John R. Spletzer
Camillo J. Taylor
GRASP Laboratory
University of Pennsylvania
Philadelphia, PA 19104, USA
spletzer@grasp.cis.upenn.edu
cjtaylor@grasp.cis.upenn.edu

Dynamic Sensor Planning and Control for Optimally Tracking Targets

Abstract

In this paper, we present an approach to the problem of actively controlling the configuration of a team of mobile agents equipped with cameras so as to optimize the quality of the estimates derived from their measurements. The issue of optimizing the robots' configuration is particularly important in the context of teams equipped with vision sensors, since most estimation schemes of interest will involve some form of triangulation.

We provide a theoretical framework for tackling the sensor planning problem, and a practical computational strategy inspired by work on particle filtering for implementing the approach. We then extend our framework by showing how modeled system dynamics and configuration space obstacles can be handled. These ideas have been applied to a target tracking task, and demonstrated both in simulation and with actual robot platforms. The results indicate that the framework is able to solve fairly difficult sensor planning problems online without requiring excessive amounts of computational resources.

KEY WORDS—optimal target tracking, sensor fusion, particle filtering

1. Introduction

The idea of using teams of small, inexpensive robotic agents to accomplish various tasks is one that has gained increasing currency in the field of robotics research. Figure 1 shows a picture of a Clodbuster robot which is based on a standard remote controlled motion platform and is outfitted with an omnidirectional video camera—its only sensor. Using teams of these modest robots, fairly sophisticated applications, such as distributed mapping, formation control and distributed manipulation, have been successfully demonstrated (Alur et al. 2000; Spletzer et al. 2001).

One of the more interesting aspects of these platforms is that estimates for relevant quantities in the world are formed by combining information from multiple distributed sensors. For example, the robots in the team shown in Figure 1 obtain an estimate for their relative configuration by combining the angular measurements obtained from all of the omnidirectional images and performing a simple triangulation operation. Similar techniques can be used to estimate the locations of other features in the environment, such as the box they are manipulating. In fact, we could choose to view the team in Figure 1 as a three-eyed stereo rig where the individual eyes can actually be moved on the fly.

This capability invites the following question: given that the robot platforms are mobile, how should they be deployed in order to maximize the quality of the estimates returned by the team? This is a particularly important question in the context of robots equipped with vision sensors, since most of the estimation techniques of interest in this case are based on some form of triangulation.

Similar questions arise when we consider the problem of integrating information from a sea of distributed sensors. Given that there is some cost associated with transmitting and processing data, which sensor readings should we use to form an estimate for the parameters of interest?

In this paper we present a theoretical framework for discussing such questions and a practical computational approach, inspired by work on particle filtering, for tackling them. The suggested approach could be viewed as an application of the theory of games since the problem of controlling the configuration of the robots is reformulated as the problem of optimizing a quality function that reflects the expected value of assuming a particular formation. Results obtained by applying this approach to a target tracking task are presented in Section 3.

It is important to note that while the approach was developed to handle the problems faced by teams of robots equipped with vision sensors, it could also be used to deploy robots equipped with other types of sensors, such as laser range finders or sonar systems.

The International Journal of Robotics Research
Vol. 22, No. 1, January 2003, pp. 7-20,
©2003 Sage Publications

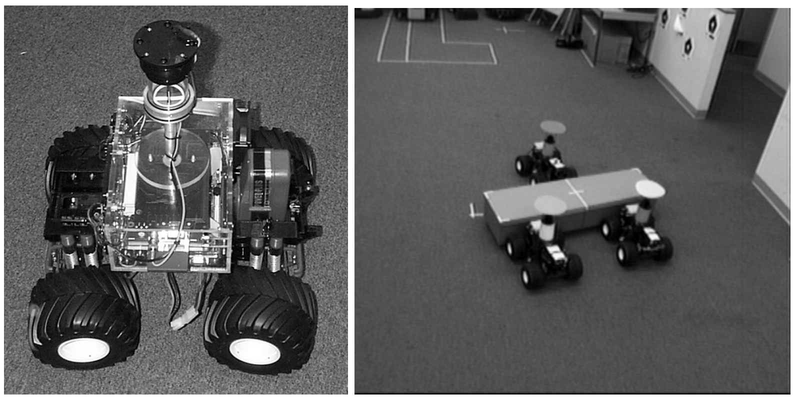
Fig. 1. A single Clodbuster robot (left) and the team performing a distributed manipulation task.

1.1. Related Work

The focus of this research is a probabilistic framework which exploits the degrees of freedom afforded by robot mobility to actively manage sensor positions for improved state estimation. We demonstrate its effectiveness in an optimal target tracking task. “Optimal tracking” can be defined using various metrics. We choose to minimize the expected error in tracking target positions. Since the measurements of multiple robots are combined to estimate target pose, this relates strongly to work in sensor fusion.

In our target tracking task, robots rely on omnidirectional cameras for tracking groups of targets. Merging measurements from multiple vision sensors for improved state estimation was considered by Bajcsy and others under the heading of *Active Perception*. Improvements were seen in various performance metrics, including ranging accuracy (Bajcsy 1988; Krotkov and Bajcsy 1993). Our framework can be viewed as an extension of this paradigm to distributed mobile robotics.

Durrant-Whyte and co-workers pioneered work in sensor fusion and robot localization. This yielded significant improvements to methods used in mobile robot navigation, localization and mapping (Majumder, Scheding, and Durrant-Whyte 2001; Dissanayake et al. 2001). Thrun and co-workers have also contributed significant research to these areas (Thrun 2001; Thrun et al. 2000). The work of both groups has emphasized probabilistic techniques for data fusion—with a recent focus on particle filtering methods. Our approach is also probabilistic, and it too leverages particle filtering methods. However, our work distinguishes itself from traditional data fusion techniques in that the sensors themselves are actively managed to improve the quality of the measurements obtained *prior* to the data fusion phase, resulting in corresponding improvements in state estimation.

Since the sensors are actively managed, our work relates to research in on-line sensor planning as well. Relevant to our approach was a methodology for distributed control proposed by Parker (1999). This framework—Cooperative Multi-Robot Observation of Multiple Moving Targets (CMOMMT)—attempted to maximize the collective time that each target was being observed by at least one robot.

The theory of games has also provided inspiration for similar research in target tracking. The pursuit-evasion problem was investigated by LaValle et al. (1997) and Fabiani et al. (2001). Both examined the task of maintaining target visibility in a cluttered environment known a priori to the pursuer. LaValle’s approach generated trajectories that minimized a loss function which grew when the target became occluded. Fabiani’s motion strategy was based on the expected maximum value of a corresponding utility function U. Of the two, Fabiani’s work is more relevant. The value of U was affected by uncertainty in the target’s position, which was indirectly affected by uncertainty in the pursuer’s position. As a result, the pursuer trajectory was influenced not just by target position, but also by known landmark positions in the environment which could be used to reduce uncertainty in target pose.

In all three of these cases, the optimization criterion was based on maintaining target observability, rather than the quality of the observation. Additionally, the work of LaValle and Fabiani was limited to the case of a single pursuer/evader. In theory, both could be extended to multiple agents. However, in practice the resulting explosion in computational complexity would be prohibitive. In contrast, the complexity of our framework is implementation-dependent, and may be tuned by the user to scale very efficiently in terms of the number of robots and targets.

In the next best view (NBV) problem, sensor placement is of primary concern (Pito 1999; Stamos and Allen 1998). Given, for example, previous range scans of an object, an NBV system attempts to determine the next best position of the scanner for acquiring the object’s complete surface geometry. As in our framework, the emphasis is optimizing sensor placement. However, NBV is intended for use in a static environment. Inherent in our approach is the ability to handle dynamic scenes which makes it more akin to a control law for distributed sensors.

