors were manually calibrated and the details are discussed in [12].

Wireless Communication Module. To aid the cost of development we choose the pre-assembled Radiotronix Wi.232 DTS transceiver module [13] as the wireless bridge between the micro controller and the station. The module costs \$27, is $2cm \times 2.5cm$, and operates at 16mA. As a low-power UART-to-antenna serial interface, this module can be easily integrated with both the Megabitty and the tracking computer. Operating on the 902-928MHz ISM band, the module has the flexibility to transmit and receive on separate channels, thus allow us to achieve a full duplex system. The wireless communication module is shown in figure 4. With the current setup, we achieve a maximum data rate of 57.6 kbit/sec, which is sufficient for the 30Hz positioning update.

D. Software Architecture

The software architecture includes a low level control layer and a user application layer. The control layer consists of four parts: a task scheduler, a basic motion (steering and speed) controller, sensor measurement acquisition, and

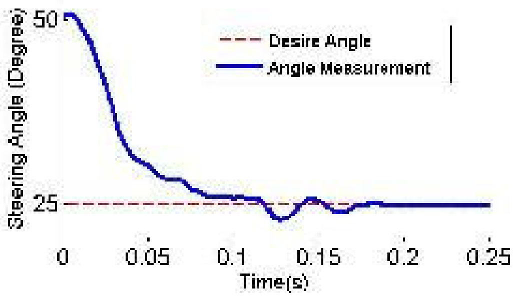

Fig. 5. Steering response, from far left (50°) to center (25°), as measured by the potentiometer.

communication. Application software can access the control layer to alter the vehicle motion, communicate with tracking system, and retrieve range sensor data.

Scheduler. A simple task scheduler regulates the update rate of the steering, motor drive control, and sensor readings. On start up, each the task registers with the scheduler with its priority, update rate, and a callback function. Since the primary use of the scheduler is to update the various local control systems, none of the tasks are allowed to run for longer than the scheduler resolution of one ms. The scheduled task cannot perform any blocking calls; if the task is waiting for additional resources, it is required to reschedule itself to run again later. Scheduled task conflicts are resolved via priorities, if two tasks have the same priority, the tasks will be executed in the order of initial registration. One user task can be scheduled, but it has the lowest priority and can be preempted by any of the controller tasks to ensure correct operation of the vehicles.

Basic Motion Software On The Car. Both the steering and speed motor are controlled by two pulse width modulation (PWM) channels via two H-Bridges. The speed motor is simply controlled by altering the pulse width, while the steering wheel control requires task scheduling for closed-loop feedback control. A potentiometer feeds the analog voltage to the analog-to-digital converter (ADC) on the micro-processor. Task scheduling allows the ADC call to release processing time to other jobs, while waiting for the conversion to be completed. The returned ADC value feeds into the classical proportional derivative controller operating at 250Hz. The steering angle has a total of 51 degrees, 50° for far left, 25° for center and 0° for right, with an accuracy of one degree. The settling time for a 25° offset is 0.18s. Figure 5 illustrates the performance of the steering controller.

Basic Motion Software On The Tank. The tank drives two belts independently, resulting in turns of arbitrary radius, while moving forward and backward. In practice we find it simpler to construct paths composed of either straight line motion or turning in place. A state machine is responsible for the execution sequence of the two maneuvers. Both the speed and heading direction are the input parameters. We assume the heading direction has higher priority than speed. The complete motion state sequence is described in table II. Both

TABLE II
TANK MOTION

	Current heading direction	New heading direction Requested
$Speed = 0$	No motion	1) Turn to desired direction 2) Stand
$Speed > 0$	Move Forward	1) First Turn to new direction 2) Move forward

the straight line and turning maneuvers rely on the tracking system heading angle feedback. We use a simple proportional derivative feedback control scheme to stabilize the tank at the desire heading direction. Note that the left and right driver are not identical, some electrical and mechanical discrepancies exist. A proportional closed-loop controller alters the left and right drive strength to maintain a straight line motion.

