Distributed Surveillance and Reconnaissance Using Multiple Autonomous ATVs: CyberScout

Mahesh Saptharishi, C. Spence Oliver, Christopher P. Diehl, Kiran S. Bhat, John M. Dolan, Ashitey Trebi-Ollennu and Pradeep K. Khosla

***Abstract*— The objective of the CyberScout project is to develop an autonomous surveillance and reconnaissance system using a network of all-terrain vehicles. In this paper, we focus on two facets of this system: 1) vision for surveillance and 2) autonomous navigation and dynamic path planning.**

In the area of vision-based surveillance, we have developed robust, efficient algorithms to detect, classify, and track moving objects of interest (person, people, or vehicle) with a static camera. Adaptation through feedback from the classifier and tracker allow the detector to use grayscale imagery, but perform as well as prior color-based detectors. We have extended the detector using scene mosaicing to detect and index moving objects when the camera is panning or tilting. The classification algorithm performs well (less than 8% error rate for all classes) with coarse inputs (20x20-pixel binary image chips), has unparalleled rejection capabilities (rejects 72% of spurious detections), and can flag novel moving objects. The tracking algorithm achieves highly accurate (96%) frame-to-frame correspondence for multiple moving objects in cluttered scenes by determining the discriminant relevance of object features.

We have also developed a novel mission coordination architecture, CPAD (Checkpoint/Priority/Action Database), which performs path planning via checkpoint and dynamic priority assignment, using statistical estimates of the environment's motion structure. The motion structure is used to make both preplanning and reactive behaviors more efficient by applying global context. This approach is more computationally efficient than centralized approaches and exploits robot cooperation in dynamic environments better than decoupled approaches.

***Index Terms*— Multiple autonomous vehicles, dynamic path planning, visual surveillance, reconnaissance, motion detection with image mosaics, object classification, moving object correspondence.**

I. Introduction

Camera-based surveillance has long been used for security and observation purposes. Surveillance cameras are typically fixed at known positions and cover a circumscribed area defined by the fields of view of the cameras. Although some recent vision work has addressed autonomous surveillance [1][2][3][4], in most cases humans perform the sensory processing, either in real time, or by reviewing footage.

Manuscript received March 27, 2001. The research described in this paper was carried out by the Institute for Complex Engineered Systems, Carnegie Mellon University, under a contract with the Defense Advanced Research Projects Agency.

All authors are with the Institute for Complex Engineered Systems, Carnegie Mellon University. (address all correspondence to: {mahesh | jmd}@andrew.cmu.edu).

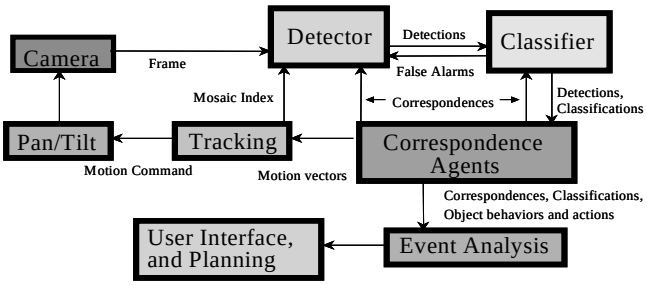

Fig. 1 Surveillance System Architecture

Likewise, humans have performed reconnaissance, or scouting, for centuries in military and other applications in order to inspect terrain and identify and classify activities in the environment. In the CyberScout project, we combine the sensory capabilities of surveillance with the mobility of reconnaissance by mounting cameras on mobile robotic platforms. The resulting groups of collaborating reconnaissance and surveillance robots pose interesting challenges in vision-based surveillance algorithms and mission planning.

This paper primarily describes the surveillance-algorithm and mission-sensitive path-planning components of CyberScout. Section II reviews previous work in autonomous surveillance with mobile robots. Section III gives a brief overview of the CyberScout system. Fig. 1 shows the architecture of the surveillance system within each CyberScout. A camera on a CyberScout captures a frame and sends it to the motion detection (detector) algorithm, which segments moving objects from the scene. A detected object is sent to the classifier, which determines the object type and sends the aggregated information to the correspondence algorithm, which temporally tracks the object. The correspondence information is then sent back to the classifier, which adjusts its classification decision based on a sequence of individually classified and segmented images of the same object. The classifier in turn gives its classifications to the detector, which uses the information in its adaptation process. Sections IV, V, and VI respectively describe the motion detection, classification and correspondence algorithms. The descriptions of the surveillance algorithms include key results and evaluations of the techniques. In order to accomplish the surveillance mission, the CyberScout has to be able to navigate to a specified geographic location and also be able to adjust its position based on feedback from the surveillance algorithms. Section VII describes a novel path-planning system (CPAD) for navigation of multiple CyberScouts in a dynamic environment, and Section VIII gives conclusions.

II. BACKGROUND

In the past decade there has been considerable effort in developing autonomous robotic vehicles for random patrols, barrier assessment, intruder detection, building or terrain mapping, explosives neutralization, and reconnaissance and surveillance. Mobile robotic platforms with the above capabilities improve the ability to counter threats, limit risks to personnel, and reduce manpower requirements in hazardous environments. The ultimate goal of the CyberScout Project at Carnegie Mellon University's Institute for Complex Engineered Systems is to develop mobile robotic technologies that extend the sphere of awareness and mobility of an individual or group performing such operations. By increasing sensory "reach" and giving the robots a significant degree of autonomy, CyberScout seeks to augment human capabilities, reduce exposure to risk, and present timely, relevant information to the user.

Similar initiatives to develop Unmanned Ground Vehicles (UGV) for remote reconnaissance and surveillance have been reported in the literature. Sandia National Laboratories developed the Surveillance And Reconnaissance Ground Equipment (SARGE) [5] and SARGE II robots (retrofitted Yamaha all-terrain vehicles). SARGE was teleoperated and was used by the US Army and Marines to develop robotic ground vehicle tactics and doctrine. Similar teleoperated battlefield mobile robot research sponsored by US DoD is reported in [6], [7] and [8]. SARGE II included multiprocessor computing, differential GPS (DGPS), and Laser Radar for autonomous navigation.

Other notable surveillance UGVs include the Mobile Detection Assessment and Response System-Exterior (MDARS-E) [9] and the Autonomous Robot for Surveillance Key Applications (ARSKA) [10]. Each is capable of some degree of autonomous navigation via DGPS and other sensors and can convey position and environmental information to a ground station. ARSKA has been used to guard an Army storage area.

The above-mentioned systems have several drawbacks. They focus on the use a single platform, and they achieve minimal supervised autonomy using costly and power-intensive sensors (e.g. optical range imaging sensors) and high-bandwidth, bulky teleoperated command-and-control ground stations. The path-planning algorithms used in these systems require often unavailable *a priori* knowledge of free paths and obstacles in the environment. Due to a lack of modularity, the incorporation of higher levels of autonomous capabilities and new sensors requires a significant redesign of the control architecture. These deficiencies pose considerable challenges when such platforms are employed in applications or environments for which they were not initially designed.