2. Developing the Framework

2.1. A Theoretical Approach

In this section we describe the theoretical framework used to discuss the problem of sensor deployment. In order to ground the terminology, we describe how various elements in the framework relate to the scenario depicted in Figure 2. In this example, three robots are tasked with localizing a single moving target.

Let $\mathcal{C}_r$ denote the configuration space of the robot platforms. In this case, we can consider the vector formed by concatenating the positions and orientations of the three platforms with respect to some base frame of reference $[x_1, y_1, \theta_1, x_2, y_2, \theta_2, x_3, y_3, \theta_3]^{\mathrm{T}}$. Let $\rho \in \mathcal{C}_r$ denote an element of this configuration space. Similarly, let $\mathcal{C}_w$ denote the configuration space of the target parameters under consideration. In Figure 2, this space is particularly simple since we need only consider the position of the moving target with respect to the base frame denoted by the vector $[x_t, y_t]^{\mathrm{T}}$. In general, however, this space can be much more complicated. Let $\omega \in \mathcal{C}_w$ denote an element of this configuration space.

Let $\hat{Z}$ represent the set of all possible sensor measurements, and let $\hat{z} \in \hat{Z}$ denote our measurement vector. In this example, $\hat{z}$ corresponds to the vector formed by concatenating the three angles measured by the robots $[\alpha_1, \alpha_2, \alpha_3]^{\mathrm{T}}$. The hat serves to remind us that these measurements are corrupted by noise. It is assumed that the designer has some model for the noise process, which is given in the form of a conditional probability density function $P(\hat{z}|\rho, \omega)$. This function allows us to predict the distribution of measurements as a function of ρ and ω.

Let $Est : \mathcal{C}_r \times \hat{Z} \to \mathcal{C}_w$ denote a function that produces an estimate of the target's position, $\hat{\omega}$, from the noisy measurements, $\hat{z}$, and the robots' configuration, ρ. $Disp : \mathcal{C}_w \times \mathcal{C}_w \to \mathcal{R}^+$ is a function which returns an indication of the disparity between an estimated value $\hat{\omega}$ and the actual value ω. For our target tracking example, an appropriate function might be the Euclidean distance $Disp(\omega, \hat{\omega}) = ||\omega - \hat{\omega}||_2$. Other applications may require more sophisticated error metrics.

$P(\omega)$ denotes a probability density function on the configuration space $\mathcal{C}_w$ which can be used to model prior information about the values of the parameters of interest. For example, we may have some information about where the target could be based on prior measurements or a dynamic model for the target's motion. Given this terminology, we can define a quality function $Q(\rho)$ as follows:

$$Q(\rho) = \int_{\mathcal{C}_w} \int_{Z} Disp(\omega, Est(\rho, \hat{z})) P(\hat{z}|\rho, \omega) P(\omega) \, \mathrm{d}\hat{z} \, \mathrm{d}\omega. \tag{1}$$

This function captures how the expected error in the estimate, $\hat{\omega}$, varies as the configuration of the robots changes. Note that there are, of course, several alternative definitions for this quality function that are equally reasonable. We could consider the maximum expected error in the estimate or the median expected error. Different choices for $Q(\rho)$ may be more appropriate in certain situations.

With these notions in place, we can formulate the problem of choosing an appropriate configuration for the robots as an optimization problem as follows:

$$\min_{\rho \in \Delta} Q(\rho). \tag{2}$$

The goal in this case is to find a choice of $\rho \in \Delta$, where $\Delta \subset \mathcal{C}_r$, which minimizes the quality function $Q(\rho)$. Limiting the optimization to a subset of $\mathcal{C}_r$, Δ, allows us to model situations where certain configurations cannot be achieved due to obstacles in the environment, sensor constraints or limitations on the range of motion of the robots.

Note that, even though the approach is being discussed in the context of target tracking, the framework is general enough to be applied to a wide range of sensor planning problems. The specifics of the task would be reflected in the definitions of $\mathcal{C}_r$, $\mathcal{C}_w$, $\hat{z}$, *Est* and *Disp*.

2.2. A Computational Solution

For most interesting systems, the optimization problem given in eq. (2) is difficult to solve analytically. However, it is possible to approximate this process computationally. To do this we draw inspiration from prior work on particle filtering (Isard and Blake 1998).

In particle filtering, probability distributions such as $P(\omega)$ are approximated by sets of tuples (ω_j, π_j), where ω_j is a single sample from $\mathcal{C}_w$ and π_j is a weight that reflects the likelihood of ω_j representing the state ω. By making use of this approximation, we can replace the integrals of eq. (1) with a summation:

$$Q(\rho) \approx \frac{1}{N} \sum_{j=1}^{N} Disp(\omega_j, Est(\rho, \hat{z})). \tag{3}$$

where N corresponds to the number of samples selected at random from $P(\omega)$ and $P(\hat{z}|\rho, \omega)$. By choosing N sufficiently large, this single summation can reflect the effects of both integrations in eq. (1).

The computation of $Q(\rho)$ is outlined in Algorithm 1. For a more concrete explanation, we refer to Figure 3. To estimate $Q(\rho)$, a sample ω is chosen at random from $P(\omega)$, and projected from target space to each robot's image frame (Figure 3(a)). The corresponding image measurements are then corrupted with noise from our sensor model, and projected back into target space using our *Est* function (Figure 3(b)). We can obtain the disparity between this new sample $\hat{\omega}$ and

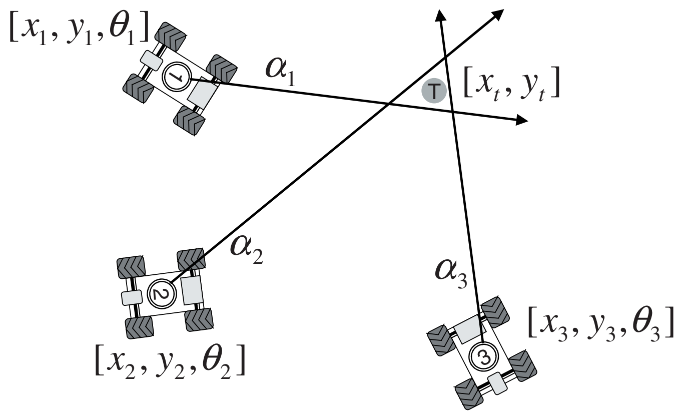

Fig. 2. Target localization by a robot team.

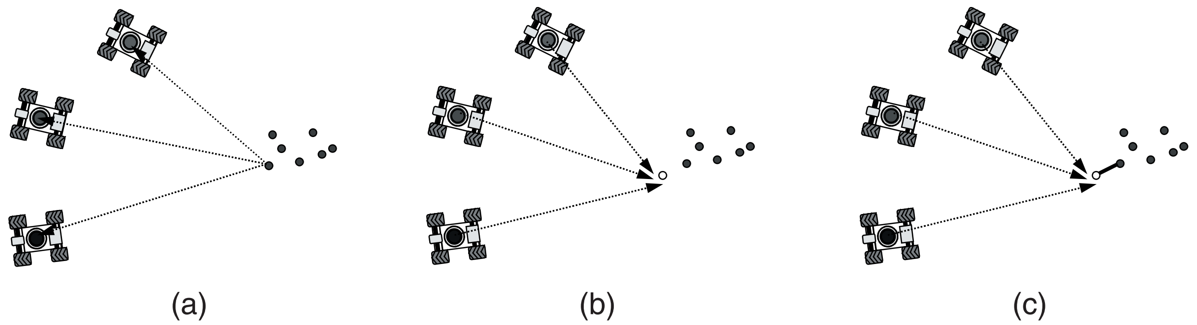

Fig. 3. Evaluating $Q(\rho)$. A sample ω is taken randomly from $P(\omega)$ and projected into the sensor frames (a), corrupted with noise from our sensor model and reprojected as $\hat{\omega}$ (b), so that $Disp(\omega, \hat{\omega})$ can be calculated (c). This is repeated for N random samples.

our original sample from $Disp(\omega, \hat{\omega})$ (Figure 3(c)). This procedure is repeated for N samples. Using $||\omega - \hat{\omega}||^2$ as our *Disp* function, $Q(\rho)$ then reflects the expected mean-squared error in target position.

