Content-Based Retrieval of Functional Objects in Video Using Scene Context*

Sangmin Oh, Anthony Hoogs, Matthew Turek, and Roderic Collins

Kitware Inc.

Abstract. Functional object recognition in video is an emerging problem for visual surveillance and video understanding problem. By functional objects, we mean objects with specific purpose such as postman and delivery truck, which are defined more by their actions and behaviors than by appearance. In this work, we present an approach for content-based learning and recognition of the function of moving objects given video-derived tracks. In particular, we show that semantic behaviors of movers can be captured in location-independent manner by attributing them with features which encode their relations and actions w.r.t. scene contexts. By scene context, we mean local scene regions with different functionalities such as doorways and parking spots which moving objects often interact with. Based on these representations, functional models are learned from examples and novel instances are identified from unseen data afterwards. Furthermore, recognition in the presence of track fragmentation, due to imperfect tracking, is addressed by a boosting-based track linking classifier. Our experimental results highlight both promising and practical aspects of our approach.

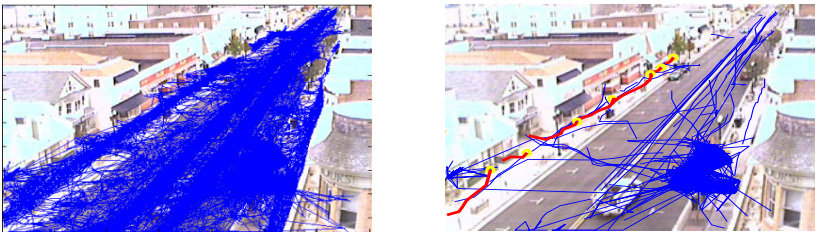
1 Introduction

Functional object recognition in video is an emerging problem for visual surveillance and video understanding problem. By functional objects, we mean objects with specific purpose such as postman and delivery truck, which are defined more by their actions and behaviors than by appearance. Yet, most object recognition algorithms attempt to classify objects based solely on appearance, largely because static imagery was the only data available. With the recent, explosive increase in video sensors, it is now possible to classify objects based on their movements and activities in applications such as surveillance and aerial videos.

In this work, we present an approach for content-based learning and recognition of the function of moving objects given video-derived tracks. As an example, Fig. 1(a) shows subset of our dataset where 1,442 tracks are computed from an uncontrolled webcam video spanning 86 minutes (among total 7 days). Fig. 1(b)

* This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. W31P4Q-09-C-0256. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA.

K. Daniilidis, P. Maragos, N. Paragios (Eds.): ECCV 2010, Part I, LNCS 6311, pp. 549–562, 2010.



(a) Track collection (b) Postman example

Fig. 1. (a) Collection of 1442 tracks spanning 86 minutes of webcam video. (b) Successfully identified sequence of a postman in a scene. His trajectory is fragmented into seven red tracks where yellow circles mark the beginnings of tracks, and many other simultaneous tracks are nearby (blue).

shows an example of a postman sequence (in red) which was successfully recognized among concurrent tracks (in blue). The video was recorded from a webcam located in Ocean City, NJ, USA. The frame rate is mostly 1Hz or worse, and the pixel-level noise is very high, contributing to degraded tracking performance typical in many surveillance video data.

The postman example illustrates the major challenges for functional object recognition. First, functional models should capture location-independent behavioral semantics, because location variability can be substantial across examples and only limited number of training examples are often available for interesting objects. Imagine the case of a delivery truck. It can literally stop at any parking spots, and person from the truck may visit any store in a scene. In our knowledge, learning of location-independent functional object models is novel, and mostly unexplored territory. Another core challenge is that the trajectories of target objects, marked in red in Fig. 1(b), are often fragmented into multiple tracks, and many other tracks (blue) are nearby in the same time interval – each track is too short to characterize the function, so that we must link tracks to identify functions. Tracking errors are often innate. Tracking algorithms can miss objects entirely, lose an object after tracking it for a while, or virtually any combination of these. Trackers are often optimized to avoid identity switching errors, which will result in greater track fragmentation.

Our solution for location-independent semantic behavior learning is to incorporate *scene context*. By scene context, we mean local scene regions with different functionalities such as doorways and parking spots which moving objects often interact with. They are 'contexts' because they are surrounding information, providing additional cues about mover's functions. Every track is encoded with Boolean features which capture its interactions w.r.t. scene contexts, e.g., the activity of people walking into roads can be characterized by attributes such as `move_on_sidewalk`, `move_towards_road`, and `move_on_road`. In particular, we explored the use of both manually and automatically obtained scene contexts. Based on these representations, functional models are learned from examples, and novel instances are recognized from unseen data. To model behavior

over time in the presence of track fragmentations, we have formulated a two-level modeling scheme. At the lower level, collected tracks are clustered based on features relating them to scene elements, resulting in elementary models corresponding to different categories of low-level behaviors such as "walking on sidewalk" and "crossing road". This scheme is motivated by the observation that tracks tend to be fragmented when there are substantial changes in low-level activities. At the higher level, composite full functional object models are learned in a supervised fashion, using the elementary models as building blocks, abstracting away low-level information. In terms of modeling regimes, we have investigated three approaches: (1) unigrams, (2) bigrams, and (3) HMMs.