Coupled with appropriate leadership and coordination, human teams can accomplish harder and more extensive tasks than can any one individual. Similarly, autonomous robotic systems may benefit from the use of multiple platforms. This has led to increased research in the area of multi-agent or swarm systems [11][12], non-centralized collections of relatively autonomous entities interacting with each other in a dynamic environment. In single-platform surveillance systems, the principal cost is the sensor suite and payload. A distributed multi-agent approach offers several advantages. First, a larger number of sensors can be deployed over a greater area. Intelligent cooperation may allow the use of less expensive sensors. Second, mission robustness is increased, since even if some agents fail, others remain to perform the mission. Third, mission performance is more flexible, since groups of agents can perform different tasks at various locations. For example, the likelihood of classifying an object or target increases if multiple sensors (of possibly different modalities) are focused on it from different locations.

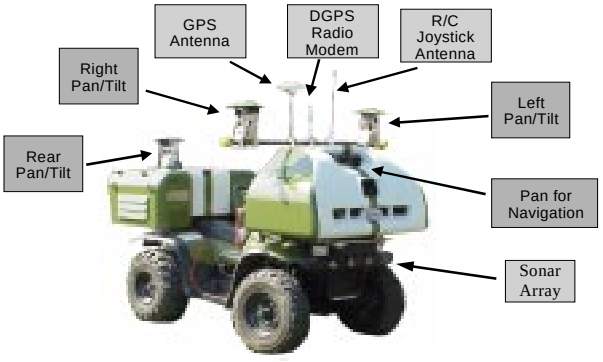

Fig. 2. Lewis, one of our two retrofitted ATVs.

III. THE CYBERSCOUT SYSTEM

In pursuing our goal of creating a mobile autonomous surveillance and reconnaissance system, we have created a group of two mobile and four stationary sentries [13][14]. The mobile sentries, named after the famous explorers Lewis and Clark, are Polaris All-Terrain Vehicles (ATVs) (Fig. 2) retrofitted with automated throttle, steering, gearing, and braking, and computation for control, navigation, perception, and communication. The computational architecture is two-tiered: locomotion (low-level processing) is performed by a PC/104, while planning, perception, and communications (high-level processing) are performed by a set of three networked PCs in a custom housing mounted on the front of the vehicle. Each ATV has multiple cameras for both navigation and surveillance. All sentries are able to communicate with one another via wireless Ethernet, and all run the same perception algorithms for performing surveillance. We have developed a distributed, agent-based software framework called CyberARIES (**A**utonomous **R**econnaissance and **I**ntelligent **E**xploration **S**ystem) [15] to accommodate cooperation among algorithms and between sentries. A user may task and observe the system using a command and control graphical user interface running on a laptop with a wireless link.

IV. MOTION DETECTION

Most surveillance systems [1][2][3][4] have found background subtraction to be an efficient means of motion detection with a stationary camera. Our basic algorithm is similar to those described in [2], [3] and [4]. The algorithm described in [2] uses an IIR filter to model the background. In

[3], Grimson et al. use a gaussian mixture to estimate the background of the scene. Both [2] and [3] use color imagery in their background modeling process. The technique described in [4] estimates the maximum and minimum intensity differences for each pixel while there are no moving objects in the scene. This information is then used to detect moving objects. We use grayscale imagery to estimate the background. Key differences between our motion detection algorithm and prior work are:

- The use of feedback from higher-level processes such as the classifier and the correspondence agents to adapt the detection process. This feedback helps us work with just grayscale imagery while performing just as well as color-based detection processes such as [2].
- The construction and use of background mosaics to extend background subtraction to also work when the camera is panning and tilting.

A. Multi-modal background subtraction

As in [2] and most other techniques that use multi-modal models of the pixel color distributions, we rely on estimating the number of modes and for each mode: the center μ, the width σ and a probability ρ (the probability of seeing that particular mode). When an intensity value seen by the pixel falls within a mode ($[\mu-\sigma,\mu+\sigma]$) and the associated probability of that mode (ρ) is greater than a preset threshold, the pixel is declared as background. When a particular intensity value does not fall within any of the modes for that pixel, a new filter is added with an associated low probability, a default width, and a center equal to the new intensity value. When a mode accounts for an intensity value, its probability is increased and the probabilities of the rest of the modes associated with that pixel are decreased. All pixels in the image declared as foreground are grouped using connected components analysis and are passed on to the classifier. At each detection cycle the parameters associated with the modes corresponding to a pixel adapt to better fit the observed values. In practice, we have found that no more than three modes are required for a robust background model. An auto-regressive filter is used to adapt the parameters μ, σ and ρ after each frame. For any parameter θ [1] at a time instance n and an observed value v we use the update rule $\theta[n+1]=\theta[n]+\lambda(\gamma)(v[n]-\theta[n])$ to adapt the parameter. The learning rate, $\lambda(\gamma)$ controlling the adaptation takes on values in the interval [0,1] and is a function of the classification confidence γ. The classifier notifies the detector of those moving objects that it rejected as foliage or spurious detections with the associated classification confidence γ. If no detection is associated with a pixel, then a default learning rate $\lambda_{Default}$ is used. On the other hand, if the pixel is declared as foreground and is part of a confidently classified object, $\lambda(\gamma)$ is set to a low value $\lambda_{Confident} \ll \lambda_{Default}$. Similarly, if the pixel is declared as foreground and is part of an object declared as spurious or rejected by the classifier then the learning rate is set to a high value $\lambda_{rejected} \gg \lambda_{Default}$. Using this adaptation strategy, the detector quickly reduces false alarms due to moving foliage and camera jitter while continuing to detect valid moving objects robustly. Also, moving objects that are stationary for an extended period of time are absorbed into the background using a longer time constant. The classifier's rejection mechanism is described in section V.

[1] We use θ for notational convenience as a variable to designate any parameter μ, σ or ρ.

B. Background subtraction using mosaics

The above-described algorithm continuously updates the background for a fixed pan and tilt position of the camera. We propose the use of image mosaics as an effective way to perform motion detection as the camera pans or tilts in saccadic movements. Key differences between our algorithm for motion detection [16] while the camera is panning and tilting and prior work are:

- Significantly increased speed and elimination of user intervention in constructing a background image mosaic
- A fast and robust way to index the background image mosaic given a particular pan and tilt position in the presence of large moving objects in the scene

Several techniques have been proposed to create an image mosaic from sequences of images [17][18][19]. They obtain the registration between images by minimizing the sum-squared error of image intensities at each pixel. Although these techniques produce very accurate registration results, they tend to be slow, and typically require user interaction to initialize the registration. We create an image mosaic in near real time by locating and tracking feature points in the image sequence. This technique is much faster than the techniques developed previously, and does not require any user intervention.