A simple but effective approach to optimizing the robot configuration is to approximate the gradient of the quality function, $\frac{\partial Q}{\partial \rho}$, by sampling its value in the vicinity of the current robot configuration. The control law governing robot motion would then be

$$\dot{\rho} \propto -\frac{\partial Q}{\partial \rho}. \tag{4}$$

Gradient-based approaches can often lead to local minima—and consequently sub-optimal performance—in traditional optimization problems, and local minima do in fact exist when tracking multiple targets. However, in a dynamic environment $Q(\rho)$ is actually redefined whenever the estimated positions of the targets change with respect to one another. As a result, we often have only a single time-step for optimizing a given $Q(\rho)$, and in such cases can generate "piecewise optimal" trajectories using a gradient approach. Alternatively, we could employ standard optimization techniques, such as the Simplex method (Press et al. 1993), to choose the best achievable robot configuration in the vicinity for the next time instant.

Algorithm 1. Calculate $Q(\rho)$ for a given $P(\omega)$

```
  $err \Leftarrow 0$
  **for** $i = 1$ to N **do**
    $\omega_i \Leftarrow\in P(\omega_i)$
    $\{$Choose a sample at random from $P(\omega_i)\}$
    $\hat{z}_i \Leftarrow\in P(\hat{z}|\rho, \omega)$
    $\{$Choose a sample at random from $P(\hat{z}|\rho, \omega)\}$
    $\hat{\omega}_i \Leftarrow Est(\rho, \hat{z}_i)$
    $\{$Generate an estimate for $\hat{\omega}\}$
    $err \Leftarrow err + Disp(\omega_i, \hat{\omega}_i)$
  **end for**
  $Q \Leftarrow \frac{err}{N}$
```

Note that it is possible to incorporate knowledge of system dynamics into this framework in the usual manner. In the

CONDENSATION algorithm described by Isard and Blake (1998), a particle distribution $P(\omega)$ is propagated at each time-step according to a known dynamic model. This same $P(\omega)$ serves as the assumed input for our framework, and establishes a complementary relationship between sensing and control, as the same particle sets used for tracking targets are also used to control the robot team for improving future tracking estimates.

3. Experimental Results

3.1. Simulation Experiments

In order to demonstrate the utility of the proposed framework, we first apply it to three sensor planning problems in simulation: tracking a single point target; tracking multiple point targets; and tracking a box. We then extend the point target tracking problem by incorporating a dynamical model for the target. Finally, we integrate motion planning techniques for local obstacle avoidance and we demonstrate target tracking in a cluttered workspace. Each of these scenarios is explained in more detail below.

We have assumed in all of these scenarios that the robots can accurately measure their positions and orientations with respect to one another, since it is the robot positions relative to the targets that are of interest. Note that we could consider the error in the positioning of the robots within this framework by adding extra noise terms to the measurements or by including the configuration of the robots as part of the state to be estimated.

3.1.1. Tracking a Single Point Target

For the first scenario, we consider two robots equipped with omnidirectional cameras, and tasked with tracking a single target. $\mathcal{C}_r$ represents the concatenation of the robot positions, $\mathcal{C}_w$ the target position, and $\hat{z}$ the two angles to the target measured by the members of the robot team. We assume $\hat{z}$ to be corrupted with random noise generated from our sensor model. $Est(\rho, \hat{z})$ returns an estimate for the target position, $\hat{\omega}$, which minimizes the squared disparity with the measurements, $\hat{z}$, and $Disp(\omega, \hat{\omega})$ simply returns the Euclidean distance between the estimated target position and the actual value.

In our simulations, robot motions are constrained by the maximum robot velocity and the robot positions are limited by mandating a minimum standoff distance to the target. These serve to define the valid configuration space for the robots, Δ. Below, we provide results from Matlab simulations for two robots with both static and dynamic targets. For these trials, 100 exemplars were used to approximate $P(\omega)$, and the sensor model (for all trials) was assumed to be Gaussian noise with $\sigma = 1°$.

Figure 4 shows the static target case for two robots. Trajectories for this symmetric case are predictable and consistent with simulation results, as are the dramatic drops in estimation error over time. Similar results are obtained for the case of an unpredictably moving target, as shown in Figure 5 (Extension 1).

3.1.2. Tracking Multiple Point Targets

For the second scenario, we examine the more interesting problem of n robots tracking m independently moving, unpredictable point targets. This problem can be tackled in much the same manner. $\mathcal{C}_w$ now represents the concatenation of possible target positions, and $\hat{z}$ the corresponding $n \times m$ angles measured from robots to targets. $Est(\rho, \hat{z})$ approximates the position of every target, and $Disp(\omega, \hat{\omega})$ returns the summed disparities between estimated and true target positions.

The results from a pair of simulation runs can be found in Figures 6 and 7 (Extension 2). In these trials, three unpredictable targets were tracked by five and four robots, respectively.

Note the behavior of the robots as they move from their original positions to more advantageous vantage points. The robots automatically split off to track targets without any need for sophisticated switching rules for arbitrating robot–target assignment. The final configuration is simply a consequence of the definition of the $Q(\rho)$ function that the system attempts to optimize. Note also that it is not possible in these scenarios to assign two robots to every target, so the robots distribute themselves automatically to come up with the best composite estimate. This is significant, as relatively complex tracking behaviors can be implicitly encoded into $Q(\rho)$, and the need for explicit switching controllers is mitigated.

3.1.3. Tracking a Box

For the third case, we consider the problem of using the measurements from the robots to estimate the configuration of a box in the scene. This example demonstrates how the proposed framework can be applied to scenarios where the state estimate is not simply the concatenation of a set of point locations. Here the configuration space $\mathcal{C}_w$ is identified with $SE(2)$ and elements of this set denote the position and orientation of the box. The robots can measure the angles to all of the visible box corners, $\hat{z}$. The estimation function $Est(\rho, \hat{z})$ as always is nonlinear, and minimizes the disparity between the predicted angles to the visible corners and the actual measurements. $Disp(\omega, \hat{\omega})$ returns the sum of the distances between the recovered box corners and the actual corners. For these trials, 20 "box" exemplars were used to estimate $P(\omega)$. Sample simulation results can be found in Figures 8 and 9.

In both cases we can see that the robots not only migrate to positions more advantageous for tracking the corner features, but also for maximizing the *number* of visible features. The latter effect is a result of the *Est* function using only the visible corners to estimate the box pose. Inherently better estimates are obtained when more features are available.