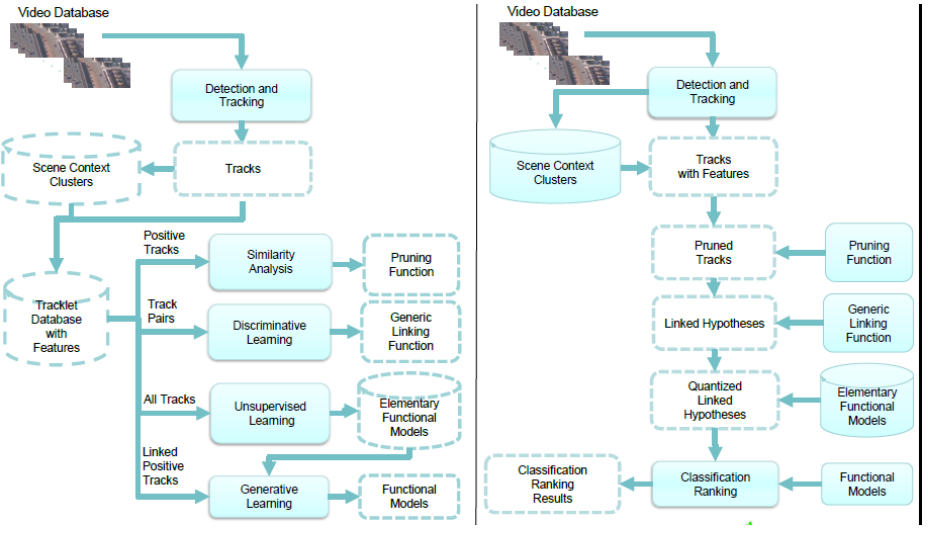
To address the fragmentation issues during recognition, we have developed a track linking classifier based on Adaboost.M1 [1]. The classifier computes link probabilities for every pair of tracks based on agreement between their features. Then, sequences of tracks with higher link probabilities are formed into functional behavior hypotheses to be evaluated against full functional object models. Additionally, a pruning scheme has been developed to filter out large portion of unrelated concurrent tracks prior to recognition, which leads to reduced number of hypotheses and alleviates computational demand substantially.

2 Related Work

The concept of functional object recognition for static objects was pioneered with work on recognizing chairs in static images [2]. Later, work has appeared on integrating observed human limb activities in video for static object recognition [3,4]. The notion of functional objects in this work goes beyond previous work in that they are actively moving entities which interact with environments.

Most work on trajectory analysis [5,6,7] focuses on grouping trajectories based on their locations and learning normalcy models. The resulting models are mostly location-dependent, primarily due to the adopted track features which heavily rely on either image or world coordinate information. In this work, we incorporate scene context features which largely abstract away location information. Accordingly, resulting models in our work depends less on location information and deliver interpretable semantics. A related work is [8] where the states of HMMs are semantic primitives such as `CloseTo` and `Moving`. In our work, however, semantic primitives are learned in an unsupervised manner.

The tracklet link classifier in this work differs from previous work [9,10] in that links are formed non-exclusively. The motivation is that we focus more on improving true positive retrieval ratios of links that belong to occurrences of functional objects, allowing multiple links to be formed from a track to other tracks, aiming to examine as many linked hypotheses as possible. The potential risk of exponential growth in number of formed hypotheses is compensated by pruning seemingly unrelated tracks prior to linking (see Sec. 6). Previous work addresses this issue by imposing exclusive linking constraints which is solved by global solutions such as Hungarian algorithm [9] or cost-flow network [10]. Our link classifier is learned within boosting framework with two noteworthy differences from hybrid boosting approach [10]. First, we directly learn binary link



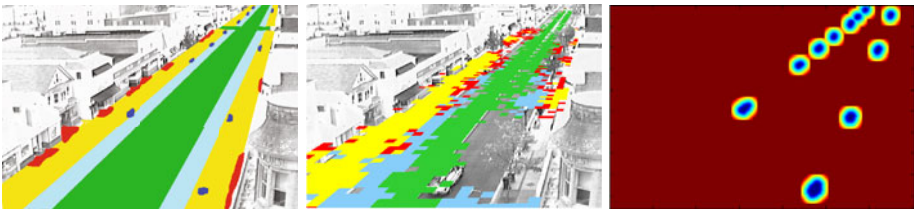
(a) Learning architecture (b) Recognition architecture

