Pyramid Coding for Functional Scene Element Recognition in Video Scenes

Eran Swears[1], Anthony Hoogs[1], and Kim Boyer[2]

[1]Kitware Inc., {eran.swears|anthony.hoogs}@kitware.com
[2]ECSE Department, Rensselaer Polytechnic Institute, kim@ecse.rpi.edu

Abstract

Recognizing functional scene elements in video scenes based on the behaviors of moving objects that interact with them is an emerging problem of interest. Existing approaches have a limited ability to characterize elements such as cross-walks, intersections, and buildings that have low activity, are multi-modal, or have indirect evidence. Our approach recognizes the low activity and multi-model elements (crosswalks/intersections) by introducing a hierarchy of descriptive clusters to form a pyramid of codebooks that is sparse in the number of clusters and dense in content. The incorporation of local behavioral context such as person-enter-building and vehicle-parking nearby enables the detection of elements that do not have direct motion-based evidence, e.g. buildings. These two contributions significantly improve scene element recognition when compared against three state-of-the-art approaches. Results are shown on typical ground level surveillance video and for the first time on the more complex Wide Area Motion Imagery.

1. Introduction

We present a new approach to video scene modeling and scene element recognition that makes several improvements over existing state-of-the-art approaches. More specifically, we recognize stationary scene elements in video using descriptors derived from the moving objects (people/vehicles) that interact with them. When these scene elements have a specific purpose or function they are referred to as functional scene elements [1,2,3]. Some examples include: parking-spot, sidewalk, building, doorway, cross-walk, and bus-stops. Relying on descriptors derived from automatically computed tracks, as opposed to pixel features, enables the detection of scene elements that cannot be discriminated based on appearance alone. For example, cross-walks may or may not have the black and white zebra patterns and doorways can be completely occluded in high altitude aerial video or have very few pixels, as is the case with one of the aerial video datasets analyzed here (Figure 1). Fortunately, the moving objects that interact with them are easier to detect [9] and track [10] which enables the detection of these visually ambiguous or poorly seen elements.

Existing functional scene element recognition approaches either characterize the scene elements by clustering descriptors/features from individual grid cells [3,4], a flat layer of clusters [2,3], and/or through the use of manually defined scene element detectors [1,4]. These work well when the scene elements have a sufficient number of moving objects with well-defined behaviors passing over them. For example, when there are many examples of pedestrians crossing the road on the cross-walk. But, they can fail to recognize scene element when the activity is low, multi-modal, or indirect. Multi-model elements have multiple behavior characteristics; for example, roadways have vehicles driving on them but they can also have vehicles stopping and turning to enter a parking-spot. Additionally, indirect activity is when the activity associated with a scene element (building) occurs nearby (person-entering-building), but not within the scene element's bounds (no walking on the roof).

Our solution for recognizing scene elements with low, indirect, and/or multi-model activity is to introduce a new pyramid coding approach that creates a hierarchy of descriptive clusters to form a pyramid of codebooks over a local behavioral context window. Our first contribution is the characterization of scene elements using a sparse-dense pyramid of codebooks and the path of the descriptors through the pyramid. This pyramid coding approach implicitly captures all behavior granularities, up

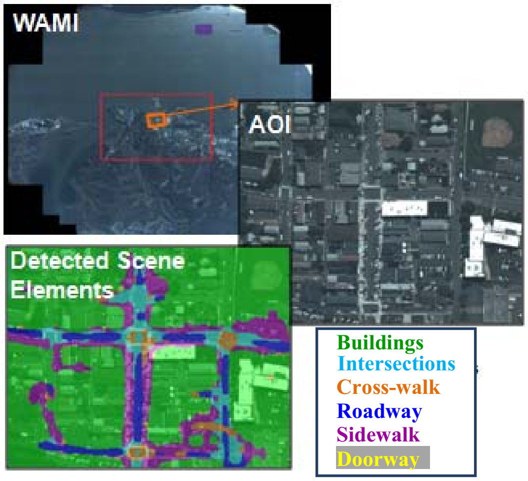


Figure 1, (Top) ARGUS-IS collected data stitched full frame image, (Middle) Defined AOI around main street, (Bottom) detected functional scene elements.