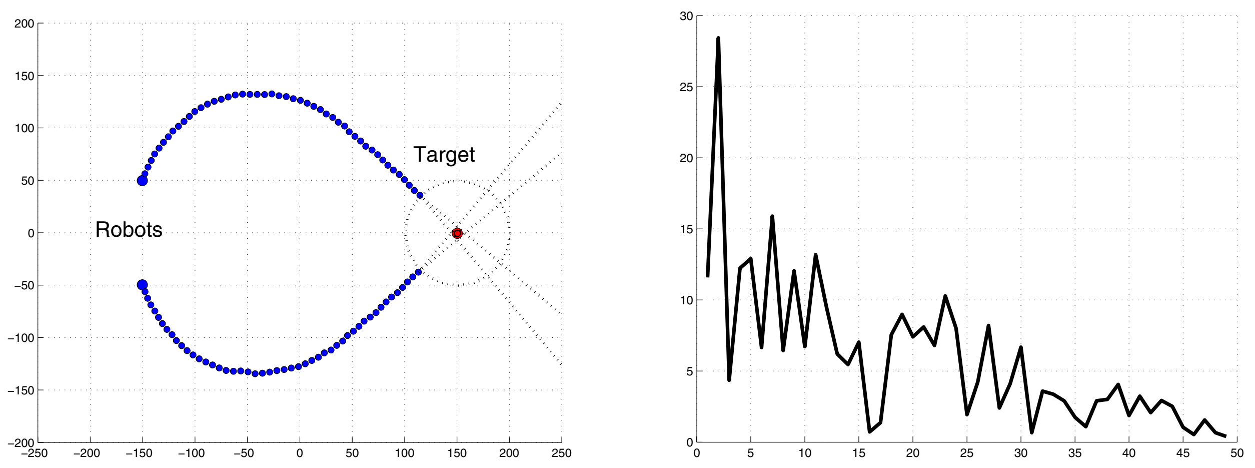

Fig. 4. Generated trajectories (left) and disparity measurements (right) for two robots tracking a static point target.

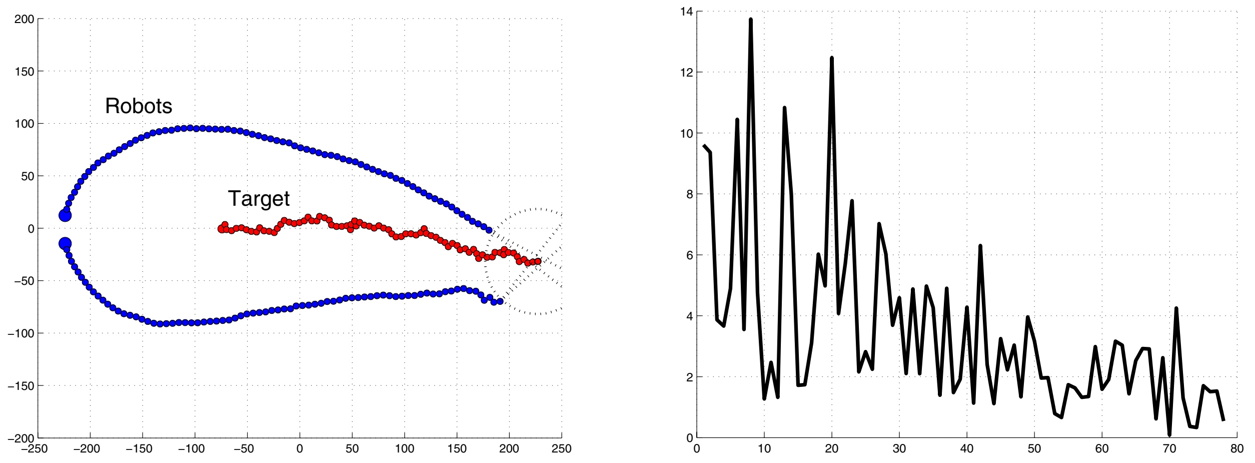

Fig. 5. Generated trajectories (left) and disparity measurements (right) for two robots tracking an unpredictable moving point target.

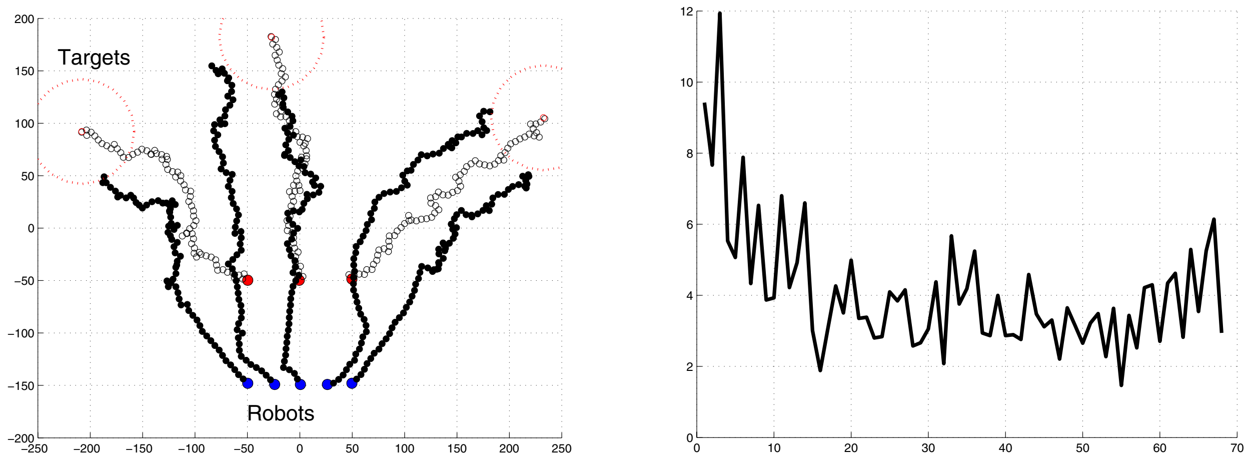

Fig. 6. Generated trajectories (left) and summed disparity measurements (right) for five robots tracking three point targets.

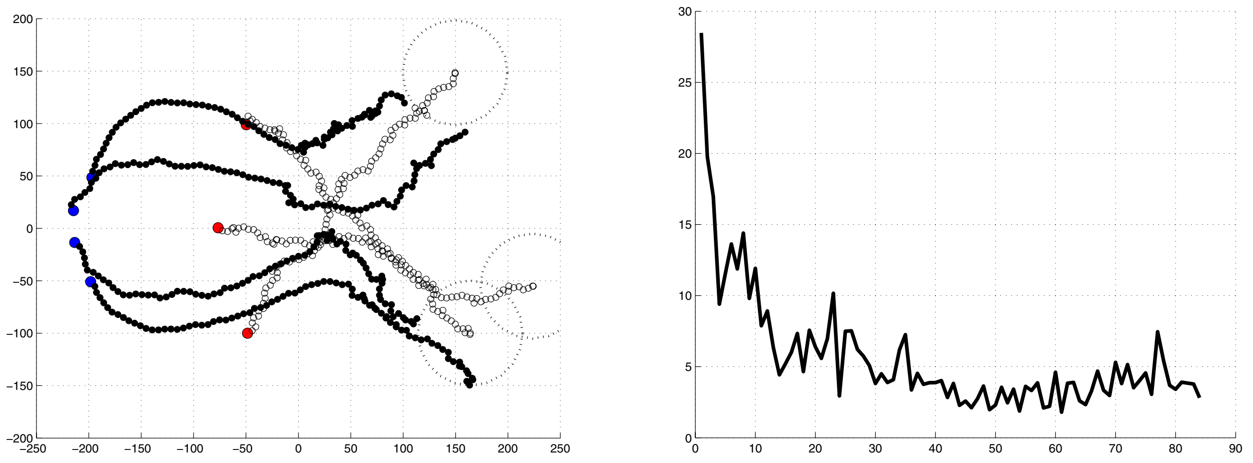

Fig. 7. Generated trajectories (left)and summed disparity measurements (right) for four robots tracking three point targets.