Fig. 2. (a) Architecture for learning functional object models. (b) Architecture for recognition. Solid boxes denote algorithmic processing units or existing information. Dashed boxes represent newly computed outcomes after every processing module.

classifier and do not need to adopt ranking-based methodology in [10], because we do not impose exclusive linking constraint through global solution in latter stages. Second, agreements between semantic Boolean features are used as crucial information for linking in our work, different from detailed kinematics and appearance features in [10]. In far-field videos, such features are less reliable due to the challenges of mover detection with accurate tight bounding boxes.

3 Overview of System Architecture

The overall architecture of functional model *learning* is illustrated in Fig. 2(a). Solid boxes denote algorithmic processing units or existing information. Dashed boxes represent newly computed outcomes, e.g., learned models, after every processing module. First, input video is stabilized and geo-registered (not shown), then object tracks are extracted using our tracking method which uses background subtraction and performs global multi-object tracking, similar to [9]. Note that our overall approach does not depend on the tracker, and in principle any video detection and tracking system could be used. For scene contexts (Sec. 4), manually labeled information can be used. Otherwise, scene contexts can be automatically identified. Then, tracklet database is formed where each track is attributed with comprehensive set of features, including its interaction with scene contexts (Sec. 4). Then, four major modules are learned from the feature-indexed database: (a) pruning function (Sec. 6), (b) a track linking function



(a) Manual (b) Unsupervised (c) Image Distance map

Fig. 3. (a) Manual scene context labels: road (green), parking spots (light blue), sidewalks (yellow), building entrances (red), and trash cans (dark blue). (b) Unsupervised scene context learning results with four clusters. (c) An example distance map on image to trashcan context (in blue on the leftmost figure).

(Sec. 6), (c) shared elementary functional models (Sec. 5) , and (d) full composite functional models of interest (Sec. 5). In particular, a set of elementary functional models are learned from a large pool of tracks, relations and actions using unsupervised clustering. Then, we learn full composite functional models of interest from a selected set of linked positive examples where the pre-computed elementary functional models are used as building blocks. A noteworthy advantage of the current architecture is that most of the intermediate computational results such as scene context clusters, generic linking function, and elementary functional models can be reused to learn new functional models.

In the *recognition* phase shown in Fig. 2(b), novel data are fed into the system either as video streams or a set of video clips from which object tracks are extracted. The same features and relations used in learning phase are computed for each track. Then, potentially irrelevant tracks w.r.t. the functional object under search are pruned using learned pruning function, leaving only promising ones for further processing. The survived set of tracks are linked via the learned generic linking function to yield linked hypotheses. Then, every linked hypothesis is quantized into a sequence of elementary functional models which will be scored w.r.t. a full functional model. Finally, either ranking methodology or detection thresholds are applied to suggest promising instances to operators.

4 Scene Context and Track Features

Scene Contexts. Once track(let)s are computed, the next level of representation is the characterization of tracks by relations between tracks and scene. In particular, the manner and timing of semantic interactions between a moving object and nearby static scene contexts can indicate significantly different types or behaviors. Fig. 3(a) illustrates five different types of manually identified and labeled scene contexts: road, parking spots, sidewalks, building entrances, and trash cans. Such manual information is becoming increasingly available through various geo-spatial databases, e.g., Google Maps. On the other hand, scene regions with different functionalities can be automatically grouped in an unsupervised manner. In this work, unsupervised scene contexts are obtained by

Table 1. List of track features in three categories: (1-7) track-level, (8-46) contextual, and (47-48) composite. The rightmost three columns indicate which features are used for latter stages of linking (L), track clustering (C), and pruning (P).