Our algorithm uses a feature tracker to robustly identify and track feature points through the background image sequence [20]. Consider a pixel $\vec{x}=(x,y)$ in an image I that translates to $\vec{x}'=(x+t_x,y+t_y)$ in the subsequent image J. We estimate the translation $\vec{t}=(t_x,t_y)$ by minimizing the intensity error between I and J where W is a 7×7 feature window containing $\vec{x}$ as shown in equation (1):

$$E(\vec{t})=\sum_W\left(J(\vec{x}+\vec{t})-I(\vec{x})\right)^2 \quad (1)$$

After a first-order Taylor series expansion of equation (1), we obtain equation (2), where $\vec{e}=J(\vec{x})-I(\vec{x})$ and $\vec{g}^T=\nabla_{\vec{x}}J(\vec{x})$:

$$E(\vec{t})=\sum_W\left(\vec{e}+\vec{g}^T\vec{t}\right)^2 \quad (2)$$

The solution to equation (2) takes the form of a least-squares problem as shown in equation (3):

Fig 3. The top image shows a panoramic background image mosaic constructed using a set of background images. The next row shows two images where the right image is a frame captured for the current pan and tilt position and the left image is the corresponding background image indexed from the mosaic. The bottom image shows the segmented foreground object.

$$\underbrace{\left(\sum_{W} \bar{g}\bar{g}^{T}\right)}_{Z}\vec{t} = -\left(\sum_{W} \bar{e}\bar{g}\right) \tag{3}$$

The eigenvalues of the 2×2 matrix Z give a good measure of the texture in an image window. In particular, we choose a feature point if the minimum eigenvalue is greater than a set threshold. We select and track N feature points ($N \in [80,100]$) through the image sequence. We use the coordinates of the tracked feature points to fit an affine transformation model between the two consecutive images in the sequence. The affine transformation allows for the representation of the motion of a planar surface under orthographic projection. A more accurate motion model is the perspective transformation, which is characterized by 8 parameters. However, for our application, we have found that an affine model is sufficiently accurate. Moreover, it is extremely simple to implement and is very fast.

Once the affine parameters have been calculated, we can warp all the images with respect to a common coordinate system. We have arbitrarily chosen the first image as the reference, and warped all other images into the first image's coordinate system. We use a triangular weighting function (with maximum weight at the image center and zero weight at the edges) to blend overlapping regions between different warped images.

Our system initially builds a background mosaic of the entire viewable environment. During surveillance, as the camera pans and tilts, we index and update the corresponding viewable subset of the background mosaic to perform motion detection and segmentation. Currently, no algorithms discussed in the literature solve the indexing problem in the presence of moving objects. We propose a simple real-time method to handle the indexing of the mosaic. During the mosaic construction, we store the pan and tilt positions and affine warp parameters for all the images in the sequence. For given pan and tilt positions we can coarsely index the mosaic using the stored affine parameters. However, due to hysteresis and other errors in the pan-tilt unit, the indexed image is offset relative to the actual camera image by approximately 2-3 pixels. We solve this problem by performing registration between the indexed mosaic image and the actual camera image. We mask moving objects in both images by using the information provided by the detector and subtracting the images while ignoring regions with large errors. Then we solve for the translation between the two masked images. Once we find the translation parameter, we can index the correct portion of the mosaic and can perform accurate motion segmentation. In Fig 3 we show a panoramic background constructed by the algorithm, a raw image, the corresponding indexed background from the mosaic and the segmented moving object.

V. Classification

The detection algorithm provides the classifier with a set of segmented objects from the scene. The objective of the classifier is to label each object as "people," "person," or "car." The correspondence algorithm also provides the classifier with a sequence of images corresponding to a particular object. The classifier must assign one label to this sequence of individually labeled images. In this section, we present a strategy for designing a classifier that recognizes known objects accurately while rejecting unknown and spurious detections. The key contributions of our strategy are:

- The classifier performs extremely well with (low-information) 20×20 binary silhouettes of objects as inputs and a logistic linear hypothesis space. The final classifier had an error rate of 7.3% on class 'person,' 7.9% on class 'people' and 2% on class car.
- The classifier provides unparalleled rejection performance. The classifier successfully rejects 72% of the foliage and spurious detections.

A. Object Classifier Structure

CyberScout's real-time constraint dictates computational simplicity of the classifier, whereas novelty detection and rejection capability require a high-dimensional feature space to aid in discriminating between known and unknown object classes. For real-time object classification, the cost of exploiting spatial *and* temporal variation in the appearance of an object as a discriminator does not justify the potential performance benefits. We limit our focus to spatial features and show that both classification accuracy and rejection performance are extremely high.

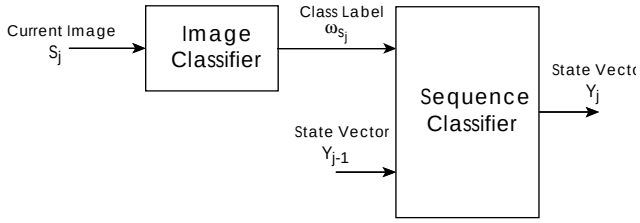

Fig. 4. The structure of the object classification algorithm

Since the classification process examines only the spatial features of each image, the classification of an image sequence becomes a two-step process, as shown in Fig. 4. In the image classification phase, the current image S_j from the image sequence *S* is analyzed to determine the most likely class label ω_{S_j}. In the sequence classification phase, the evidence from the classification of the current image is integrated with past evidence to produce an overall decision with a confidence level. Given our objective, it may seem natural to derive estimates of the *a posteriori* class probabilities $\hat{P}(\omega_k | S_j)$ and threshold the probability of correct classification (*maximum a posteriori* class probability) to obtain an approximation of the Bayes-optimal classification and rejection strategies. However, even assuming one can estimate the posterior probabilities for a given feature vector S_j, the estimate of the probability of correct classification gives misleading results when the training data are not consistent with the underlying distribution

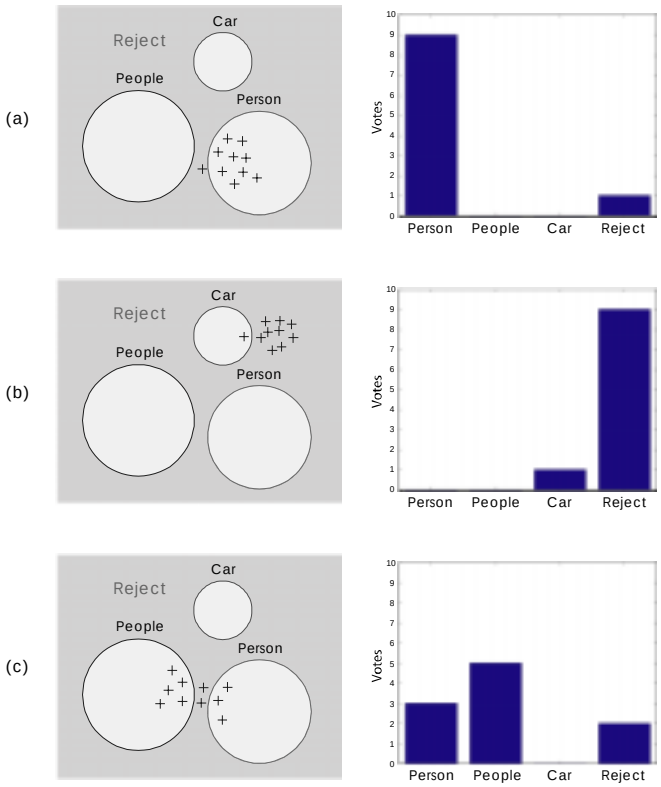

Fig. 5. Examples of image sequence class label distributions: (a) confident classification: vast majority of the images lie in one decision region (b) consistent rejection: significant fraction of the images lie in the rejection region (c) classifier confusion: images distributed over two or more regions.