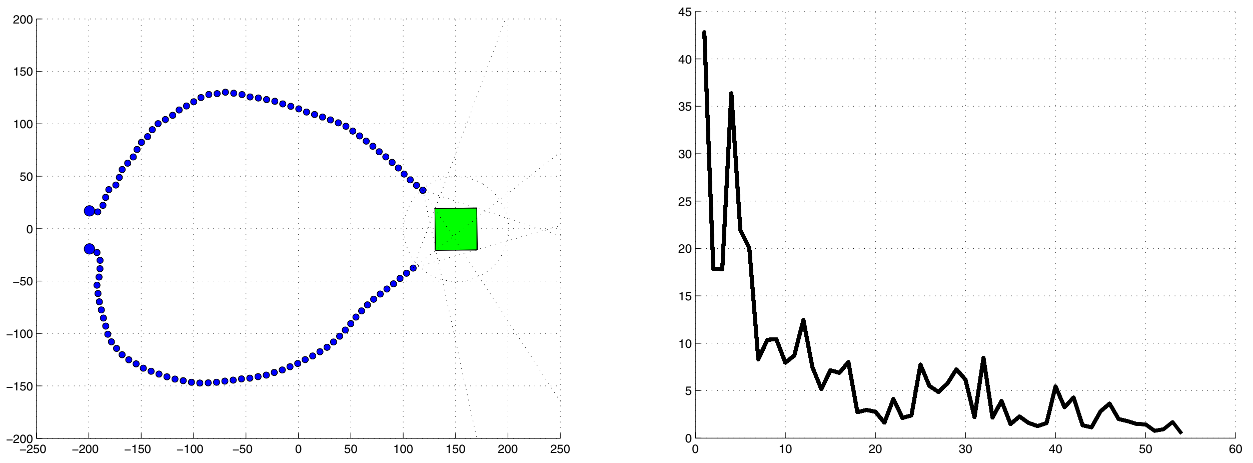

Fig. 8. Generated trajectories (left) and summed disparity measurements (right) for two robots tracking a static box.

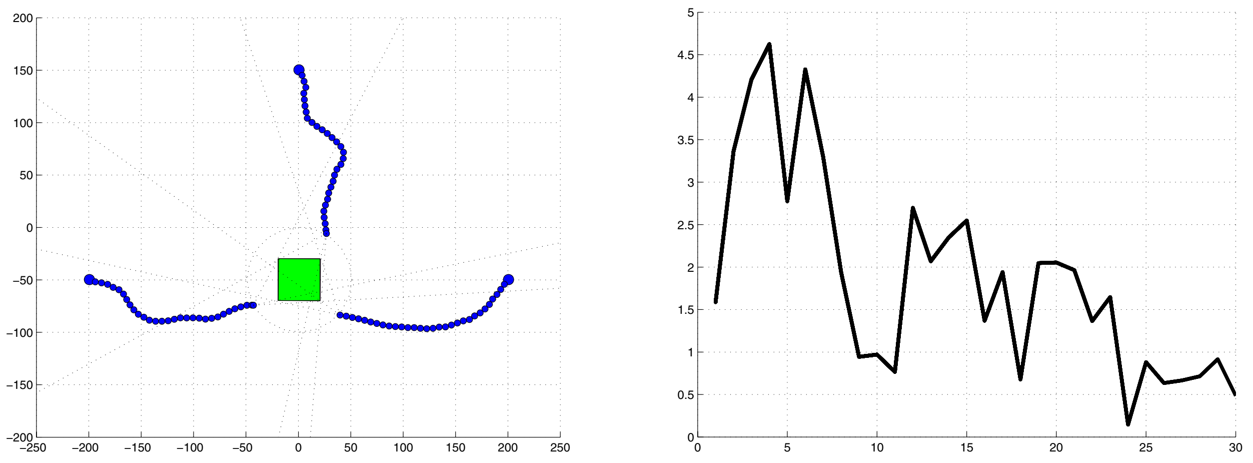

Fig. 9. Generated trajectories and summed disparity measurements for three robots tracking a static box.

3.1.4. Incorporating the Dynamical Model

Integrating target dynamics into sensor planning often provides significant improvements in tracking performance. Dynamical models can be obtained using an approximation of target dynamics, or through "learned" models as demonstrated in Isard and Blake (1998). For our simulations, we employed the former approach.

Consider the case of n observers on the ground tracking a ball traveling through the air with some unknown initial velocity V_t. We again model these observers as robots equipped with omnidirectional cameras. In this case, $\mathcal{C}_r$ represents the concatenation of the robot positions, which are constrained to operations in the x–y plane, and $\mathcal{C}_w \subset R^3$ represents the space of target positions. The measurement vector $\hat{z}$ denotes the n azimuth and elevation angle pairs to the target measured by members of the robot team.

We approximated the dynamical model for the ball $\frac{d\hat{\omega}}{dt}$ by assuming constant acceleration under gravity g and estimating its velocity from position measurements over time. Actual ball dynamics $\frac{d\omega}{dt}$ in the simulation were slightly more realistic and also modeled drag effects. The corresponding equations of motion are given in eqs. (5) and (6), where μ is density, and m, d, C_D and v_0 are the mass, diameter, drag coefficient and current velocity of the ball, respectively:

$$\frac{d\hat{\omega}}{dt} = v_0 t + \frac{g t^2}{2} \tag{5}$$

$$\frac{d\omega}{dt} = v_0 t + \frac{\pi d^2 t^2 (4\mu_{ball} g d - 3 C_D \mu_{air} v_0^2)}{24m}. \tag{6}$$

As before, $P(\omega)$ was initially approximated by a randomly generated set of exemplars that were constrained to lie within the intersection of the sensor error cones, and all of the particles were given equal weight. The distribution was then propagated using standard particle filtering techniques. At each time step t, deterministic drift was applied to $P(\omega)$ based on this dynamic model, followed by stochastic diffusion to account for model uncertainty. We sampled $P(\omega)$ at this point, as it allowed the robot pose ρ_{t+1} to be optimized over the expected target position ω_{t+1}—*not* over the current target position ω_t. That is, the robots moved so as to optimize their ability to localize the target at the next time instant based upon where they predicted the target would move.

In our simulations, robot motions were constrained by the maximum robot velocity $V_r \ll V_t$. This served to define the limits of the set over which the optimization occurs, Δ. Results from a sample Matlab simulation for three robots are provided below. For this trial, 100 exemplars were used to approximate $P(\omega)$, and the sensor model was assumed to be Gaussian noise with $\sigma = 1°$.

Figure 10 (Extension 3) shows a representative simulation run of three robots tracking a single target. Robot trajectories are inefficient from a "distance-traveled" point of view, as they attempt to optimize position estimates over the target's entire flight rather than its endpoint. Figure 11 shows the error in measured target position for the same target trajectory from both stationary (dashed line) and moving (solid line) observers. When viewed in this light, the benefits of the otherwise curious robot trajectories become readily apparent. Reductions in measurement errors by a factor of 4 to 5 over the stationary case clearly demonstrate the effectiveness of the integrated optimization/dynamical modeling approach.

3.1.5. Tracking Targets in a Cluttered Workspace

In the simulation results we have presented thus far, constraints to $\mathcal{C}_r$ were limited solely to pursuer dynamics and a mandatory target standoff distance. This is adequate for operations in an uncluttered workspace, but does not handle the more generic case where obstacles are present. To address the resulting additional constraints on $\mathcal{C}_r$ (and $\mathcal{C}_w$), we assumed that the robots were able to obtain accurate information about obstacles in their immediate vicinity. This was consistent with our approach of generating locally optimal trajectories, and did not require a priori information of obstacle locations or a global map of the environment. Δ was then defined by the local obstacle-free configuration space.