Infrared Sensor Measurement Acquisition. The infrared sensor uses another ADC channel to provide an instantaneous digitized measurement. The execution rate is specified in the task scheduling, typically at 25Hz. Various filters can be coded to fit application needs. For the application described in this paper, we implement a cumulative sum algorithm for obstacle detection (see below). Another application, that of dynamic visibility [12], uses an ENO scheme method for processing spatial point cloud data obtained by the range sensors.

Communications Software. Vehicles' tracking data are stacked together and sent through the serial port as one package. The receipt of the tracking data is interrupt-driven. Upon receiving the whole package, the vehicle can extract its own and other vehicles' tracking information. We utilize the Windows API to interface with the serial port on the tracking computer.

E. Tracking System

The tracking of the vehicles is accomplished via an analysis of images from two overhead cameras. The physical setup is the same as in [1]. The tags used to identify the cars are enlarged by 15% and the pattern of the rectangular bits on the tags has been changed from linear to checkered to avoid misidentification. Figure 4 (right) shows a sample tag pattern for the new vehicles.

Additionally, the performance of the tracking algorithm has been enhanced by identifying contours in a thresholded image as those formed by the car tags. Specifically, instead of building a minimum area rectangle around each contour (both sides of the rectangle being parallel to the image plane), a bounding rectangle is built such that it encloses a contour without the restriction of being parallel to the image. Thereby, restrictions can be made on the length and width of the rectangle, instead of its area, the former being a more discriminative selection process.

The revised video tracking algorithm achieves a maximum heading error of 3°, where the old algorithm has a maximum 9° error, and also maintain an average position error of 1 pixel. Additionally, without the presence of noise or occlusion (i.e. for clean image input files), the probability of misidentifying a vehicle has been reduced.

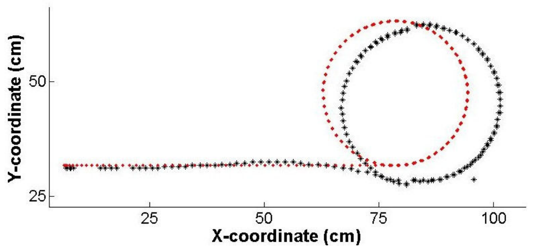

Fig. 6. Open-loop test of vehicle motion laws (1-2). The stars represent the motion of the car on the testbed, while the dots represent a computer simulation of the motion.

III. Vehicle Motion Models

A. The Simple Car Model

The following system of equations, adapted from [1], model the motion of the cars.

$$\dot{x} = v\cos(\theta), \qquad \dot{y} = v\sin(\theta), \tag{1}$$
$$M\dot{v} = (q_1/q_{1max})F - \beta v, \qquad \dot{\theta} = \frac{v}{L_{car}}\tan\left(\frac{q_2}{180}\pi\right), \tag{2}$$

where x and y represent the position coordinates of the vehicle in the laboratory frame, v is the vehicle speed (positive if going forward and negative if going backwards), θ indicates the angle of the vehicle heading, M is the vehicle mass, F is the maximum driving force of the vehicle, and β is the friction coefficient with the floor. The parameter L_{car} is the length of the car. The input control parameter $q_1 \in \{-255, 255\}$ corresponds to the strength of the throttle, -255 being full backward, 255 being full forward, and $q_{1max} = max \mid q_1 \mid$. The parameter $q_2 \in \{0, 50\}$ indicates the 51 possible steering angles of the wheels. Figure 6 compares a computer simulation of vehicle motion to the physical realization of the commands on the testbed. Here $F = 631.8 \times M\ cm/sec^2$ where M is the mass of the vehicle. L_{car} is the length from the front wheels to the back wheels and $\beta = 3.0\ M/sec$.

B. Differential Drive Model

We adapted the model in [18] to formulate a first order system for the tank,

$$\dot{x} = r/2(\omega_l + \omega_r)\cos(\theta), \tag{3}$$
$$\dot{y} = r/2(\omega_l + \omega_r)\sin(\theta), \tag{4}$$
$$\dot{\theta} = r/L_{tank}(\omega_l - \omega_r), \tag{5}$$

where ω_l and ω_r are the angular velocities (rad/sec) of the left and right tank belts, r is the radius of the circle that has the same circumference as a tank belt, and L_{tank} is the width of the tank minus the width of one of the belts. In practice, we restrict the tank's motion such that $\frac{d\theta}{dt} > 0$ only when $\frac{dx}{dt} = \frac{dy}{dt} = 0$. Given this restriction, $|\omega_l| = 3.3357\omega_c/255 - 0.9656$, where $\omega_c \in \{0, 160\}$ is the control parameter to the left belt. When rotating counterclockwise, $\omega_l = -|\omega_l| = -\omega_r$. When rotating clockwise $\omega_r = -|\omega_r| = -\omega_l$. When going forward (backward), $\omega_l = \pm|\omega_l| = \omega_r$.