ID	*Category*	*Type*	*Feature Description*	*L*	*C*	*P*
1	*Track*	*Continuous*	*2D initiating locations in world*	*o*		
2	*Track*	*Continuous*	*2D terminating locations in world*	*o*		
3	*Track*	*Continuous*	*2D initiating locations in image*	*o*		
4	*Track*	*Continuous*	*2D terminating locations in image*	*o*		
5	*Track*	*Continuous*	*2D average speed in world (m/s)*	*o*		
6	*Track*	*Continuous*	*Initiating time (in seconds)*	*o*		
7	*Track*	*Continuous*	*Terminating time (in seconds)*	*o*		
8	*Track*	*Boolean*	*Tracking bounding box size indicates person?*	*o*	*o*	*o*
9	*Track*	*Boolean*	*Tracking bounding box size indicates vehicle?*	*o*	*o*	*o*
10	*Track*	*Boolean*	*Fast moving (within normal vehicle speed)?*	*o*	*o*	*o*
11	*Track*	*Boolean*	*Slow moving (within normal human speed)?*	*o*	*o*	*o*
12-26	*Context*	*Boolean*	*Move on scene contexts within world?*	*o*	*o*	*o*
27-31	*Context*	*Boolean*	*Move nearby scene contexts within world?*	*o*	*o*	*o*
32-36	*Context*	*Boolean*	*Move nearby scene contexts within image?*	*o*	*o*	*o*
37-41	*Context*	*Boolean*	*Move away from scene contexts within world?*	*o*	*o*	*o*
42-46	*Context*	*Boolean*	*Move toward scene contexts within world?*	*o*	*o*	*o*
47	*Composite*	*Boolean*	*Possibly a human?*	*o*	*o*	*o*
48	*Composite*	*Boolean*	*Possibly a vehicle?*	*o*	*o*	*o*

accumulating the intersecting track behaviors per region, and clustering them, similar to [11]. Additionally, it is worthwhile to note that functional scene region detectors can be trained in a supervised manner, and can be used to detect semantic concepts such as parking spots or doorways [12]. Fig. 3(b) shows the scene context clusters obtained through this approach where parameters were tuned to produce similar number of clusters. It can be seen that the results are fairly interpretable and similar to manual labels. Each unsupervised cluster delivers interpretations of road, sidewalk, parking spots, and short activity areas such as garbage cans and partial doorways. The experimental results in Sec. 7 report recognition results based on both manual and unsupervised scene contexts.

Track Features. Every track is attributed with three categories of features: (1) track-level, (2) contextual, and (3) composite. Track-level features consists of both continuous and Boolean features while the other two categories include only Boolean features. Manual camera calibration provided a homography mapping image locations to the ground plane. First, track-level features record information such as speed and location. Second, context features capture the interactions between tracks and computed scene contexts both on image and world coordinates. Third, composite features roughly categorize tracks to be human or vehicle based on heuristics using information such as bounding box size, speed, and location (on sidewalk or road etc). This level of information should capture all salient aspects of functional object behavior. The entire set of features are enlisted in

(a) Vehicles parking. (b) People on sidewalk. (c) A location cluster.

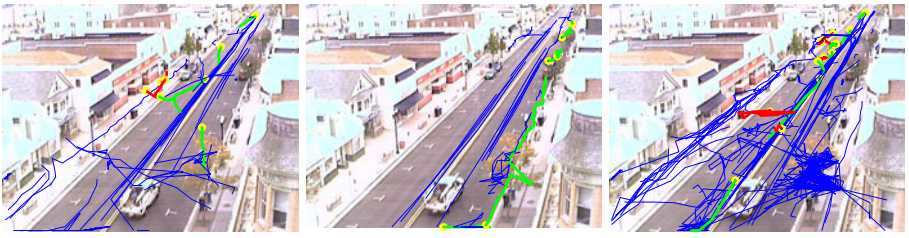
Fig. 4. (a-b) Two cluster results based on our approach (among total eleven). Each class delivers interpretable behaviors : parking and people on sidewalk. (c) A cluster obtained by [6]. Green and red marks indicate track heads and tails.

Table 1. Note that Boolean features are not mutually exclusive, e.g., both composite features can be true, indicating that a track maybe both human and a vehicle, embracing the uncertainty and mitigating more strict decision to future computational modules. Context features capture interactions based on the changes of distance within every track to scene contexts. Fig. 3(c) shows a precomputed context distance maps on image for the trash can scene context (shown in blue in Fig. 3(a)). Using distance maps on both image and world coordinates in conjunction with tracking, we can compute interactions efficiently. While most types of context features include 5 features, one each for every manual scene context, `Move_on` type includes 10 additional features, totaling 15. Tracks frequently move across different scene contexts, and 10 additional unordered pairwise fields were formed (out of 5), e.g., `Move_on_Sidewalk_and_`ParkingLot, to encode such behavior. Note that different subsets of features are used at different future processing modules (marked in Table 1) which include link classifiers (L), elementary function clustering (C), and pruning function (P).

5 Functional Object Model Learning

To model functional object behavior over time in the presence of track fragmentations, we have formulated a two-level hierarchical approach. At the first lower-level, a codebook for individual track quantization is learned to provide a vocabulary of low-level activities based on encoded features. The learned codebook is then used to assign every track to one of different elementary functions. At the higher level, full function models are learned from a sequence of quantized tracks, regardless of the detailed feature information encoded in every track.