to a maximum number, to enable the characterization of multi-modal behaviors. The most similar approach to ours uses Hierarchical K-Means (HKM) to generate a "vocabulary tree" [14] for characterizing objects/images. However, this approach focuses on using the leaf clusters through entropy weighting and divides each cluster into K more clusters independent of the variability in their data, which can result in millions of sparse clusters. Our approach uses clusters from all layers in the pyramid and produces a smaller number of clusters that are dense in content. The dense clusters are created by bifurcating clusters based on the variance of their assigned data. Our bifurcation process also has an inherent subset of unique clusters (sparse pyramid) that are used to characterize the full pyramid. This sparse pyramid results in $2(K-1)$ dense clusters instead of the $K(K-1)/2$ clusters from the full pyramid or the HKM's K^L sparse clusters, where K is the number of clusters and L the number of layers.

Our second contribution is the incorporation of local behavior context to compensate for both the low and indirect activity. Local behavioral context is captured by aggregating (pooling) behaviors from that surround the scene element of interest, not just a single grid cell [1,2,3,4]. This increases the observed amount of activity and couples the scene element with nearby activity.

Our overall approach recognizes spatial regions that have similar functional behaviors as the presented training examples. Our framework for this is similar to those used by standard Pyramid Matching (PM) approaches for image/object classification [6,12], where there is a coding and a pooling step. The components of our coding step are shown in Figure 2 where we start with a set of descriptors derived from moving objects indicated by $\boxed{1}$, see section 3. The spatially independent descriptors are then fed into the pyramid coding algorithms $\boxed{2}$ that use hierarchical divisive clustering based on Gaussian Mixture Models (GMMs) to form the pyramid of codebooks. This clustering process results in two unique clusters per layer as indicated by the red and blue clusters, where the red cluster has the highest variance and is bifurcated.

After pyramid coding, a 2D spatial grid is applied to the scene's ground plane and encoded once for each codebook in the pyramid of codebooks $\boxed{3}$. The encoding process first assigns descriptors to clusters (codewords) and then assigns the label of the most frequently occurring cluster within each grid cell to that grid cell. Each encoded scene is referred to as a functional region map [3].

The scene element models are formed during the pooling step, where one model is created for each training example. Pooling involves accumulating the unique clusters/codewords for the Regions of Interest (ROIs) from each layer's functional region map into a histogram model. To reduce processing time during the recognition process the unique codewords from the functional region maps are stored as integral images during training.

The testing process is a recognition framework that identifies both the location and label of scene elements. During the testing process an "unknown" histogram model from a test ROI is compared to each learned model which returns the likelihood of fitting to each. The scene is raster scanned with the test ROI to produce a 2D likelihood map that is later smoothed with a Markov Random Field.

Results are shown on two datasets, the Ocean City dataset from [1,2,3] and the "Autonomous Real-Time Ground Ubiquitous Surveillance-Imagery System" (ARGUS-IS) collected WAMI data. To date, no functional scene modeling approaches have been applied to WAMI data, which offers more challenges such as a more diverse set of behaviors and fewer pixels on vehicles and pedestrians (movers). Our experiments show how modeling local context along with applying the pyramid to the coding step significantly improves recognition results, particularly when compared to the most relevant coding [14] and functional recognition [2,3,4] approaches.

2. Relevant Work

Swears and Hoogs [1] introduced functional scene element recognition in outdoor surveillance video. This approach uses manually defined Bayesian classifiers and weak activity detectors to accumulate 2D likelihood maps over a scene for the elements of interest. This was later