IV. A cooperative steering control law

A. Basic Theory

We consider a recent Frenet-Serret frame based algorithm for cooperative steering control in the presence of obstacles due to Morgan and Schwartz [14], inspired by the original work of Justh and Krishnaprasad [15]. The motion of a vehicle is described by a differentiable curve $z(s) \in \mathbb{R}^2$ parameterized by arc length. Let x denote the unit vector from the position of the vehicle, in the direction of the tangent vector dz/ds and $y = x^{\perp}$ is positively oriented with respect to x. The motion of each vehicle is modeled by

$$\dot{z}_k = x_k, \quad \dot{x_k} = u_k y_k, \quad \dot{y_k} = -u_k x_k \tag{6}$$

where k is the vehicle index. The vehicles move at unit speed and the curvature of the k^{th} vehicle path is the scalar u_k. The control law is specified by dynamically changing u_k to create pairwise interactions between vehicles. Define $u_k = \sum_{j \neq k} u_{jk}$ where

$$\begin{aligned} u_{jk} = [&-\eta\left(\frac{r_{jk}}{|r_{jk}|}\cdot x_k\right)\left(\frac{r_{jk}}{|r_{jk}|}\cdot y_k\right) \\ &-f(|r_{jk}|)\left(\frac{r_{jk}}{|r_{jk}|}\cdot y_k\right) + \mu x_j \cdot y_k] \end{aligned} \tag{7}$$

where $r_{jk} \equiv z_k - z_j$, $f(|r_{jk}|) = \alpha[1 - \left(\frac{r_0}{|r_{jk}|}\right)^2]$, and $\eta = \eta(|r|)$, $\mu = \mu(|r|)$, $\alpha = \alpha(|r|)$ are specified functions.

The term $-\eta\left(\frac{r_{jk}}{|r_{jk}|}\cdot x_k\right)\left(\frac{r_{jk}}{|r_{jk}|}\cdot y_k\right)$ aligns the vehicles perpendicular to their common baseline. The potential function $f(|r_{jk}|)$ regulates the spacing between the vehicles and the term $\mu x_j \cdot y_k$ steers the vehicles to a common orientation. This control law requires position information of other agents within a neighborhood as described below.

B. Local Coupling and Leader Following Control Law

Local coupling is manifested by limiting the visible distance range of each vehicle to a neighborhood around that vehicle. To ensure swarming, any two vehicles have to be within a specified distance of each other. The local coupling control law is

$$u_k^{GL} = \sum_{j \neq k} c(|r_{jk}|, 0, w)\, u_{jk} \tag{8}$$

where u_{jk} is (7) and

$$c(|r_{jk}|, q, w) = \begin{cases} 1 & if |r_{jk}| < w, \\ q & otherwise. \end{cases} \tag{9}$$

Such local coupling is advantageous for large numbers of agents due to the scalability of the communication step.

A designated leader vehicle steers a swarm in a particular direction. The rest of the vehicles (follower vehicles) follow accordingly, by using the local coupling control law, with stronger coupling between follower and leader vehicles.

$$u_k^{follower} = c\left(|r_{l(k)k}|, 0, w\right) l_c u_{l(k)k} \tag{10}$$
$$+ \sum_{j \neq k, l(k)} c(|r_{jk}|, 0, w)\, u_{jk}, \tag{11}$$

where l_c is a leader coupling constant and $l(k)$ is the index of the leader vehicle closest to the k^{th} vehicle. The control law for the leader vehicles depends on the particular application.