Elementary Functional Model Learning. Identification of elementary functional models is conducted by clustering all tracks based on contextual and composite features. We have explored four different clustering methods: K-means, mean-shift, spectral clustering [13], and affinity propagation [14]. The resulting clusters represent a group of tracks that share a common set of features. We have found that affinity propagation [14] produces superior clusters in terms of



(a) Delivery truck (b) Road cleaning vehicle (c) Trash truck

Fig. 5. Three examples of functional objects: (a) delivery truck, (b) road cleaning vehicle, and (c) trash truck. Tracks belonging to human and vehicle movers are shown in red and green respectively where yellow circles mark the beginnings of tracks belonging to functional objects. Blue tracks are concurrent movers.

interpretability with minimal parameter tuning efforts. Fig. 4(a-b) show tracks within two sample clusters among total eleven, which deliver interpretations of vehicles parking and people on sidewalks, all independent of event locations. Although omitted for brevity, other clusters captured activities such as vehicles passing through and people crossing roads. As a comparison, a trajectory analysis method [6] has been applied where a sample cluster dominated by spatial distribution is shown in Fig. 4(c), only showing pedestrians on the left sidewalk.

Full functional object model learning. We adopted generative learning approaches for full functional model learning. This is because the size of positive training examples are often limited and the identification of negative examples are challenging for content-based retrieval problems. Our goal is to learn full functional model for long-term object activities from a given set of positive example which are assumed to be in the form of a sequence of manually linked tracks. For example, four functional objects which include postman, delivery truck, road cleaning vehicle and trash truck are shown in Fig. 1(b) and Fig. 5(a-c). It can be seen that each example consists of fragmented tracks. For delivery truck and trash truck, tracks even switch between multiple movers, i.e., people (red) and vehicle (green). Three modeling regimes are explored : unigrams, bigrams, and HMMs. Unigram simply counts unique symbols individually, while bigrams count the unique consecutive pairs. These three models provide different spectrum on amount of information they can capture. Since we use elementary functional models to discretize the tracks within sequences, every sequence example is represented as a series of symbols. For example, a sequence may be represented as: `1121` for a four-track sequence. Additionally, we insert gap variables, e.g., zero(`0`), whenever there is a temporal gap between tracks. This modification yields a sequence representation of : `1010201` (assuming temporal gaps everywhere). The number of parameters to be estimated for each of the three modeling regimes is in increasing order of unigrams, bigrams, and HMMs.

We use unigrams and bigrams within sample-based learning framework. This approach effectively converges to nearest-neighbor classifier. There are several well-known distance metrics that measure how distinct two bag-of-words distri-

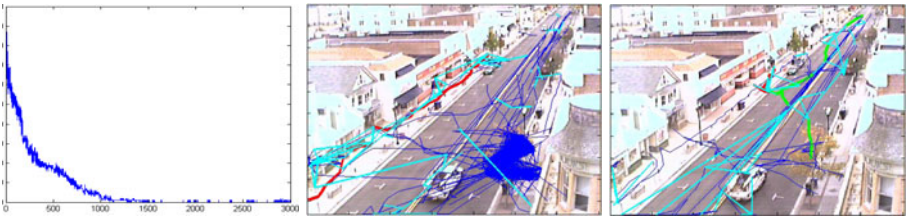


Fig. 6. (a) Boosting algorithm decreases linkage classification errors as increasing number of weak classifiers are learned. The x-axis shows the progress of boosting learning as the number of weak classifiers, and y-axis shows classification error. Detected links overlaid (in cyan) on sequences of (b) a postman and (c) a delivery truck.

butions are. In this work, we used Bhattacharyya distance. For HMM learning, we used standard initialization with uniform priors and EM learning with Dirichlet priors to compensate for limited amount of training examples. The number of hidden states was set to be the number of elementary functions.

6 Track Linking, Pruning, and Linked Hypotheses

To recognize functional objects in the presence of track fragmentations, links between tracks should be identified to generate linked hypotheses. Our insight is that, if two fragmented tracks are a source and destination pair which belong to an identical mover, then a large portion of the feature distributions on these tracks will agree. Every track is encoded with comprehensive set of features (see Table 1) which consist of two types of features : Boolean and continuous. If both Boolean features in two separate candidate tracks agree, we assign '`True`' to the corresponding dimension in the new pairwise feature, and '`False`' otherwise. For continuous features such as velocity and location, the absolute difference between the two values is computed. Additionally, we have included additional features which may be potentially useful: the number of total agreed Boolean features, and the distance between two tracks.