[21]. Another approach to defining the partition involves estimating indicator functions that define the regions [22]. Our approach directly estimates closed decision regions with minimal volume that encompass a specified fraction of the training examples.

As the image classifier processes a series of images of a given object (facilitated by the correspondence algorithm), a sequence of class labels is produced. Based on this sequence, we wish to assign a class label to the image sequence along with a level of classification confidence.

If the combination of image representation and partition provides the necessary discrimination power, the class label distributions induced by known objects, unknown objects and novel views of known objects will be sufficiently separable in class label distribution space. Ideally, the image classifier would reliably and consistently classify or reject the images in a given sequence, as illustrated in Fig. 5 (a) and (b), thereby simplifying the image sequence classification task. In reality, image sequences often induce a mixture of classifier outputs, as illustrated in Fig. 5 (c), indicating classifier confusion. When classifier confusion occurs, our ability to discriminate between known and unknown objects does not necessarily decrease significantly. Our approach to sequence classification entails mapping class label distributions to one of the known object classes. We learn a partition of the class label distribution space from the training data, and classification confidence is assessed in a principled manner.

B. Experimental Results

Since the motion detector generally provides an accurate segmentation of the moving objects, we chose to classify size-normalized binary images of the moving objects. Size normalization is achieved by first resizing each binary image so that the largest dimension, *N*, is fixed. The resized image is then zero-padded to produce a square $N \times N$ pixel image with the original image in the center. We then learned a *large margin partition* of the image feature space by training a set of discriminant functions using *differential learning* [23]. Differential learning is a generalization of a standard formulation of *support vector learning* that is ideally suited for learning partitions of high-dimensional spaces from sparse data sets [23]. Once the initial partition is learned, we define the rejection region by estimating class-conditional margin thresholds that yield a given class-conditional probability of detection on a validation set.

For this classification task, we selected the logistic linear form, which induces decision boundaries that form semi-infinite wedges in feature space. Since the set of $N \times N$ binary images defines the vertices of an N^2-dimensional hypercube, the logistic linear form induces closed decision regions on the surface of the hypercube. Increasing the threshold decreases the volume of the closed decision region. We therefore control the size of the decision regions indirectly by maximizing the separability of the data.

Two different normalization techniques were investigated. The first involved centering the images horizontally, based on the center of mass. The second involved horizontally translating the images to maximize the resulting differential

during training and testing. The classifiers trained on the unnormalized and centered images failed to reject a majority of a set of false alarm images generated by moving foliage. To overcome this, we investigated the effectiveness of horizontal image translation such that the difference between the largest and next largest discriminant function output is maximized. This strategy simultaneously learns a transformation of the image space and a partition of the resulting space that are complementary. The classifier normalizing 20×20 *pixel images* in this manner yielded a 30% median improvement in rejection performance compared to the 30×30 pixel image classifier processing centered images. The improved rejection capability also dramatically improved the performance of the detector when combined with a classification confidence-guided learning rate selection for the adaptation of the detector's parameters.

Once the image classifier is trained, we process the training image sequences to derive the associated class labels for training the sequence classifier. Each training image sequence has a corresponding class label distribution that is simply the relative frequencies of each class label in the class label sequence. Using these class label distributions, we trained a logistic linear classifier to partition the class label distribution space.

As the sequence classifier processed the observed image sequences, the image sequences assigned to a given class ω_k were rank-ordered based on the likelihood $\hat{\mathrm{p}}(\delta|\omega_k)$ where δ is the margin produced by the sequence classifier [23]. Because the likelihood is generally monotonically increasing with increasing margin, we sorted the image sequences based on the margin. This allowed the user to quickly focus on the examples, such as those in Fig. 6, that cause the greatest degree of confusion for the current classifier.

To evaluate the utility of the 20×20 pixel image sequence classifier, we classified and sorted a test set of image sequences consisting of examples from the three known classes, along with image sequences of bicycles, trucks and vans. Our main objective was to determine whether this process would allow the user to easily detect many of the examples from the unknown classes by simply scanning through a small subset of the sorted observations. In the case of the bicycle class, 70% of the observed examples were in the top 20% of the data assigned to the people class. Bicycles typically caused unique patterns of classifier confusion, which simplified their identification. If the classifier was able to view the bicycle from multiple perspectives, it was more likely to produce a mixture of rejections and class labels. Trucks and vans, on the other hand, could not be successfully discriminated from cars. Only vehicles like FedEx trucks produced classifier confusion because their appearance varies significantly from cars. At such low resolution, it is nearly impossible to reliably distinguish between cars, trucks and vans. The final classifier had an error rate of 7.3% on class 'person,' 7.9% on class 'people' and 2% on class car.

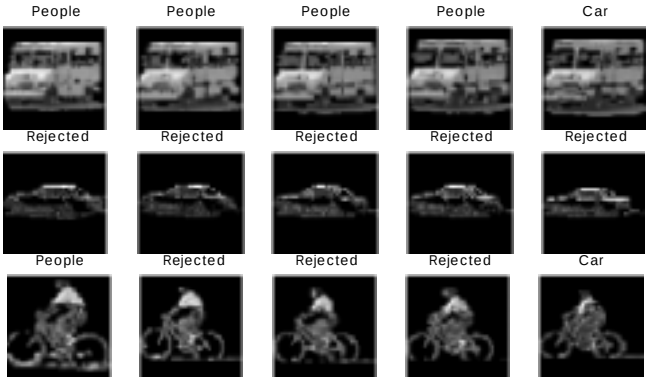

Fig. 6. Unknown image sequences identified by the image sequence classifier

VI. Correspondence

Once objects are classified, they are temporally corresponded within a sensor's field of view and between sensors. The correspondence information is crucial not only for the rejection capability described in the previous section, but also to interpret the activity in the scene. The complexity of motions in the environment precludes the use of simple positional correspondence, i.e., correspondence based purely on the positions of moving objects. Positional correspondence also fails when moving objects are relatively large with respect to the field of view of the sensors. Also, in a multiple sensor surveillance framework, we need the capability to hand off targets between sensors. In such situations, other features of the moving objects, such as different appearance traits, need to be put to good use for robust correspondence. How can we select appearance features so as to facilitate good correspondence? The measure of goodness of the features we choose not only depends on the object in question, but also on other objects in the scene. A globally good set of features can be estimated *a priori*, but only a subset of these features might be relevant to the correspondence of a particular object. We pose the estimation of the relevance of globally good features for corresponding a particular object as an on-line learning task. Differential discriminative diagnosis provides a systematic method for estimating the relevance of features and checking the temporal consistency of these features for a particular object.