C. Homotopy Control Law

In order to transition smoothly from a global control law to a local coupling leader following control law, a homotopy parameter λ is introduced. The homotopy control law $u_k(\lambda)$, $0 \le \lambda \le 1$ satisfies

$$u_k(\lambda = 0) = u_k^G, \quad u_k(\lambda = 1) = u_k^L \tag{12}$$

where u_k^G is the global control law and u_k^L is the local control law. If there are m leader vehicles, the homotopy control law for the $n - m$ follower agents is

$$\begin{aligned} u_k^{follower}(\lambda) &= \\ c\left(|r_{l(k)k}|, 1-\lambda, w\right)&[(l_c - 1)\lambda + 1]u_{l(k)k} \\ &+ \sum_{j \neq k, l(k)} c\left(|r_{jk}|, 1-\lambda, w\right) u_{jk}, \end{aligned} \tag{13}$$

where u_{jk} is given by eqn. (7) and $l_c \gg 1$ is a coupling constant which strongly attracts the followers strongly to their leaders. The leader agent homotopy control law is

$$u_k^{leader}(\lambda) = \sum_{j \neq k}[u_{jk}(1-\lambda) + \frac{s_k}{n-1}\lambda]. \tag{14}$$

When more than one leader is present, local coupling can be exploited to separate a swarm into two sub-swarms, as leader vehicles drive in different directions and follower vehicles follow their respective closest leaders. Such an example is demonstrated on the testbed in the next section.

D. Target Seeking

To move towards a specified target, the kth vehicle uses

$$\begin{aligned} u_k = \sum_{j \neq k} &\left[c\left(|r_{jk}|, 0, w\right) \left(-\alpha \left(1 - \left(\frac{r_0}{|r_{jk}|} \right)^2 \right) \right) \right. \\ &\left. (r_{jk} \cdot y_k) \right] + \gamma\alpha \left(1 - \left(\frac{r_0}{|\overline{r}_k|} \right)^2 \right) (\overline{r}_k \cdot y_k), \end{aligned} \tag{15}$$

where $\overline{r}_k$ is the vector from the location of the kth agent to the target, and γ is a weighting constant. Only the first term involves interaction between the vehicles in order to avoid collisions. The second term directs each vehicle to move toward the target. This control law does not guarantee swarming, but if the agents start out in a swarm-like orientation, it is likely that they will stay together.

E. Barrier Avoidance

Consider a fixed convex object in the plane, the exterior of which is defined by a set of m points $b_i \in \mathbb{R}^2$. The average barrier direction vector is computed as

$$v_k = \begin{cases} \frac{\sum_{i=1}^m [(b_i - y_k) c(|b_i - y_k|, 0, w)]}{|\sum_{i=1}^m [(b_i - y_k) c(|b_i - y_k|, 0, w)]|}, \\ if \quad |\sum_{i=1}^m [(b_i - y_k) c(|b_i - y_k|, 0, w)]| \neq 0, \\ 0, otherwise, \end{cases} \tag{16}$$

where $c(\cdot)$ is a step cutoff function that is zero outside of a specified radius. The control law for barrier avoidance then consists of adding the term $\left(sign\left(v_k \cdot y_k^\perp\right)\right)(v_k \cdot y_k)$ to control law in (10). This term orients the vehicle perpendicular to v_k and the sign steers the vehicle away from the average barrier direction.

V. Implementation

In this section we consider several path planning strategies that build on the above control laws. To date there have not been many published experimental studies using this class of control laws. We mention a related paper [16] that implements a curvature-based steering control law for following boundaries of walls and is demonstrated with a single agent. A companion paper [17] to ours develops a control law for motion camouflage using the framework of [15] with implementation on our new testbed. In the examples below we consider both single agent and multi-agent tasks, including ones that rely on the range sensors to identify obstacles for avoidance.

A. Steering Angle

To map the curvature control u_k to a car's desired steering angle ϕ_k, we use the formula $\rho = L_{car}/\tan\phi_k$ [18], where ρ is the car turning radius. Thus, the steering angle of a car can be calculated as:

$$\phi_k = \tan^{-1}(L_{car}/\rho) = \tan^{-1}(L_{car}u_k). \tag{17}$$

L_{car} is measured to be 4 cm. The steering law assumes the cars to have unit speed thus we scale L_{car} according to the actual vehicular speed.