We have used 'Adaboost.M1' [1] with decision stumps to learn a linking function. Fig. 6(a) shows decreasing errors w.r.t. the increasing number of weak classifiers. Qualitative results of automatic linking on datasets of a postman and a delivery truck are shown in Fig. 6(b) & (c). (see Fig. 1(b) and Fig. 5(a) for originals) where the detected links (cyan) are overlaid. The tracks belonging to the primary function tracks are well-linked, with minor number of confuser links. Based on the outputs from Adaboost link classifier, track sequences with higher link probabilities are formed into functional behavior hypotheses to be evaluated against full functional models. The foremost concern with non-exclusive linking approach is that the number of hypotheses can grow (approximately) exponentially w.r.t. the identified potential links. For example, during our studies, we have seen impractically large number of 1 billion hypotheses are generated from 15 minute query video. It is crucial that number of links to be reduced. There

are two potential approaches to address this problem: (1) build object-specific linking function, and (2) prune tracks that seemingly do not belong to the functional object class of interest prior to linking. The first approach , however, did not work well because the limited number of training examples per class (often the case for content-based retrieval) made successful learning difficult.

Accordingly, a pruning scheme has been developed to filter out large portion of unrelated concurrent tracks prior to recognition, which leads to reduced number of hypotheses and alleviates computational demand substantially. Our approach is to prune out tracks which demonstrate little similarity to the positive example tracks belonging to the function class of interest. Our successful solution is to directly use the available Boolean context features on a track to measure the similarity against the provided training tracks (see Table 1). When there are substantial number of identical Boolean context features between a candidate track and tracks within positive training dataset, we assume that it is more likely to be kept as candidate tracks, otherwise, it will be pruned out prior to linking. Similar to linking, we have used the number of agreed Boolean features as similarity score for pruning. To obtain a threshold θ for pruning, min-max similarity across positive examples is used: $\theta = \min(\{\theta_i | \theta_i = \max_{i \neq j}(\{\theta_{ij}\})\})$. Here, θ_{ij} denotes the similarity between two training tracks. For a novel tracklet, if there exists a training track with similarity score higher than the threshold, it is kept, otherwise, it will be pruned. In our test, the pruning module eliminated about 90% of negative examples while it kept most of the promising tracks (>97%). We have also explored the use of related max-min thresholds, however, it turned out to keep unnecessarily large number of tracks, lowering negative example pruning rate close to 50%. In summary, as the result of combined prior-pruning-then-linking approach, the number of generated hypotheses was reduced by several orders of magnitude where the maximum from a set of 15 minute videos was at most 2500, which is within manageable bounds.

7 Experimental Results

Link Classifier and Hypotheses Generation. To assess quantitative performance of the developed linking and hypotheses generation framework (Sec. 6), we have tested our work along with two additional linking methods on tracks collected from webcam data. First, we look into the generic linking accuracy of developed linking functions. By generic linking, we mean that linker accuracy will be assessed based on test dataset not being limited to the ones that contain functional object sequences. Linker function outputs either link probabilities (the case of Adaboost in our work) or link scores. Each of the i-th element in training data for AdaBoost is in the form of (x_i, y_i) where x_i is a multi-dimensional feature vector and $y_i \in \{0, 1\}$ is a label. For AdaBoost training, $N_p(\approx 250)$ positive examples and $N_n(\approx 500)$ negative examples were used. As competing methods, we have implemented two additional linker functions and learned the parameters through training. The first link function is learned based on RankBoost [10]. The training data consists of pair of feature vectors $(x_{i,0}, x_{i,1})$ where the preference/rank of the the first item is higher than the second. All the features used for

AdaBoost were re-used and additional track smoothness features which measure the kinematic continuity between tracks used in [10] were included. RankBoost learning process generates a strong rank function $H(x) = \sum_t \alpha_t h_t(x)$ which consists of linear chain of weak ranker $h_t(x)$ where we used decision stumps. It is desirable to assess the learning capability of AdaBoost and RankBoost given equal amount of training data. From the training data used for AdaBoost, we created $(N_p + N_n)$ number of preference pairs where $x_{i,0}$ and $x_{i,1}$ belong to positive and negative training dataset respectively. In addition, as pointed out by [10], the outputs of RankBoost classifier does not deliver precise interpretations as probability within range [0,1]. We used logistic regression to map the outputs of RankBoost to probabilities. The second link function implemented is based on more traditional idea which outputs several costs $\{c_i\}$ based on kinematic continuity and appearance similarity, e.g., [9]. In our work, we have considered such costs between tracks as link features and learned a logistic regression function $1/(1 + e^{-\sum w_i c_i})$ as a link classifier where the weight parameters $\{w_i\}$ were learned from available training data. Newly developed contextual features were not used for this linker for comparison purposes.