A. Related approaches

Much work has been devoted to efficient object correspondence and tracking. Surveillance systems described in [2], [3], [4] and [25] deal with the problem of detecting and tracking moving objects. The system described in [2] uses correlation with dynamic templates of the object. The system described in [3] uses linear prediction with Kalman filters of the position and size of the moving objects. The algorithm described in [25] combines the detection and tracking process. The technique in [4] uses correlation of the moving object's silhouette and template matching. A people-tracking technique described in [26] uses gaussian models to represent 2-D regions or blobs. Wren and Pentland in [27] extend the algorithm in [26] to a 3D context. McKenna et al. [28] use a gaussian mixture model of the color of an object to track it. Black and Jepson in [27] describe an eigenspace method for tracking specific rigid objects. Rehg et al. [30] describe a

method for tracking high-DOF articulated objects using kinematic modelling. Other notable people-tracking systems include KidsRoom [31] and Cardboard People [32].

Our proposed method relies on knowing the class of the object (person, people or vehicle). Thus, there is domain knowledge incorporated in the correspondence process. The injected domain knowledge not only helps in making the correspondence process robust, but it also helps in making it computationally efficient. By accounting for the different moving objects of interest, we come close to obtaining the versatility of class-independent correspondence. The key differences between our algorithm and prior work are that:

- In contrast to most of the methods mentioned in this section, our proposed technique poses moving object correspondence as a statistical pattern classification/discrimination problem. Thus, we rely on the appearance traits of an object that is independent of particular articulations and position of the object.
- Rather than modeling motion, our algorithm finds stable discriminating features to correspond an object. We show that training an agent to correspond an object off-line and giving it the capability to customize itself to the object on-line, leads to an efficient correspondence algorithm. A detailed description of this algorithm is provided in [1].

B. Discriminative Diagnosis

We begin by training a classifier whose objective is to decide whether or not two instances of an object at two consecutive time instants are the same or different. Thus, the classifier classifies each pair of objects as a match or no match. The input to the classifier is the absolute value of the intensity difference between the pair of objects. For this particular application we chose a single-output logistic linear neural network trained with differential learning [23] as the classifier. A total of 249 sequences were available for training the classifier. A total of 120 sequences were used for independent testing. Each sequence contained an average of 15 instances of an object. Sequences were manually sorted from data collections in different environments and sensors. Different permutations of sequence pairs were constructed for training and testing. The classifier successfully matched instances of the same moving object with an accuracy of 87%. The 95% confidence interval is [84%, 90%].

The classifier's training process selects features on the person's body that help in the classification task given the training data. These features are globally relevant, i.e., the selected features help in discriminating a majority of the moving objects without being specific to a particular object. Different environmental conditions and different scenes may reduce or increase the relevance of certain features. More importantly, only a subset of the globally relevant features may be applicable to the correspondence of a moving object. In some cases, certain globally relevant features may actually hurt the correspondence process. Thus, identifying the feature subset that is relevant to the correspondence of a particular moving object could increase performance dramatically. An agent that represents a moving object customizes itself by estimating the relevance of each feature in the input vector based on the reaction of the classifier to the input and the other objects in the scene. This estimation process is accomplished by means of differential discriminative diagnosis.

Differential Discriminative Diagnosis uses a Taylor series expansion of the classifier $C\left(X_{n,i,j}\right)$ to learn a distance metric between successive instances of the same object. The input $X_{n,i,j}$ is the input at time n corresponding to the unique objects i and j in the scene. If i and j are equal then $X_{n,i,j}$ represents an input vector that should be classified as a match. In addition, we define the expression $h_{n,i,j} \equiv X_{n,i,j} - X_{n-1,i,i}$ as the difference between two distinct input vectors. Given a classifier input $X_{n-1,i,i}$ that we have already classified as a match, we can predict the change in its output for a different input $X_{n,i,j}$ as shown in equation (4):

$$P_{n,i,j} \equiv \tilde{C}\left(X_{n-1,i,i} + R_i h_{n,i,j}\right) - C\left(X_{n-1,i,i}\right) \tag{4}$$

The term $\tilde{C}\left(X_{n-1,i,i} + R_i h_{n,i,j}\right)$ is the Taylor series expansion of the classifier around the point $X_{n-1,i,i}$. The diagonal matrix R_i scales each dimension (or feature) of the difference vector between the two inputs. If R_i was just the identity matrix, then $\tilde{C}\left(X_{n-1,i,i} + R_i h_{n,i,j}\right)$ would be a standard Taylor series expansion. We can now define the notion of a *relevance differential* $\Re_{n,i,j}$ as shown in equation (5):

$$\Re_{n,i,j} \equiv \left|P_{n,i,j}\right| - \left|P_{n,i,i}\right| \tag{5}$$

The expectation here is that, for two instances of the same object, $\Re_{n,i,j}$ will be low, whereas for two instances of different objects it will be high. This can be attributed to the fact that if two points are relatively close in space, then the Taylor series expansion of a logistic linear classifier would be reasonably close for instances of the same object. But, because of articulation of a non-rigid object and self-occlusions, certain features may actually be misleading. Thus, we want to learn a relevance matrix R_i, which would amplify features that are discriminative while suppressing features that are misleading. We accomplish this by optimizing R_i such that the cumulative relevance differential (shown in equation (6)) is maximized.

$$\sum_n \sum_j \Re_{n,i,j} \tag{6}$$

The variable n in the above equation iterates over all time that the object was present in the scene while j iterates over all the other moving objects in the scene at a particular time instance. This maximization can be done efficiently online [1]. At each time instant we associate instances of an object in the scene that have the minimum relevance differential $\Re_{n,i,j}$

using the relevance matrix R_i that is unique to each object in the scene.

C. Experimental results

The performance of the new correspondence scheme was tested on the 120 independent test sequences used to evaluate the classifier-only correspondence scheme described previously. The algorithm achieved an accuracy of 96%. The 95% confidence interval is [94.3%, 97.7%]. The customization step shows statistically significant improvements over the 87% accuracy obtained using just the classifier.

Fig. 7. shows a set of two people in the scene. The correspondence algorithm needs to robustly distinguish between these two people. The image on the bottom left in Fig. 7. shows the features that the classifier was originally keying on. The image on the bottom right in Fig. 7. shows the subset of the original features that discriminative diagnosis found most useful in discriminating between these two people. Interestingly, the algorithm instantly keys on two salient differences between the two people: the tilt in the head and the raised arm of one of the people.

VII. Mission-Sensitive Path Planning

Any surveillance or reconnaissance scenario requires the CyberATVs to plan a path to a goal and navigate along this path while avoiding obstacles. The vehicles traverse unknown or partially known terrains, so pre-planned paths cannot account for all the dynamics in the environment. Reactive behaviors are used to account for this uncertainty, but they are at best locally optimal and can be inefficient from a global planning and execution perspective. However, the non-determinism of the environment and the computational inefficiency of repetitive global re-planning make reactive behaviors unavoidable. *How can the use of reactive behaviors be made more efficient?* If the structure of the dynamic environment can be predicted, then points of uncertainty along the planned path can be identified and a global context can be applied in choosing a reaction to the uncertainty. This section describes a novel approach to this problem that combines the use of checkpoints (i.e., areas where motion conflict is likely) and dynamic priorities with statistical estimation of the structure of the dynamic environment.