Next, we applied standard motion planning techniques for collision avoidance in this local neighborhood (Latombe 1991). More specifically, the trajectory was modeled as the sum of attractive and repulsive force vectors F_{att} and F_{rep}, respectively. F_{att} corresponded to the velocity vector $\dot{q}$ generated from our optimization approach. We allowed local obstacles detected by the robot to impose a repulsive force vector F_{rep} onto this desired trajectory. The magnitude of F_{rep} was proportional to the robot velocities and inversely proportional to the distances from obstacles. The resultant force $F = F_{att} + F_{rep}$ represented the compromise robot trajectories as influenced by the presence of obstacles. This effectively constrained the optimization of $\rho \in \Delta$. A representative simulation trial can be found in Figure 12 (Extension 4).

While the presence of obstacles in this example constrained the robots' motion, the control law automatically adjusted their trajectories in order to compensate for these limitations and provide improved state estimates.

3.2. Experiments with the Clodbusters

The framework was implemented on our team of Clodbuster robots shown in Figure 1. These use omnidirectional vision for sensing, on-board Pentium III computers, PXC200 framegrabbers, and 802.11b wireless networking for inter-robot communications. In these experiments, a pair of pursuers tracked a third robot serving as a moving target. Two sets of trials were conducted to demonstrate operations in both cluttered and uncluttered environments.

The system architecture used by each pursuer robot is illustrated in Figure 13. It employed our *Live-Object* framework—

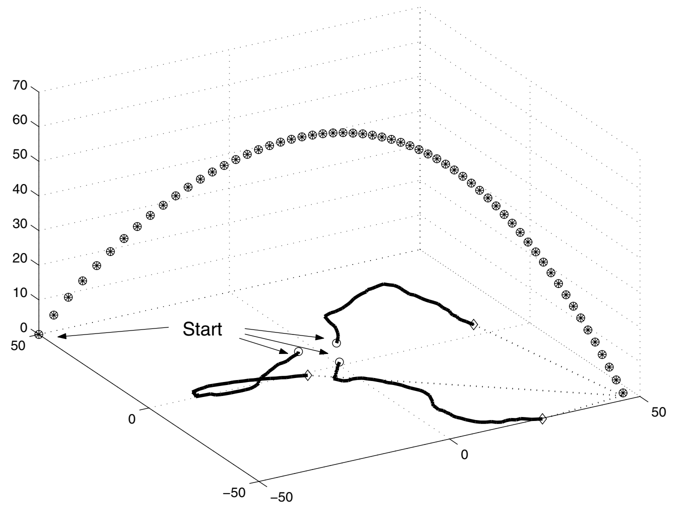

Fig. 10. Ground observer trajectories optimally tracking an aerial target.

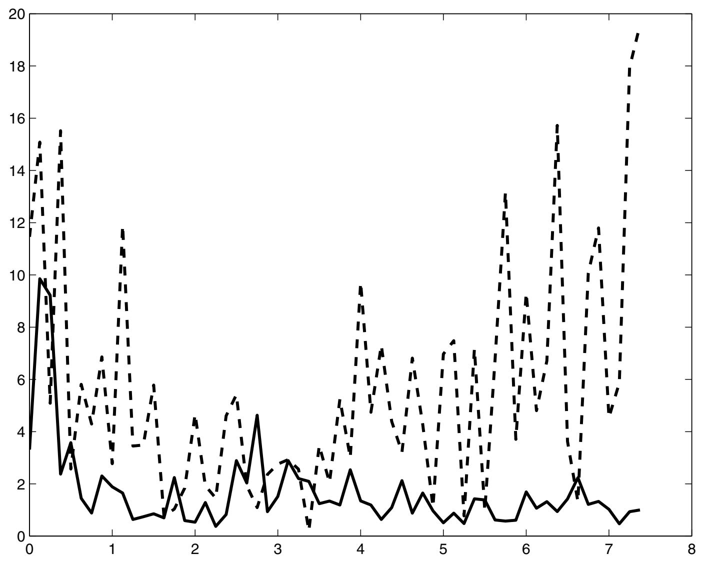

Fig. 11. Measurement errors from stationary (dashed line) and moving (solid line) robot observers. Reductions in the latter case are significant across the entire target trajectory.

a programming paradigm where objects encapsulate not only relevant algorithms and data, but also a thread within which these algorithms execute and communicate. To estimate the target position, it was first necessary for the pursuers to localize themselves. To accomplish this, the robots relied on YUV color segmentation to isolate one another in the omnidirectional camera images. By unwarping these image measurements and assuming a ground plane constraint, the pursuers were able to estimate relative range and bearing to one another, as well as the bearing to the target. Alone, these were sufficient to estimate only the relative pursuer position and target bearing. However, by communicating their respective tracking vectors, each pursuer could estimate the other's relative position *and* orientation—and subsequently the target position—from fusing the sensor measurements. The complete pursuer and target localization process ran at 15 Hz.

In the cluttered workspace trials, it was also necessary for each pursuer to estimate the position of obstacles. This was done by generating a range map to edge features in the environment as outlined in our previous work (Das et al. 2001).

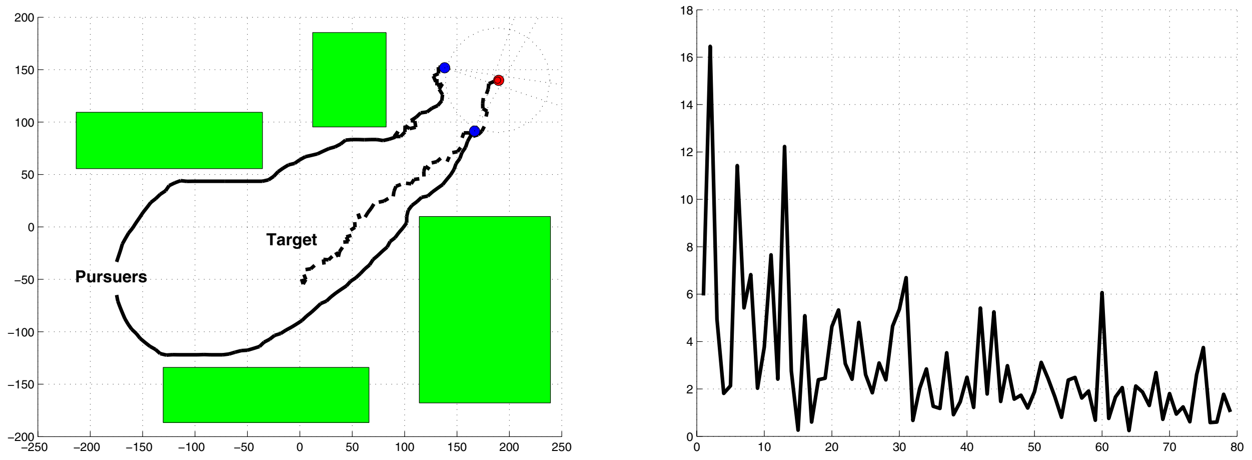

Fig. 12. Tracking a point target in a cluttered environment. Significant reductions to target position error were still realizable even in the presence of obstacles.