The generic-linking test results of all three link classifiers on test dataset of ($\approx$ 300) positive and ($\approx$ 500) negative examples are shown on the left side of Table. 2. Probability of detection (PD) and false positive rate (FP) are shown. A standard threshold of 0.5 was used as decision boundary. It can be observed that boosting-based methods outperform the traditional weighted score method in terms of PD while FPs are all similarly low. Boosting-based methods effectively exploit diverse features. On the other hand, the conventional features used for weighted score linker are presumably less reliable, mostly due to the low resolution and low frequency characteristics of the video clips. For example, low-level kinematics features which rely heavily on accurate high-frequency dynamic information is not captured well and often rejects true links by mistake. The superiority of AdaBoost over RankBoost can be attributed from two aspects. Theoretically, there is no guarantee that RankBoost will actually outperform AdaBoost for classification tasks. Accordingly, use of larger training dataset may be needed for superior performance.

In addition, we conducted experiments to assess the benefits of various submodules for identifying long-duration linked trajectories of functional objects, using 13 video clips containing functional objects.We applied all three linker functions along with two optional modules: pruning (P) prior to linking and Hungarian method (H) to impose exclusive links. The results of three combinations of approaches are reported on the right side of Table. 2. The average number of tracks per video clip was 103.9 where the average number of generated hypotheses and the PD which denotes the ratio of datasets where the generated hypotheses included full trajectories of functional objects are shown in Table. 2. If both optional modules are turned off, the space of all linked hypotheses becomes impractically large occasionally and the results are omitted. Two important observations can be made from Table. 2. First, the pruning process indeed reduces the number of generated linked hypotheses, nonetheless, it

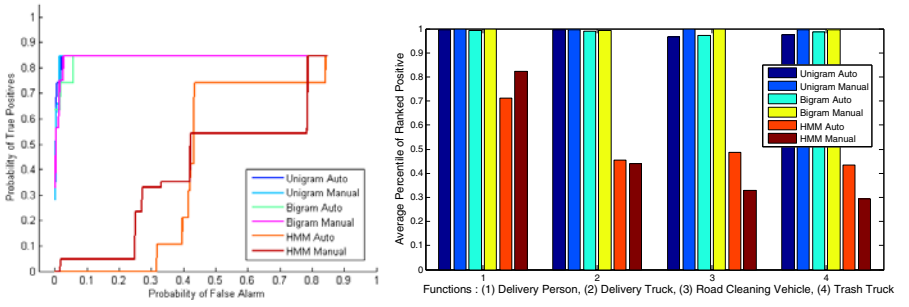
Table 2. Average performance statistics of different linking algorithms over total 13 datasets that contain functional objects. 'P' denotes the use of pruning step prior to linking. 'H' refers to the use of Hungarian method to impose exclusive linking.

	Generic Linking		Linked Hypotheses				
	PD	FP	#Tracks	P	H	# Hypotheses	PD
AdaBoost (ours)	0.82	0.08	103.9	○		341.8	0.85
					○	41.1	0.38
				○	○	35.2	0.38
RankBoost (logistic regression)	0.70	0.05		○		180.1	0.30
					○	29.3	0.08
				○	○	24.8	0.08
Weighted scores (logistic regression)	0.61	0.09		○		130.2	0.08
					○	24.4	0.0
				○	○	19.7	0.0

does not affect the overall PD to be lower. Second, although exclusive linking substantially lower the number of generated hypotheses, PDs lower as well. The results suggest that, if considerations can make brute-force style search more affordable, it will be worthwhile to pursue such direction.

Functional Object Recognition. Our primary metric for functional object recognition is ROC curves. In our efforts, samples refer to the set of all generated hypotheses. If a sample contains the entire sequence of ground truth tracks, it is considered to be a 'hit', i.e., a super-set hypothesis of a positive sample is still a hit. Test samples are either scored by Bhattacharyya distances (for unigrams and bigrams), or by data likelihood (for HMMs). In terms of decision thresholds, we adopt top ranking number as such measure. Top ranking number denotes the number of highest-scored hypotheses that are to be returned as retrieval results.

Four functional objects selected from our webcam dataset were considered: postman, delivery truck, road cleaning vehicle, and trash truck. We have manually cropped total of 13 video clips from our webcam dataset where durations range from 3 to 15 minutes. Each of these clips loosely contains an example object along with approxately 1 min of extra amount of video before and after occurrences of examples. The original number of video clips for four functional classes are : 2-3-4-4. To obtain additional realistic data, we perturbed the obtained tracks to create three additional perturbed clones per track, resulting in dataset sizes of : 8-12-16-16. For the learning of elementary functional models and linking functions, we used original clips only (total 13). The exclusion of perturbed dataset during training was to assess the generalization power across different datasets. For every learning and recognition of full functional object trajectories, leave-one-out experiments are conducted. In the case of postman class, 7 positive examples are used in training process and one left-out example is included in the test dataset along with all the other available datasets which do not belong to the trained functional category (total 44). On average, the size of the target functional object class data constitutes about 3% of each batch of test sets. The ROC curve for a particular functional class is computed as the average