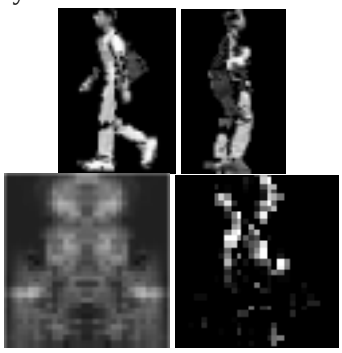

Fig. 7. The top row of images shows two distinct objects in the scene. The object on the top left is the one being tracked. The bottom row shows weights of the classifier on the left and the relevance matrix for tracked object.

A. Related approaches

Approaches to the multiple-robot path-planning problem are typically classed as centralized or decoupled. Centralized approaches regard the group of robots as a single entity such that all paths are planned simultaneously in a composite configuration space. Decoupled approaches plan for each robot individually, and then seek to resolve any resulting conflicts. Neither approach explicitly includes the notion of a dynamic environment, so changes in the environment require replanning. Given the need for replanning, centralized approaches require too extensive inter-robot communications and computation to be feasible for real-time response.

There are various decoupled methods [33][34][35] for dealing with conflicting robot paths. The method in [33] uses velocity-tuning of the robots to ensure collision avoidance in a known environment, but this method assumes a static environment for which exact velocity trajectories can be pre-calculated. In [34] traffic rules are proposed for use by each robot, but these are effective only when adhered to by all moving objects in the environment. In [35] a framework is presented for merging multiple path plans in an incremental fashion. Using this methodology, however, a given robot may have to wait an unpredictable amount of time on a higher-prioritized robot before proceeding along its path, slowing down mission completion. The proposed simple CPAD mission architecture described in the following subsection avoids these problems.

B. CPAD: Checkpoint/Priority/Action Database

The typical reconnaissance and surveillance scenario requires sensing agents to distribute themselves across a dynamic workspace while gathering information about interesting activities. We propose the Checkpoint/Priority/Action Database (CPAD) as a general framework for specifying such missions. The CPAD contains a collection of actions that can be performed by the CyberATVs to accomplish the surveillance mission. After a global surveillance task has been decomposed and distributed among the scouts, each scout plans its mission path independently. An autonomous software agent, the CPAD Builder, analyzes the combination of plans and may recompute individual plans in such a way as to minimize conflict and maximize the probability of global mission success. From this global plan, individual mission plans are constructed from available action routines in the CPAD and distributed to the respective sensing agents. The CPAD routines coordinate robot motions at possible intersections by assigning or updating relative

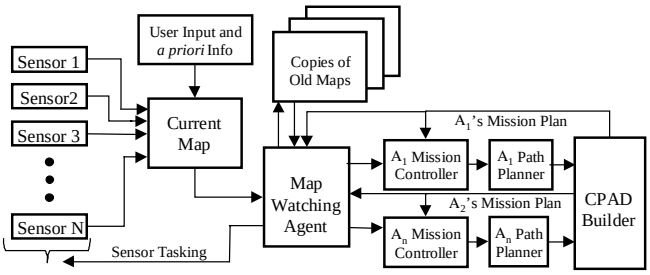

Fig. 8. Mission Planning and Execution Architecture using CPAD.

priorities as new information is obtained..

Fig. 8 shows the architecture used to synthesize the distributed sensor data for predicting possible robot collisions. The CyberScout system begins with little or no *a priori* information about the workspace. The human user selects an area or specific points to be surveyed by the sensing agents. Task assignments are then centrally decomposed from the global specification and distributed to the individual ATVs, $A_1, A_2, \ldots, A_n$, for planning. Each ATV A_i attempts to plan a path τ_i through the workspace based on the limited knowledge available at the time.

An individually planned path τ_i may contain intersections with another such path τ_j, where j represents either a robot or a detected moving obstacle. Let $I(i, j, k)$ denote the k^{th} possible intersection between two paths τ_i and τ_j. CPAD coordinates multiple paths by first assigning a "soft" relative priority $p_{i,j}$ to each pair of robots $A_{i,j}$. At each possible intersection, checkpoints are placed around the intersecting region. If A_i has a higher priority than A_j, then it has the "right of way" through the region and proceeds without stopping. If the lower-priority A_j approaches the checkpoint, it must stop and ensure that A_i is at a safe distance from the region before proceeding. Once A_j enters the intersecting region, the relative priorities between the two robots are reversed. Thus, if A_j encounters unforeseen temporary deadlock within the region, an approaching A_i will not constrain the space while A_j attempts to vacate the area. CPAD handles moving obstacle coordination in a similar way, except that an obstacle's "virtual path" is assigned the highest priority and cannot be shifted lower.

The previous example involves "soft", or adjustable prioritization. CPAD is also able to handle hard prioritization. Consider the two overlapping paths τ_1 and τ_2 shown in Fig 9.

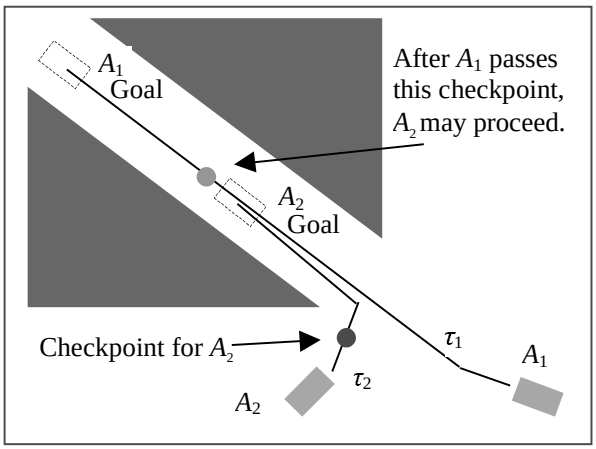

Fig 9. Example of prioritized path coordination to resolve conflicting path plans.

In order for both scouts to pass through the narrow corridor and achieve the proper goal allocation, A_2 must wait at a "hard-prioritized" checkpoint until A_1 is safely inside the corridor. Hard prioritization is also useful for specifying automated convoying capabilities [13], which allows a scout with faulty or absent localization sensors to exploit the GPS (localization) information of a lead vehicle.

The CPAD builder in Fig. 8 analyzes the collection of plans $T = \{\tau_1, \tau_2, \ldots, \tau_n\}$, identifies all possible intersections $I_i = \{I(i,1,1), \ldots, I(i,n,k)\}$ along a path τ_i, inserts checkpoints appropriately around each intersection, and assigns relative priorities $p_{i,j}$. Individual mission plans are provided to the respective scouts, and the conglomerate plan is available to the autonomous map-watching agent described in the next subsection. The CPAD builder itself can be resident on any CyberATV or central command station via CyberARIES (see section III). The map-watching agent identifies possible moving-obstacle trajectories that may intersect a path τ_i and reports them to A_i's mission controller. The mission controller then either adds the proper checkpoints to improve reactive behaviors near the obstacle or decides that a new path must be planned.