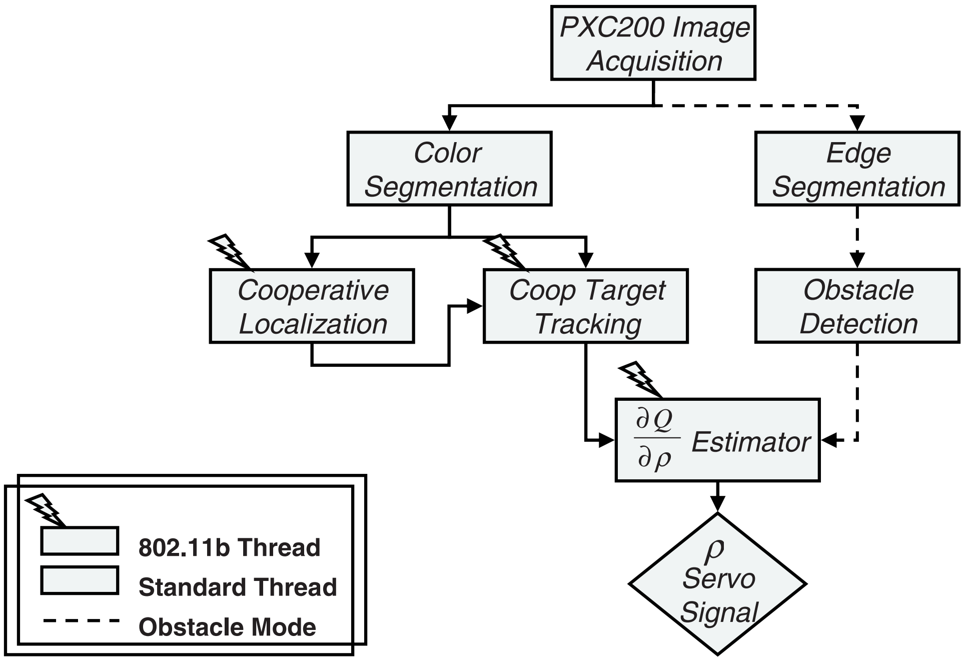

Fig. 13. Experimental architecture used by each pursuer. The 802.11b blocks correspond to threads where cooperation—and as a result communication—with other robots is required.

Target and pursuer robots were then discriminated from obstacles using their relative pose as estimated during the localization phase.

Experimental implementation followed closely with that used in the corresponding simulation experiment. Derivative estimation techniques were used to approximate the gradient of $Q(\rho)$ for optimizing the pursuers' headings. The maximum robot speed and a prescribed standoff distance served to define Δ for a given time-step. The sensor model assumed that angular measurements obtained by the robots were corrupted with additive errors drawn from a normal distribution with a standard deviation of $\sigma = 0.5^\circ$. For the cluttered workspace trials, obstacles exhibited repulsive forces when the separation was less than 1 m. Using 100 particles to approximate the probability $P(\omega)$ over the target configuration space, we were able to compute locally optimal robot configurations at a rate of 15 Hz.

A representative trial from our obstacle-free experiments is shown in Figures 14 and 15 (Extension 5). The former shows a series of images from an overhead view of the scene, while the latter shows the corresponding position error estimates. Both the trajectory and the dramatic drop in the error estimate correlate well with the corresponding simulation results presented previously in Figure 5.

Figures 16 and 17 (Extension 6) show the corresponding trial for a cluttered workspace. The effect on the motion of the right pursuer robot was significant. In contrast to the obstacle-free case, its motion was constrained to a much narrower region. However, the control scheme automatically adjusted the path of the left pursuer to compensate for this limitation. As a result, the estimated target tracking error still fell dramatically.

It should again be noted that no explicit controllers were needed for maneuvering the formation. Trajectories were implicitly generated by $Q(\rho)$, which captured the notion of a good configuration.

4. Complexity Analysis

Referring to Algorithm 1, we can see that the computational complexity will depend heavily on the number of particles used to approximate $P(\omega)$ and $P(\hat{z}|\rho, \omega)$, the *Est* function used, and the number of times $Q(\rho)$ must be computed by our *Q-optimizer* function. Thus, the complexity is implementation-dependent and can be tuned by the user based upon available computational resources and desired estimator robustness.

For our target tracking example, let m represent the number of robots, and n the number of targets. We assumed a constant number N particles for each target. Our implementation employed a least-squares *Est* function, which runs in $O(m^3)$ time. Finite difference techniques approximated $\frac{\partial Q}{\partial \rho}$, and ran in $O(m)$ time for a total complexity of $O(m^4 n)$. The framework lends itself to distributed computation. Taking advantage of this, the workload can be divided among the m robots so that the time complexity for each would scale linearly in the number of targets, and cubically in the number of robots.

Alternately, our *Est* function could employ a weighted average of the $O(m^2)$ stereo pair position estimates for each target. A straightforward weight factor is $w_{ij} = \frac{r_i r_j}{sin\theta_{ij}}$, where r_i is the estimated distance from robot i to the target, and θ_{ij} is the vergence angle. This is a first-order approximation for stereo error from the determinant of the Jacobian. Using a constant number of simultaneous perturbations to estimate $\frac{\partial Q}{\partial \rho}$, this scheme results in an $O(m^2 n)$ complexity, or $O(mn)$ when distributed.

These are only two possible implementations, but they demonstrate the flexibility of our approach. Robust estimators that scale potentially exponentially could be used for small numbers of robots and/or targets, or more expedient estimation techniques could be applied when computational resources are at issue. The choice is left to the user's discretion.

5. Conclusions and Discussion

In this paper we present an approach to the problem of controlling the configuration of a team of mobile agents so as to optimize the quality of the estimates derived from their measurements. The ideas were applied to target tracking tasks, where a team of robots was charged with optimally estimating the positions of a group of targets. This was demonstrated both in simulation and on robot platforms. The results indicate that this approach can solve fairly difficult sensor planning problems on-line without requiring excessive amounts of computational resources.

Our approach has several positive attributes. It provides an actual measurement reflecting the expected error in the estimated target positions. This is a side effect of using a numerical representation for $P(\omega)$. Also, implicit rules for switching between robot–target assignments are embedded in the optimization of $Q(\rho)$, which negates the need for explicit switching controllers.

Perhaps the most attractive feature of the approach is its flexibility. It can be used with heterogeneous sensors, as the *Est* and $Q(\rho)$ functions abstract away specific sensor characteristics. It is also scalable in the number of robots and targets, and the computational effects of this scaling can be regulated by choosing appropriate estimation and optimization techniques.

In spite of these attributes, room for improvement still exists. The robots must combine measurements to estimate the position of targets. So while distributed computation is readily realizable, some level of centralization is inherent.

Our work to this point assumed that the robot positions ρ were known a priori, and there are means to estimate the formation pose as demonstrated in Section 3.2. However, these estimates themselves are subject to uncertainty. We are

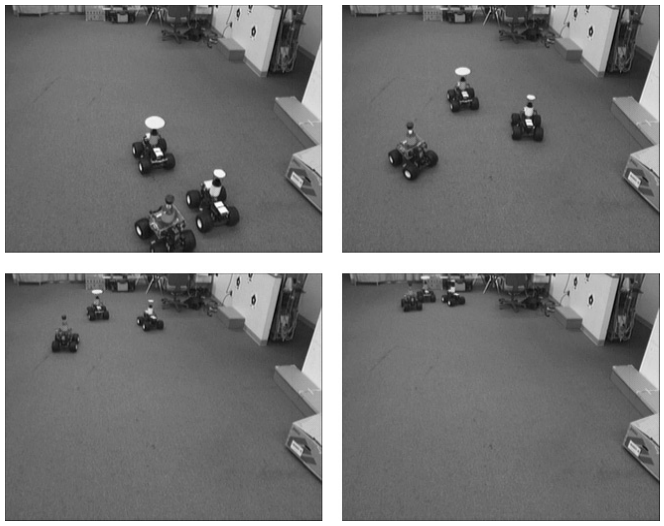
Fig. 14. Trajectory for two pursuer robots tracking a moving target robot in an obstacle-free environment.