(a) Delivery truck ROC (b) Average ranking ratios

Fig. 7. (a) ROC curve for delivery truck dataset. 'Manual' and 'Auto' refer to the use of manual and learned scene context clusters respectively. Results of six different approaches: [Unigram, Bigram, HMM]×[Auto, Manual]. Note : bigram results are similar to unigram results, and may not be visible due to unigram curves. (b) Average ranking ratios of for four functional objects under six varied experimental settings. Higher ranking ratios represent more accurate classification.

from multiple experiments conducted for that particular class. In particular, we have conducted the whole pipeline of learning and recognition experiments using both manual (see Fig 3(a)) and automatically obtained scene contexts (see Fig. 3(b)) to assess how much impact the accurate manual identification of scene contexts make. Then, normalized average rankings (AR) were obtained from the ranking results w.r.t. the total number of generated hypotheses. Accordingly, an AR close to 1.0 indicate accurate classification.

An example ROC curve for delivery truck class is shown in Fig. 7(a). An interesting outcome is the promising performance of unigrams and bigrams: both achieve very high PDs at very low cost of FPs. Considering the simplicity and computational efficiency associated with these models, the accurate identification results show that they can capture the general characteristics of particular functional classes well enough to yield favorable recognition. Another finding from the ROC curve analysis is that simpler models such as unigrams and bigrams are outperforming more sophisticated counterpart such as HMMs. The observed weak performance of HMMs can be explained from the generalization point of view. Given the limited number of training samples which ranges from 7 to 15 samples, comparably far larger number of HMM parameters fail to capture the general characteristics of data. These encouraging results for simpler models are likely due to the current problem setting: content-based learning with limited number of training examples. Analogous analysis can be drawn from the average ranking ratio results for all four functional objects under six experimental settings, shown in Fig. 7(b). The ROC curves for the other three object classes showed very similar characteristics (omitted for brevity). Note that among four classes, delivery truck and trash trucks are more challenging because they include both person and vehicle movers. No hit hypothesis could be generated from a few datasets belonging to these two classes, accordingly, corresponding

ROC curves never achieve perfect PD of 1.0 (see Fig. 7(a)). Additional finding is that the overall recognition degrades only marginally even when unsupervised scene contexts are used, in comparison to more accurate manual contexts. This observation suggests that rough identification of scene contexts may be sufficient to deliver favorable functional object recognition results in many cases. It would be worth noting that our results may be more optimistic than true reality, primarily due to the facts that perturbed data is used and the ratio of true positive examples may be less in practice than our current experimental setting, probably appearing less than 1% of time while these objects constitute average 3% of data in this work. We plan to investigate these issues in our future work.

References

1. Freund, Y., Schapire, R.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and System Sciences 55, 119–139 (1997)
2. Stark, L., Bowyer, K.: Achieving Generalized Object Recognition through Reasoning about Association of Function to Structure. PAMI 13, 1097–1104 (1991)
3. Peursum, P., West, G., Venkatesh, S.: Combining image regions and human activity for indirect object recognition in indoor wide-angle video. In: ICCV (2005)
4. Gupta, A., Davis, L.: Objects in Action: An Approach for Combining Action Understanding and Object Perception. In: CVPR (2007)
5. Junejo, I., Javed, O., Shah, M.: Multi feature path modeling for video surveillance. In: ICPR (2004)
6. Wang, X., Ma, K.T., Ng, G.W., Grimson, E.: Trajectory analysis and semantic region modeling using a nonparametric Bayesian model. In: CVPR (2008)
7. Yang, Y., Liu, J., Shah, M.: Video scene understanding using multi-scale analysis. In: ICCV (2009)
8. Chan, M., Hoogs, A., Schmiederer, J., Petersen, M.: Detecting rare events in video using semantic primitives with HMM. In: CVPR (2006)
9. Perera, A., Srinivas, C., Hoogs, A., Brooksby, G., Hu, W.: Multi-object tracking through simultaneous long occlusions and split-merge conditions. In: CVPR (2006)
10. Li, Y., Huang, C., Nevatia, R.: Learning to associate: Hybridboosted multi-target tracker for crowded scene. In: CVPR (2009)
11. Turek, M.W., Hoogs, A., Collins, R.: Unsupervised learning of functional categories in video scenes. In: ECCV (2010)
12. Swears, E., Hoogs, A.: Functional scene element recognition for video scene analysis. In: IEEE Workshop on Motion and Video Computing (2009)
13. Zelnik-Manor, L., Perona, P.: Self-tuning spectral clustering. In: NIPS (2006)
14. Frey, B.J., Dueck, D.: Clustering by passing messages between data points. Science 315, 972–976 (2007)