C. Identifying motion structure

Although relatively little work [36][37][38][39] has been done in this area, Kruse, Gutsche and Wahl [36][37] have recently considered analytical characterizations of environmental motion dynamics in mobile-robot path planning. They suggest an effective statistical method for estimating motion patterns and collision probabilities. Our approach is conceptually similar to theirs, but differs considerably in method and operating characteristics. First, we do not have constant coverage of the full environment, so that statistical assessments about the structure of the environment have to be made with very limited data.

Second, Kruse et al. [37] adhere to the principle that "the robot is adapted to its environment, the environment is minimally disturbed by the robot." The size, appearance and sounds of the CyberATV inherently "disturb" the environment. Not only does the robot adapt to the environment, but the environment also adapts to the robot. As an illustrative example, consider walking through a crowded field. Not only do we take steps to efficiently reach our destination, but the crowd also accommodates our actions to the best of its ability.

The motion structure we want to estimate is the set of paths that moving obstacles take in the environment. In other words, we wish to "beat a virtual path" in the map constructed by the scout. A virtual path indicates regions where the scout may possibly encounter moving obstacles. Virtual paths can be taken into account both during the planning phase and while the plan is executing. While the former use of the virtual paths attempts to eliminate uncertainty in the scout's path, the latter attempts to apply a global context in formulating a reaction to local uncertainty. Virtual paths can also be used to determine locations for surveillance.

Each scout faces computational and real-time constraints in all its actions. Given these constraints, all CyberScout agents attempt to process information that has a low spatial and temporal resolution with maximum efficiency. With computational parsimony in mind, the task of estimating and predicting motion structure in the environment uses residual data from the mapping algorithm to make its assessments. The mapping algorithm uses a Bayesian network [40] to identify static obstacles and constructs an occupancy grid of the observable environment. It also identifies, but disregards, moving obstacles. The motion structure recognition algorithm exploits this moving obstacle information.

Each identified moving obstacle does not contain enough discriminating information for accurate temporal correspondence. Thus, estimation of potential virtual paths in the map depends purely on a temporally ordered sequence of moving obstacles. Each moving obstacle is represented by its location and size. The suggested approach shows that even with relatively little information about the moving obstacles in the environment, sufficient detail can be inferred to improve the quality of the planned path. We first estimate a highly probable topography of virtual paths given the observed samples of moving obstacles. After each estimation step, we predict points of uncertainty along the scout's planned path. Virtual paths are learned using unsupervised clustering of the positions of the moving obstacles using a set of Cauchy kernel functions.

The clustering algorithm takes two sets of object seen at two consecutive time instances and matches the objects that are closest to each other. The algorithm then places a kernel over each pair of associated points with eigenvectors of the kernel's covariance matrix corresponding to the direction of movement. If two kernels have an overlap greater than a specified amount, then they are merged such that the receptive field of the merged kernel encompasses the union of the receptive fields of each of the original kernels.

After every clustering step, given a candidate path for the scout, we can determine if checkpoints are needed as a result of potential uncertainty. We use a linear neural network to decide whether or not a checkpoint is needed. Given the values of the mixture of Cauchy kernels along a stretch of the scout's path, the network is a two-class classifier that classifies the input as "checkpoint" or "no checkpoint." The linear network can simply be considered as a weighted sum of uncertainty along the scout's path. If this uncertainty exceeds the bias of the network, then a checkpoint is placed.

D. Experimental results

Our approach has been applied to various simulation data sets representing moving obstacles. Fig. 10 presents the tasking of two scouts to locations suitable for observing an area. The scenario involves detecting virtual paths and coordinating the scouts' motions accordingly. A_2 has a higher priority relative to A_1, but since A_1 enters the region surrounding the intersection $I(1,2,1)$ first, it proceeds to its goal location. Once in position, A_1 is able to collect enough data to inform A_2 that a potential collision may occur along the path τ_2. A_2's mission controller recognizes the need to re-plan and produces the new path τ_2'. Thus A_2 does not fall victim to local obstacle avoidance maneuvers in the corridor that will fail in the presence of the approaching obstacle.

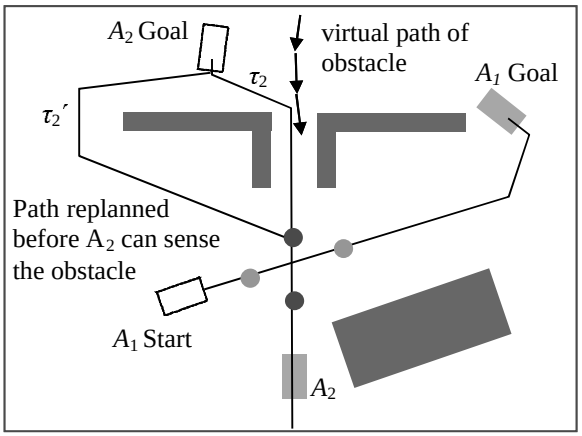

Fig. 10. A simulation with multiple ATVs and a moving obstacle. The system instantiates checkpoints appropriately during the planning stage and identifies the need to replan after collecting statistical information about the environment.

VIII. Conclusion

We have developed an autonomous surveillance and reconnaissance system called CyberScout using a network of all-terrain vehicles and stationary sensors connected by a distributed, agent-based software framework. In this paper, we have described significant contributions resulting from this work in two areas: robust, efficient vision surveillance algorithms (detection, classification, and tracking) and dynamic path planning. All surveillance algorithms execute in real time, processing multiple frames per second. The detection algorithm is able to perform as well on greyscale imagery as prior color-based detectors by using feedback from the classifier and tracker. The mosaicing algorithm extends the state of the art by creating panoramic images in near real-time and rapidly indexing moving objects. The classification algorithm performs well (less than 8% error rate for all classes) with coarse inputs (20x20-pixel binary image chips), has unparalleled rejection capabilities (rejects 72% of spurious detections), and can flag novel moving objects. The tracking algorithm achieves highly accurate (96%) frame-to-frame correspondence for multiple moving objects in cluttered scenes by determining the discriminant relevance of object features. The CPAD mission coordination architecture uses checkpoint and dynamic priority assignment and statistical motion structure to overcome some of the computational and efficiency disadvantages of fully centralized or decoupled approaches.

Acknowledgments

This work was supported in part by DARPA contract F04701-97-C-0022, "An Autonomous, Distributed Tactical Surveillance System".

REFERENCES

[1] M. Saptharishi, J. Hampshire, and P. Khosla. "Agent-Based moving object correspondence using differential discriminative diagnosis," *Proceedings of CVPR*, pp. 652-8, vol. 2, June 2000.

[2] A.J. Lipton, H. Fujiyoshi and R.S. Patil, "Moving target classification and tracking from real time video," *Proceedings of the IEEE Workshop on Applications of Computer Vision*, pp. 8-14, 1998.

[3] W.E.L. Grimson, L. Lee, R. Romano, and C. Stauffer, "Using adaptive tracking to classify and monitor activities in a site," *Proceedings of CVPR*, pp. 22-31, 1998.

[4] I. Haritaoglu, D. Harwood and L. Davis, "W4: Who? When? Where? What? A real time system for detecting and tracking people," *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 222-227, 1998.

[5] J. Bryan Pletta and J. Sackos, "An Advanced Unmanned Vehicle for Remote Applications", Sandia National Laboratories Report, 1998.