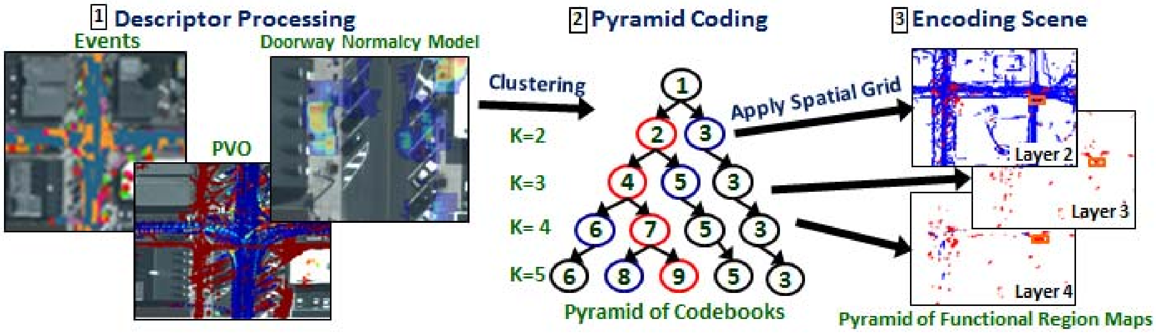


Figure 2, Overall pyramid coding approach. The track based descriptors $\boxed{1}$ are fed into the hierarchical divisive clustering process (red/blue clusters are unique) $\boxed{2}$ to produce the pyramid of codebooks. These are then used to encode the scene $\boxed{3}$ for each layer.

extended in [2] by converting the likelihood maps to track descriptors and passing them into a hierarchical divisive clustering algorithm. The Functional-Category approach in [3] is a completely unsupervised method that clusters histograms of descriptors using a flat mean-shift clustering algorithm. These approaches only use the leaf or flat layer of clusters to characterize functional scene elements and do not take local context into account.

The functional scene element recognition approach in [4] implements supervised binary scene element detectors to produce 2D likelihood maps for each element and then imposes local class adjacency constraints to perform spatial smoothing with a Markov Random Field (MRF). However, it does not scale to a wide variety of descriptors and there is no local behavioral context taken into account. The work in [5] offers a more complex approach that uses manually defined complex Markov Logic Networks to recognize interactions between moving objects specific to the scene element of interest. However, the logic representation is limited to evidence that has well-defined semantic meaning, which is not always available, is subjective, and requires a subject matter expert to define.

Other work to classify images/objects uses HKM clustering [14] to form the pyramid of codebooks. This work has shown that a larger set of leaf clusters leads to improved recognition when focusing on the leaf clusters. However, our work shows that using our dense clusters from all layers in the model leads to improved recognition over emphasizing sparse leaf clusters.

3. Track Based Descriptors

Our pyramid coding algorithms can use virtually any feature derived from detections or tracks, where both are referred to as track based descriptors. Moving objects are detected in video using a standard background subtraction algorithm [9] and then associated to tracks [10] resulting in multiple detections per track. Tracks are converted to the ground plane using an approximate image-to-ground projective camera model that is computed from available metadata. The tracks are then processed through event detectors [4,5], track-type classification [1], and normalcy modeling algorithms [1,2]. Table 1 shows the set of descriptors used here. For brevity only summaries of these algorithms are discussed here, see [1,2,4,5] for details.

Simple low-level event detectors based on speed thresholds are used here to generate the probability of events on a per detection basis such as vehicle-stopping and vehicle-driving-fast. Similarly, the vehicle-turning event detector is based on angular difference thresholds. The person/vehicle/other (PVO) classifier descriptors are generated from a simple Bayesian classifier where the parameters for the person, vehicle, and other classes have been manually defined, as in [1,2]. Spatial normalcy models are 2D likelihood maps that show where a behavior of interest is more likely to occur. A portion of the normalcy model for doorways is shown in Figure 2 $\boxed{1}$ as a heat map, where darker red regions indicate where doorways are likely to exist. These normalcy models are generated by accumulating evidence from weak detectors over time to produce stronger signatures [1,2]. Each of the track's detections, n, is assigned a value from the D descriptors resulting in a $x_n \in \mathbb{R}^{1\times D}$ descriptor vector.

Table 1, Track based descriptors including PVO classification, event detection, and normalcy model types.

ID	Description	Type
1	Probability of being a person	PVO
2	Probability of being a vehicle	PVO
3	Probability of vehicle driving slow	Event
4	Probability of vehicle driving fast	Event
5	Probability of vehicle starting	Event
6	Probability of vehicle stopping	Event
7	Probability of vehicle turning	Event
8	Probability of person walking	Event
9	Doorway likelihood map	Normalcy
10	Parking-spot likelihood map	Normalcy

All of the descriptors are whitened using FAST ICA [15]. That is, eigenvalue decomposition is used to remove correlation and to enforce a variance of one. This whitening creates a descriptor space that is better conditioned for optimization during hierarchical divisive clustering. Note, any event detector, PVO classifier, or normalcy model generator can be used as descriptors here.