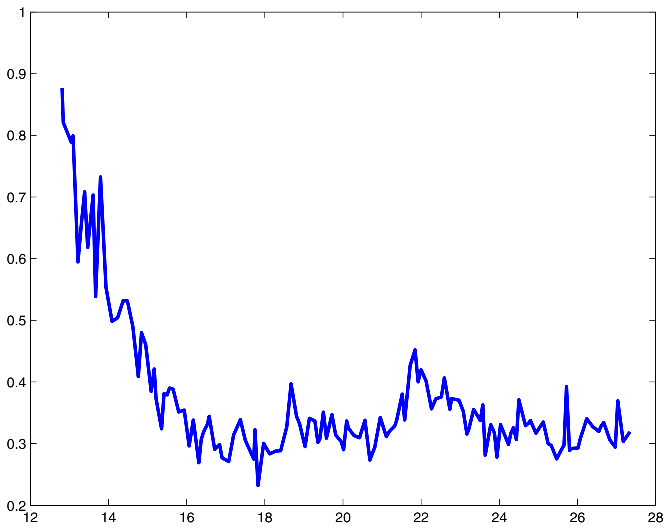

Fig. 15. Estimated RMS position error (cm) versus time for the single target case.

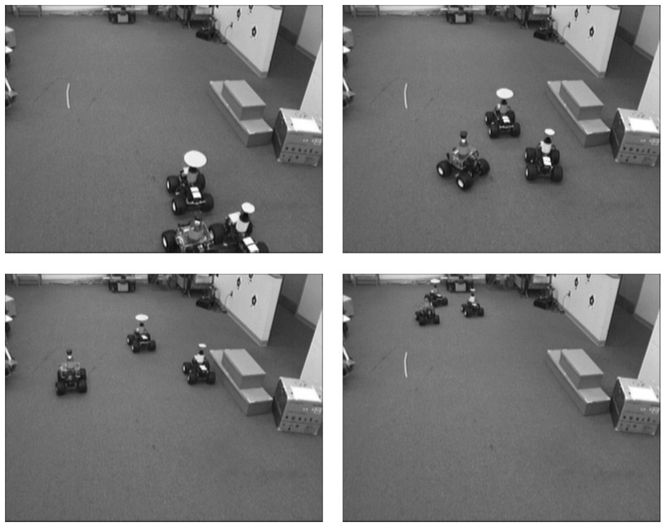
Fig. 16. Trajectory for two pursuer robots tracking a moving target robot in a cluttered workspace. The left pursuer adapts its trajectory to the right pursuer's mobility constraints.

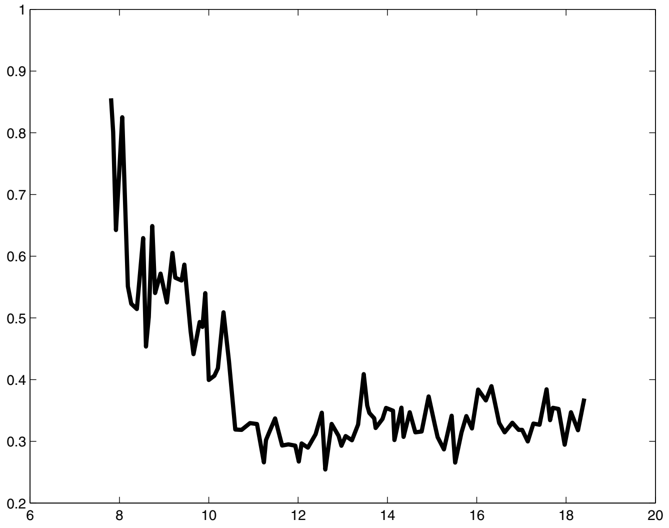

Fig. 17. Estimated RMS position error (cm) versus time for the single target case with obstacles. The results are comparable to the obstacle-free case.

currently reposing the problem within our framework as a simultaneous localization and target tracking task to address this.

The approach was applied to target tracking tasks in both open and cluttered workspaces. However, the latter was accomplished by merging with traditional motion planning techniques. As a result, it was subject to similar shortcomings (e.g., becoming trapped in local minima). Additionally, our work in cluttered environments only addressed issues relating to motion planning and not occluding obstacles. The latter topic is the subject of ongoing research.

Lastly, to this point we have assumed a sensor model with an omnidirectional field of view (FOV). Adapting our approach to limited FOV sensors involves assimilating optimal assignment techniques with trajectory generation. This is also the topic of ongoing work.

Appendix: Index to Multimedia Extensions

The multimedia extension page is found at http://www.ijrr.org.

Table of Multimedia Extensions

Extension	Type	Description
1	Video	Simulation of two robots optimally tracking an unpredictable point target
2	Video	Simulation of four robots optimally tracking three unpredictable point targets
3	Video	Simulation of three ground observers using a dynamical model to optimally track an aerial target
4	Video	Simulation of tracking a point target in a cluttered workspace
5	Video	Experimental trial with two pursuer robots tracking a third target robot in an obstacle-free workspace
6	Video	Experimental trial with two pursuer robots tracking a third target robot in a cluttered workspace

Acknowledgments

This material is based upon work supported by the National Science Foundation under a CAREER Grant (Grant No 9875867) and by DARPA under the MARS program.

References

Alur, R. et al. December 2000. A framework and architecture for multi-robot coordination. In *Proceedings of the 7th International Symposium on Experimental Robotics*, Honolulu, Hawaii.

Bajcsy, R. August 1988. Active perception. In *Proceedings of the IEEE, Special Issue on Computer Vision* 76(8):996–1005.

Das, A. et al. May 2001. Real-time vision based control of a non-holonomic robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Seoul, Korea. pp. 1714–1719.

Dissanayake, G., Newman, P., Durrant-Whyte, H., and Csorba, M. 2001. A solution to the simultaneous localization and map building. *IEEE Transactions on Robotics and Automation* 17(3):229–241.

Fabiani, P., Gonzalez-Banos, H., Latombe, J., and Lin, D. 2001. Tracking a partially predictable object with uncertainties and visibility constraints. *Journal of Autonomous Robots* 38(1):31–48.

Isard, M., and Blake, A. 1998. Condensation–conditional density propagation for visual tracking. *International Journal of Computer Vision* 29(1):5–28.

Krotkov, E., and Bajcsy, R. 1993. Active vision for reliable ranging: Cooperating focus, stereo, and vergence. *International Journal of Computer Vision* 11(2):187–203.

Latombe, J. 1991. *Robot Motion Planning*. Kluwer Academic, Dordrecht.

LaValle, S., Gonzalez-Banos, H., Becker, C., and Latombe, J. April 1997. Motion strategies for maintaining visibility of a moving target. In *Proceeding of the IEEE International Conference on Robotics and Automation*, Albuquerque, NM. pp. 731–736.

Majumder, S., Scheding, S., and Durrant-Whyte, H. 2001. Multi-sensor data fusion for underwater navigation. *Robotics and Autonomous Systems* 35(1):97–108.

Parker, L. 1999. Cooperative robotics for multi-target observation. *Intelligent Automation and Soft Computing* 5(1):5–19.

Pito, R. 1999. A solution to the next best view problem for automated surface acquisition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21(10):1016–1030.

Press, W., Teukolsky, S., Vetterling, W., and Flannery, B. 1993. *Numerical Recipes in C*. Cambridge University Press, Cambridge.

Spletzer, J. et al. October 2001. Cooperative localization and control for multi-robot manipulation. In *International Conference on Intelligent Robots and Systems*, Maui, Hawaii.

Stamos, I., and Allen, P. June 1998. Interactive sensor planning. In *Computer Vision and Pattern Recognition Conference*, Santa Barbara, CA. pp. 489–495.

Thrun, S. 2001. A probabilistic online mapping algorithm for teams of mobile robots. *International Journal of Robotics Research* 20(5):335–363.

Thrun, S., Fox, D., Burgard, W., and Dellaert, F. 2000. Robust Monte Carlo localization for mobile robots. *Artificial Intelligence* 128(1–2):99–141.