[6] C. D. Metz, H. R. Everett and S. Myers, "Recent Developments in Tactical Unmanned Ground Vehicles", *Association for Unmanned Vehicle Systems*, Huntsville, AL, 1992.

[7] S. Szabo, K. Murphy, H. Scott, S.Legowik and R. Bostelman, "Control System Architecture for Unmanned Vehicle Systems", *Association for Unmanned Vehicle Systems*, Huntsville, AL,1992.

[8] W. A. Aviles, "Issues in Mobile Robotics: The Unmanned Ground Vehicle Program TeleOperated Vehicle", *Proceedings of the SPIE*, pp. 587-597, vol.1388, 1990.

[9] J. Kurtur, *Robotic Rover*, 1993, Systems Technology, MD.

[10] T. Schonberg, J. Suomela, A. Torpo and A. Halme. "A Small Scaled Autonomous Test Vehicle for Developing Autonomous Off-Road Applications", *Proceedings of the International Conference on Machine Automation*, 1994.

[11] N. Minar, R. Burkhart, C. Langton and M. Askenazi, *The Swarm Simulation System: A Toolkit for Building Multi-Agent Simulations*, Sante Fe Institute, 1996.

[12] M. J. Mataric, *Interaction and intelligent behavior*, Department of Electrical Engineering and Computer Science. 1994, Massachusetts Institute of Technology: Cambridge.

[13] A. Soto, M. Saptharishi, J. Dolan, A. Trebi-Ollennu, and P. Khosla, "CyberATVs: Dynamic and Distributed Reconnaissance and Surveillance Using All-Terrain UGVs", *Proceedings of the International Conference on Field and Service Robotics*, pp. 329-334, Pittsburgh, PA, August 1999.

[14] J. Dolan, A. Trebi-Ollennu, A. Soto, and P. Khosla, "Distributed Tactical Surveillance with ATVs", *SPIE Proceedings on Unmanned Ground Vehicle Technology*, vol. 3693, pp. 192-199, AeroSense '99, Orlando, FL, April 1999.

[15] C. Diehl, M. Saptharishi, J. Hampshire, and P. Khosla, "Collaborative Surveillance Using Both Fixed and Mobile Unattended Ground Sensor Platforms", *SPIE Proceedings on Unattended Ground Sensor Technologies and Applications*, vol. 3713, pp. 178-185, April 1999.

[16] K.S. Bhat, M. Saptharishi and P.K. Khosla, "Motion detection and segmentation using image mosaics," *Proceedings of the ICME*, pp. 1577-80, vol. 3, July 2000.

[17] R.Szeliski and H.Shum. "Creating full view pararomic image mosaics and environment maps," *Computer Graphics Proceedings, Annual Conference Series*, pp. 251-258, 1997.

[18] M. Irani, P. Anandan, J. Bergen, R. Kumar, and S. Hsu, "Mosaic representations of video sequences and their applications," *Signal Processing: Image Communication, special issue on Image and Video Semantics: Processing, Analysis, and Application*, Vol. 8, No. 4, pp.327-351, May 1996.

[19] F. Dufaux and F Moscheni. "Background Mosaicking for low bit rate video coding," *Proceedings of the IEEE ICIP*, pp. 673-676, September 1996.

[20] J. Shi and C. Tomasi. "Good features to track," *Proceedings of CVPR*, pp. 593-600, June 1994.

[21] S. Roberts and W. Penny. Novelty, confidence and errors in connectionist systems. Technical report, Neural Systems Research Group, Imperial College of Science, Technology and Medicine, April 1997.

[22] B. Schölkopf, R. Williamson, A. Smola, and J. Shawe-Taylor. SV estimation of a distribution's support. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*, Morgan Kaufmann, 2000.

[23] J. Hampshire II. *A Differential Theory of Learning for Efficient Statistical Pattern Recognition*. PhD thesis, Carnegie Mellon University, September 1993.

[24] C. Diehl. *Toward Efficient Collaborative Classification for Distributed Video Surveillance*. PhD thesis, Carnegie Mellon University, December 2000.

[25] I. Cohen and G. Medioni, "Detecting and tracking moving objects for video surveillance," *Proceedings of CVPR*, pp. 319-325, 1999.

[26] C. Wren, A. Azarbayejani, T. Darrell and A. Pentland, "Pfinder: real-time tracking of the human body," *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 51-56, 1996.

[27] C. Wren and A. Pentland, "Dynamic models of human motion," *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 22-27, 1998.

[28] S.J. McKenna, Y. Raja and S. Gong, "Tracking colour objects using adaptive mixture models," *Image and Vision Computing* 17, pp. 225-231, 1999.

[29] M.J. Black and A.D. Jepson, "EigenTracking: Robust matching and tracking of articulated objects using a view-based representation," *International Journal of Computer Vision*, vol.26, no 1, pp. 63-84, 1998.

[30] J. Rehg and T. Kanade, "Model-based tracking of self-occluding articulated objects," *Proceedings of the Fifth International Conference on Computer Vision*, pp. 612-617, 1995.

[31] A. Bobick, J. Davis, S. Intille, F. Baird, L.Campbell, Y. Irinov, C. Pinhanez and A. Wilson, "Kidsroom: Action recognition in an interactive story environment," M.I.T. TR No: 398, 1996.

[32] S. Ju, M.J. Black, Y. Yacoob, "Cardboard People: A parameterized model of articulated image motion," *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 38-44, 1996.

[33] K. Kant and S.W. Zucker, "Toward Efficient Trajectory Planning: Path Velocity Decomposition", *International Journal of Robotics Research*, vol. 5, pp. 72-89, 1986.

[34] S. Kato, S. Nishiyama, J. Takeno, "Coordinating mobile robots by applying traffic rules", *Proceeding of the IEEE International Conference on Intelligent Robots and Systems*, vol. 3, pp. 1535ff., IROS'92, Raleigh, (USA), July 1992.

[35] R. Alami, "A Multi-Robot Cooperation Scheme Based on Incremental Plan-Merging", *Proceedings of the Seventh International Symposium on Robotics Research*, Giralt & Gerhard (Eds.), Springer, 1995.

[36] E. Kruse and F.M. Wahl, "Camera-based Observation of Obstacle Motions to Derive Statistical Data for Mobile Robot Motion Planning", *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 662-667, Leuven, Belgium, May 1998.

[37] E. Kruse, R. Gutsche, and F.M. Wahl, "Acquisition of Statistical Motion Patterns in Dynamic Environments and their Application to Mobile Robot Motion Planning", *Proceeding of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 712-717, 1997.

[38] E. Kruse, R. Gutsche, and F.M. Wahl, "Estimation of collision probabilities in dynamic environments for path planning with minimum collision probability", *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1288-1295, 1996.

[39] R. Gutsche,C. Laloni and F.M. Wahl, "Path Planning for Mobile Vehicles Within Dynamic Worlds Using Statistical Data", *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 454-461, 1994.

[40] E. Besada, J.A. Lopez-Orozco, "Data Fusion for Building Maps", Universidad Complutense de Madrid, CC. Fisicas. Tech Report Dacya 99/12.