4. Pyramid Coding

The pyramid coding process first forms the sparse-dense pyramid of codebooks and then encodes the scene into a pyramid of functional region maps. This process starts with a set of descriptors that are derived from all the track's detections, $X = [x_1, \dots, x_N]^T \in \mathbb{R}^{N\times D}$, where N is the number of detections and D the number of descriptors. The GMM based hierarchical divisive clustering algorithm starts at layer two, k=*2*, by bifurcating X into two clusters using an Expectation Maximization (EM) algorithm, which is initialized with two points farthest from each other in the $L^2 - norm$ sense. The two new dense clusters maximize the following likelihood:

$$max_\lambda P(X|\lambda) = \prod_{n=1}^{N} \sum_{m=1}^{2} \pi_m \, \mathcal{N}(x_n|\mu_m, \Sigma_m), \qquad (1)$$

where $\lambda = \{\pi, \mu, \Sigma\}$, π is the prior distribution, and $\mathcal{N}(\cdot)$ is their normal distribution with mean vector μ and covariance Σ. The N data points are then assigned to their most likely cluster. The next layer is created by bifurcating the cluster in layer two that has the largest variance (determinant of the cluster's covariance) and applying the GMM EM algorithm to only its data points. This process is repeated at each layer until the maximum number of clusters is reached, or until the model fit to the data vs.

complexity no longer improves, as determined using the Bayesian Information Criterion [8].

One significant benefit of this approach is that the data points in *X* have a clear path through the pyramid, where the sum of the points in the child clusters equals the number in their parent cluster. This results in only two unique clusters at each layer, which reduces the model complexity and creates our sparse pyramid.

The k^{th} layer in the pyramid initially results in a full codebook with k clusters (codewords), which are used to encode the scene. This is accomplished by overlaying an $I \times J$ grid onto the ground plane and assigning each grid cell the most frequently occurring codeword. More specifically, each of the data points in grid location *(i,j)* are assigned to one of the k codewords. Grid cell *(i,j)* is then assigned the codeword label, $m = \{1, \dots k\}$, that occurs most frequently for layer *k*.

$$f_k(i,j) = \operatorname{argmax}_m \{|v_{k,m}(i,j)|\}, \qquad (2)$$

where $|v_{k,m}(i,j)|$ denotes the number of times that codeword *m* occurs in grid cell *(i,j)* for layer *k*. Since the codewords represent groups of common track descriptors, they characterize different types of behaviors. Therefore, $f_k(i,j)$ captures the normal behavior that occurs in grid cell *(i,j)*, where f_k is the "functional region map". The final result of the encoding process is a pyramid of functional region maps, Figure 2 [3]. After the functional region maps are created only the two unique clusters per layer are kept, which reduces the full pyramid from 595 clusters to 68 for the sparse-dense pyramid when *K=35*.

5. Modeling Local Behavioral Context

Local behavioral context is modeled by performing the pooling step on the training example's ROI from each layer in the pyramid of functional region maps. Average-pooling accumulates the number of times that the two unique clusters (mixture of behaviors) occur in each layer into a histogram model, H_{ex}, for training example *ex* and then normalizes by the size of the region, Equation 3. By design, the two new clusters always have indices *b=1* and *b*=2, resulting in the following histogram model:

$$H_{ex}(b') = \frac{1}{area(R)} |v_{k,b}(R)|, \qquad b'=2k-2+b \qquad (3)$$

where $b = \{1,2\} \forall k$ and $|v_{k,b}(R)|$ is the number of times that codeword $v_b$occurs in region R of f_k.

To improve processing speeds during recognition the codeword counts, $H_{ex}(b')$, are stored as integral images, $\Upsilon_b{}' \in \mathbb{R}^{I \times J}$, [7] during training. Compared to a brute force search and count during testing this reduces the per example processing for a 3024×2304 AOI from 25 minutes to 32 seconds using Matlab on a laptop with quad core I7 processor and 8GBs of memory.