B. Implementation of a Basic Circle Tracker

This example has the car follow the circumference of a circle of given radius about a particular point on the test bed. We use a two-car model in which one of the cars is fixed at the center of the circle. We set $\mu = 0$, $\alpha = \eta = 1$, r_0 equal to the circle radius r and $|\mathbf{r}|$ equal to the distance from the car to the center of the circle. The basic control law reduces to

$$u = \eta(|r|)(\frac{-r}{|r|}x)(\frac{-r}{|r|}y) - f(|r|)(\frac{-r}{|r|} \cdot y)$$

where $r = r_2 - r_1$, the distance between the cars. Figure 7 shows an implementation of this basic circle following control law, on the testbed platform. A single car follows three different circular paths in sequence with a change in r upon reaching a particular heading.

C. Implementation of Homotopy Control Law

Due to the testbed physical size constraints, the homotopy control law discussed above is modified to a sequence of three stages. In stage one, two leaders are designated, with steering program $u_k^{leader}(\lambda = 0)$ given by eqn. (14) which is reduced to global control law eqn. (7). The regular global control law, which in this case includes only the leaders. Four followers are given control laws $u_k^{follower}(\lambda = 0)$ from (13). In stage two, the leaders are switched to $u_k^{leader}(\lambda = 1)$

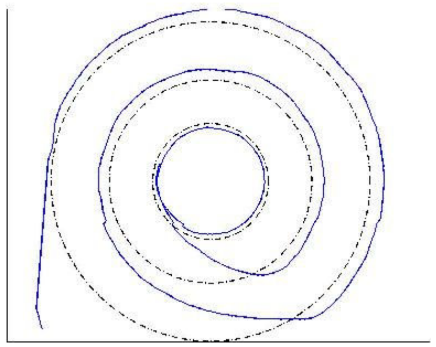

Fig. 7. A single car path (solid line) tracing, in sequence, concentric circles (dashed lines) of radii 65.5cm, 41.7cm, and 28.3cm.

where s_k is an explicit non-interactive control law causing the leaders to drive away from each other. The control law of the followers remains the same, causing a propagation of two swarms led by separate leaders. In stage 3, the global control law eqn. (7) between the leaders is reinstated, causing the two swarms to merge back together. Note that the control law (7) does not define the direction of the swarm, the heading direction is arbitrary and the outcome of the re-merging of the group can result in a different overall heading after merging, as seen in the two experiments shown in Figures 8.

D. Implementation of Target seeking, Dynamic Barrier Detection and Avoidance

We combine the target seeking [IV-D], barrier avoidance [IV-E], and cumulative sum algorithm [19] for obstacle detection to generate a path dynamically. A box $(W23 \times L6 \times H13)$ cm is placed along the path of a swarm of four cars moving toward a common target. All cars have prior knowledge of the box dimensions and orientation, but not its location. The box's widest surface is perpendicular to the cars' initial heading direction. We designate two front cars as observers. They use the on board long range infrared sensor to estimate the box location. Once the obstacle is located, data is sent to the computing station, which generates a virtual barrier. This information is distributed to all four cars. The virtual barrier is constructed as follows: let $\vec{p_i}$ be the point on the obstacle detected by the i^{th} observer. A rectangle of length $(2L_{obstacle} - |\vec{p_1} - \vec{p_2}|)$ and width $2W_{obstacle}$ is constructed with the center of the barrier side, closest to the swarm, located at $(\vec{p_1} + \vec{p_2})/2$. Since we only have two measures of distance to the obstacle, we extend the barrier to ensure collision avoidance. To avoid crossing paths and collision between cars after passing the obstacle, we reduce the barrier term weight when the car passes a certain distance from the barrier. Figure 9 shows the trajectories and snapshots of the implementation.

Cumulative Sum Algorithm For Obstacle Detection. As the observer approaches the obstacle, sensor readings increase from background noise to a level indicating the presence of the object. Figure 10 shows an example of raw sensor readings. To filter the signal, we use a particular version [19] of a standard cumulative sum algorithm [20]. Let X_n denote the raw sensor signal at time level n and μ denote the mean of the background noise when no obstacle is present. Define $Z_n = X_n - \mu - c$ where c is a fraction of the expected change in sensor reading due to the obstacle. Next, define $W_n = \max(0, Z_n + W_{n-1})$. The calculated value W_n should remain around zero until the change of state occurs, at which point it ramps up. An example is shown in Figure 10. Once W_n passes a designated threshold (large enough to avoid false alarms with a high probability) the object is detected. Using the car chassis at 1/5 of the full throttle, we test the cumulative sum algorithm for different values of c ranging from 150 to 400. The results are well-reproduced in multiple trials. These values lie closely on a linear fit, therefore we use

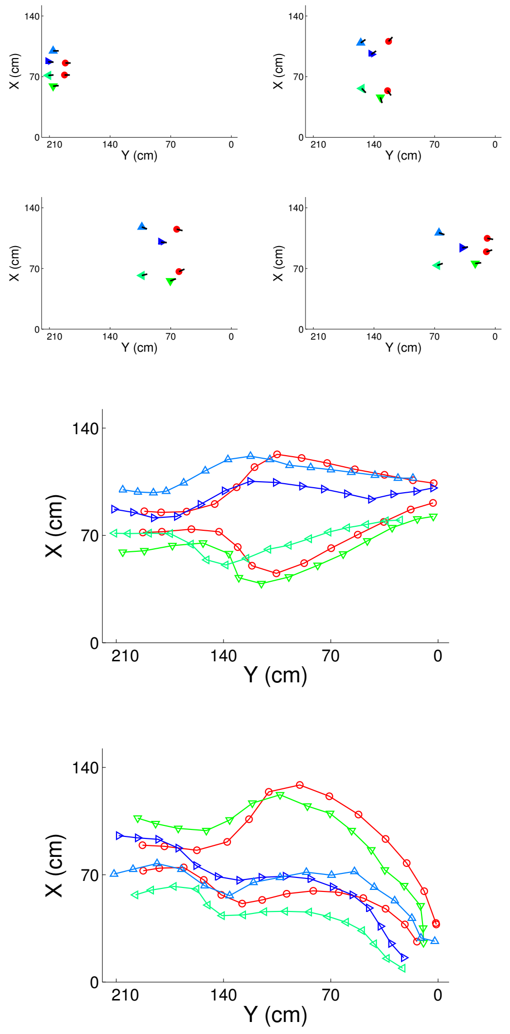

Fig. 8. The top four figures show testbed data of a time sequence of six cars performing the maneuver described in section V-C. The trajectories are shown in the middle figure. The circular dots show the position of the leaders; the triangular dots show the position of the followers. The bottom figure shows trajectories for second experiment in which one follower goes with the top leader and three go with the bottom leader. In this second run, the overall heading of the group is different after the merger.In the implementation, the tracking information is interpreted in pixel domain and the corresponding parameters are $\mu = \alpha = \eta = 1$,and $w = 600$ for every agents, $r_0 = 50$ for leaders and $r_0 = 40, l_c = 20$ for followers.