6. Functional Recognition

The recognition process starts by cycling through each training examples, *ex*, from the scene element type of interest, $l = \{1, .., L\}$. For each training example the scene is raster scanned using a test window whose center *(i,j)* is the point being evaluated based on its local context. The local context during testing is captured by pooling the codewords from region *R*, defined by the current training example, into the test histogram, H_{test}. The unique codeword counts, $H_{test}(b')$, are easily extracted from the integral images using *R's* upper-left corner point (i^{ul}, j^{ul}) and lower-right corner point (i^{lr}, j^{lr}):

$$H_{test}(b') = \Upsilon_{b'}(i^{lr}, j^{lr}) - \Upsilon_{b'}(i^{ul}-1, j^{lr}) - \Upsilon_{b'}(i^{lr}, j^{ul}-1) + \Upsilon_{b'}(i^{ul}-1, j^{ul}-1) \qquad (4)$$

The Laplace kernel shown in Equation (5) is used for matching the histogram from training example *ex* and the test window at every *(i,j)* grid cell location.

$$\phi_{ex,l}(i,j) = e^{\left(-\frac{\|H_{test}- H_{ex,l}\|}{\sigma}\right)} \qquad (5)$$

This results in a 2D likelihood map for functional scene element type *l*, where higher likelihood values indicate a better match to the model. Averaging over all *E* likelihood maps for class *l* produces the mean likelihood map:

$$\Phi_l(i,j) = \frac{1}{E} \sum_{1 \le ex \le E} \phi_{ex,L}(i,j) \qquad (6)$$

The mean likelihood map, $\Phi_l(i,j)$, is then discriminatively normalized by calculating two CDFs, one from the positive training examples, $P(Y \le \mathrm{y})$, and one from all negative training examples, $P(Z \le \mathrm{z})$. The values for the CDFs are extracted from their corresponding manually annotated bounding polygons. The normalized likelihood map is then their joint probability, with an independence assumption:

$$\Phi_l' = P(y,z) = P(Y \le y) P(Z \le z) \qquad (7)$$

Intuitively, this normalization is a probabilistic interpretation of contrast enhancement [13] that enhances regions associated with the positive distribution. Other normalization approaches such as histogram equalization, min-max, z-score, and tanh where initially used, but they are either not discriminative or require significant parameter tuning. A Bayesian classifier was also used, but had no significant improvement.

7. MRF Scene Element Decoding

The likelihood maps produced by Equation (7) have been assumed to be independent up to this point which can lead to spatial inconsistencies when labels are assigned to the scene. To overcome this we use an MRF, as in [4], to perform spatial smoothing that enforces class adjacency constraints. These constraints enforce the likelihood that two scene element types occur next to each other spatially.

This smoothing problem is formulated as 2D lattice MAP-MRF inference with max-product belief propagation [16] and pair-wise potentials. The optimization problem finds the labels that maximize the joint probability of the set of discrete hidden label nodes, $\{h\} \in \mathbb{R}^{I \times J}$, that have L states and data compatibility matrix $\Phi' \in \mathbb{R}^{I \times J \times L}$. Notice Φ' is the normalized likelihood maps from Equation (7), which are derived from the track descriptors and the observations, O. For the purpose of analyzing the joint probably these are converted to 1D arrays with M elements, where $M = I \times J$, $\{h\} \in \mathbb{R}^{M}$, and $\Phi' \in \mathbb{R}^{M \times L}$. The joint probability of the hidden nodes (labels) and the observations is then:

$$P(\{h\}, O) = \frac{1}{\mathrm{Z}} \prod_{(r,s) \in \mathcal{N}} \psi(h_r, h_s) \prod_r \Phi'(h_r) \qquad (8)$$

where Z is a normalization factor, $s \in \mathbb{R}^{M}$, and $\mathcal{N}$ is the neighboring four nodes.

The class constraints, $\psi(h_r, h_s)$ in Equation (8), are imposed using an adjacency graph, Figure 3. The edges in the graph represent classes that are more likely to be spatially adjacent to each other. More specifically, $\psi(h_r, h_s) = \alpha$ where there is an edge, one for self-adjacency, and β otherwise, where $\alpha > \beta$.

Automatically defining the class constraints requires extensive training data and can lead to unrecoverable errors. Fortunately, the edges here are very intuitive, easily defined and if needed easily changed by the user. Because of this, we manually define the adjacency graph, where $\alpha = 0.8$ and $\beta = 0.5$ for the WAMI data.