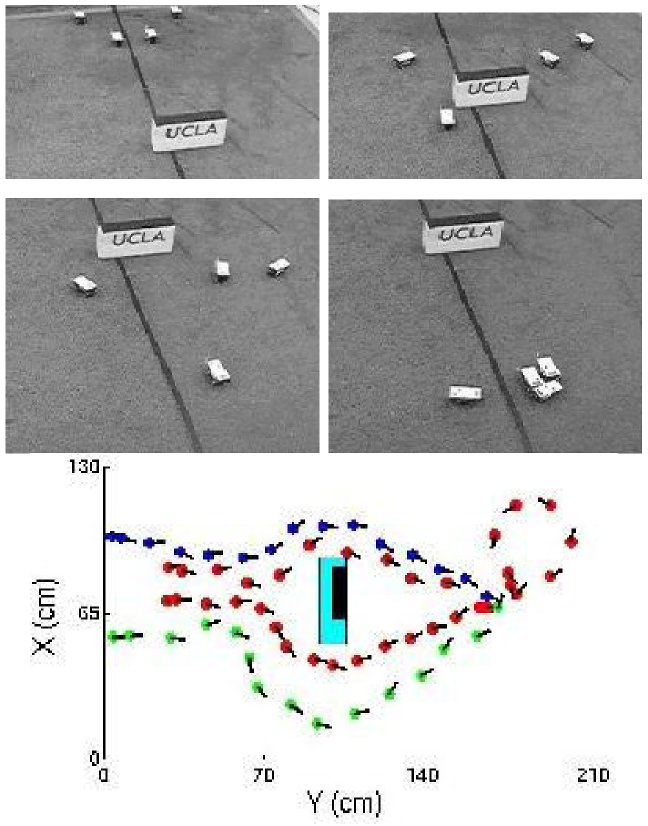

Fig. 9. Target seeking with barrier avoidance. Top four panels show snapshots, at different times, of a single demonstration of the maneuver. The time progresses from top left to bottom right. The bottom figures shows trajectories of the cars compared to both the actual barrier (dark) and the larger virtual barrier (light) as computed from the range sensors of the observers. In the implementation, the tracking information is interpreted in pixel domain and the corresponding parameters are $\alpha = 1$, $\gamma = 25$, $r_0 = 60$, and $w = 100$ for the target seeking term. The weighting constant for the barrier avoidance term is 85 and is reduced to 1 after passing the barrier with $w = 180$.

the $c = 200$ state in practice for the most advanced warning.

VI. Conclusion

The second generation testbed is a marked improvement over the the first version, with the addition of onboard processing and sensing, improved communication rate, vehicles diversity, and scalability. All upgrades are implemented while maintaining the low cost and micro-scale feature of the original testbed. The availability of two different vehicle platforms, enables a wider range of study in cooperative control. In this paper, we validate the effectiveness of several dynamic multi-agent cooperative control laws in which some modification of the algorithms are required to work with the hardware architecture.

Acknowledgments

We thank Alexander Tartakovsky for suggesting the the cumulative sum algorithm and Dave Morgan for useful comments about the path planning algorithm. This research is supported by ARO MURI grant 50363-MA-MUR, ONR grant N000140610059, and ARO grant W911NF-05-1-0112. In addition, AJ and VV are supported by NSF research training group grant DMS-0601395.

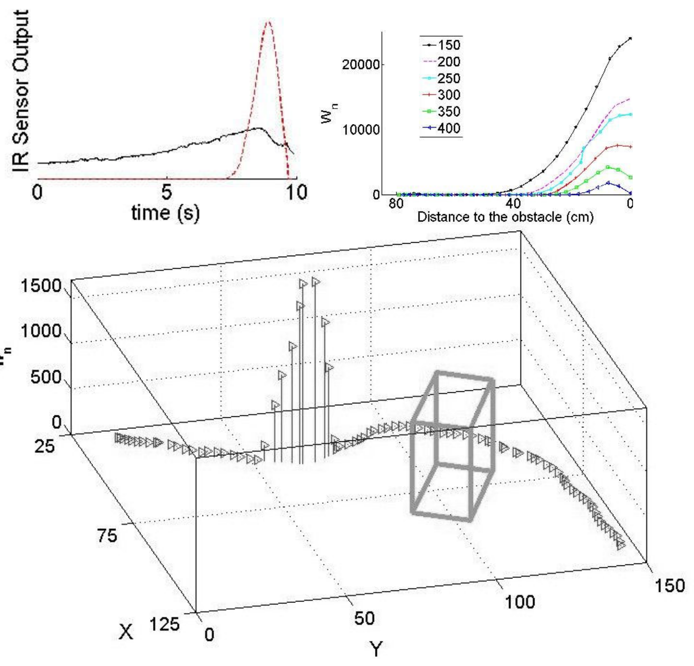

Fig. 10. Cumulative sum algorithm applied to sensor data from a single car approaching an obstacle. (top left) raw data and cumulative sum, (top right) cumulative sums for different choices of c, (bottom) sample car path avoiding obstacle with cumulative sum sensor output.

References

[1] C. H. Hsieh, Y. Chuang, Y. Huang, K. K. Leung, A. L. Bertozzi, and E. Frazzoli, "An Economical Micro-Car Testbed for Validation of Cooperative Control Strategies", *Proc. of the 2006 American Control Conference, Minneapolis*, MN, June 14-16, pages 1446-1451.

[2] T. Chung, L. Cremean, W.B. Dunbar, Z. Jin, E. Klavins, D. Moore, A. Tiwari, D. van Gogh, and S. Waydo, "A platform for cooperative and coordinated control of multiple vehicles: The Caltech Multi-Vehicle Wireless Testbed", *Proc. of the 3rd Conference on Cooperative Control and Optimization*, Dec. 2002.

[3] Z. Jin and S. Waydo and E. B. Wildanger and M. Lammers, H. Scholze and P. Foley and D. Held and R. M. Murray, "MVWT-II: The Second Generation Caltech Multi-vehicle Wireless Testbed", *Proc. of the 2004 American Control Conference*, pp. 5321-5326, 2004.

[4] E. King, Y. Kuwata, M. Alighanbari, L. Bertuccelli, and J. How, "Coordination and control experiments on a multi-vehicle testbed", *Proc. of the 2004 American Control Conference*, pp. 5315- 5320 , 2004.

[5] J. Spletzer, A.K. Das, R. Fierro, C.J. Taylor, V. Kumar, J.P. Ostrowski,"Cooperative localization and control for multi-robot manipulation",*Proc. of the 2001 Intelligent Robots and Systems*, pp. 631-636, 2001.

[6] T. W. Mclain and R. W. Beard, "Unmanned Air Vehicle Testbed for Cooperative Control Experiments",*Proc. of the 2004 American Control Conference*, pp. 5327-5331, 2003.

[7] R. D'Andrea, Michael Babish,"The RoboFlag Testbed",*Proc. of the 2003 American Control Conference*, pp. 656-660, 2003.

[8] A. Stubbs, V. Vladimerou, A. T. Fulford, D. King, J. Strick, and G. E. Dullerud," A hovercraft testbed for networked and decentralized control", *IEEE Control Systems Magazine*, pp. 56-69, June 2006.

[9] R. DAndrea, "Robot soccer: A platform for systems engineering", *Computers in Education Journal*, 10(1):5761, 2000.

[10] The Minnow Project at Carnegie Mellon University, http://www.cs.cmu.edu/ coral/minnow/.

[11] Robotic Embedded Systems Laboratory, Univ. of Southern California (http://robotics.usc.edu/ embedded/research/videos.html).

[12] Y. Landa, D. Galkowski, Y. R. Huang, A. Joshi, C. Lee, K. K. Leung, G. Malla, J. Treanor, V. Voroninski, A. L. Bertozzi and R. Tsai, ”Robotic Path Planning and Visibility with Limited Sensor Data”, to appear in the *2007 American Control Conference*.

[13] http://www.radiotronix.com,

[14] D. S. Morgan and I. B. Schwartz, ”Dynamic coordinated control laws in multiple agent models,” *Physics Letters A*, vol. 340, pp. 121-131, 2005.

[15] E. W. Justh and P. S. Krishnaprasad, “Equilibria and steering laws for planar formations”, *Systems and Control Letters*, vol. 52, pp. 25-48, 2004.

[16] F. Zhang, A. O’Connor, D. Leubke, and P. S. Krishnaprasad, “Experimental Study of Curvature-based Control Laws for Obstacle Avoidance”, *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, New Orleans, LA, April 2004.

[17] Carmeliza Navasca, Ani Asatryan, Vatche Attarian, Yuan R. Huang, Kevin K. Leung, Abhijeet Joshi, Vlad Voroninski, Meghdi Aboulian, and Krsytle McBride, “Implementations of control laws for motion camouflage,” preprint 2006.

[18] S. M. LaValle, “Planning Algorithms”, Cambridge University Press, 2006.

[19] A.G. Tartakovsky, B.L. Rozovskii, R. Blazek, H. Kim, ”Detection of intrusions in information systems by sequential change-point methods,” *Statistical Methodology*, vol. 3, Issue 3, pp. 252-340, 2006.

[20] M. Basseville and I.V. Nikiforov, Detection of Abrupt Changes: Theory and Applications. Prentice Hall, Englewood Cliffs, 1993.