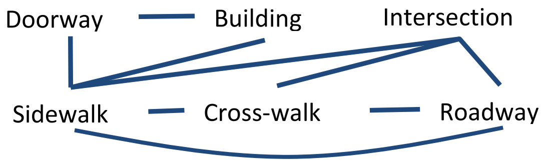


Figure 3, Adjacency matrix for WAMI scene elements.

The output of the MRF inference process produces the decision labels for each grid cell along with updated belief probability maps, which are used for evaluation.

8. Experiment Approaches and Results

Experiments are performed on two datasets that cover the high altitude aerial domain as well as the ground surveillance domain of outdoor scenes. The ARGUS-IS collected WAMI data is captured from a moving aerial platform while the Ocean City data from [1,2,3] is from a stationary webcam. Results from our approach are compared against three similar state-of-the-art functional recognition approaches [2,3,4]. The performance is determined by using several metrics: Mean Average Precision (mAP), Precision-Recall (PR) curves, and Probability of Correct Classification (PCC). The PCC is the number of correctly classified examples divided by the number of total examples.

Our evaluation focuses on how well the entire manually annotated functional scene element regions are detected. Therefore, all the grid cells within the bounds of each of the test examples are evaluated. The evaluation of our pyramid coding approach is carried out using a two-fold cross-validation approach. In order to ensure a fair comparison all approaches use the same track-based descriptor set from Table 1.

8.1. Data

Experiments on the ARGUS-IS collected WAMI data use the 3024 × 2304 AOI shown in Figure 1 that represents 21 minutes of video at ~3.33Hz. This data generated 3.9k fragmented tracks on both vehicles and pedestrians. The Ocean City (OC) web-cam data [1,2,3] is 704 × 480 at ~2Hz with significant perspective changes between near-field and far-field. There are over 10k highly fragmented tracks from 8 hours of video with many false tracks caused by camera artifacts and lighting changes.

The WAMI data includes numerous examples of various functional scene elements that have varying levels of shape, appearance, and behaviors, particularly within the same class. On the other hand, the OC data is characteristic of the video in [4] in that there are only a few examples of functional scene elements that are also very similar in both appearance and behaviors. Unfortunately, the data used in [4] is not available for comparison and the CAVIAR dataset in [2] only has two to three scene element types with few examples of each.

8.2. Comparison Approaches

Our approach is compared against the “Direct-Clustering”, “Functional-Category”, and “Supervised-MRF” approaches from [2], [3], and [4], respectively. Additional comparisons are also made against two discriminative variations of our approach and the HKM and KM coding approaches.

The Direct-Clustering approach uses descriptors from scene element specific normalcy models as an input to a hierarchical divisive clustering algorithm. This approach relies on a flat codebook using the leaf clusters and does not take into account local behavioral context. Our comparison against this approach is to demonstrate the performance impact of the pyramid of codebooks. Therefore, we limited our approach to just the leaf clusters and show the overall improvement.

The unsupervised Functional-Category approach [3] clusters the descriptors into a codebook and then clusters the histograms into functional categories using mean-shift clustering for both. This approach has the benefit of not needing to define training examples or scene element detectors [2,4]. However, we found that it is very sensitive to the grid cell size and mean-shift parameter, since it clusters on descriptor statistics derived from the grid cells.

The approach in [4] uses manually defined functional scene element models based on binary descriptors combined with an MRF for smoothing. To implement this approach the continuous descriptors from Table 1 are converted to binary descriptors by thresholding the average probability of a descriptor in a particular grid cell. When the average probability is greater than 0.5 it is set to +1 and zeros otherwise. The manually defined functional scene element models then encode the combination of descriptors that characterize the scene element of interest. In particular, for the scene element of interest a +1 is used for descriptors that should occur, 0 when it is irrelevant, and a -1 when it should not occur. The grid cell's binary features are compared against the manually defined models using the hamming distance, which are normalized to form the data compatibility matrix used in the MRF.

8.3. ARGUS-IS WAMI Results

The WAMI AOI initially has 254 annotated functional scene elements over six classes of interest: Building (79), Intersection (22), Cross-Walk (25), Roadway (38), Sidewalk (76), and Doorway (14). However, in order to recognize scene elements based on behaviors there need to be moving objects in and/or around them. Therefore, we have limited our analysis to the scene element examples with more than 5% of the overall activity for the class of interest, reducing the number of examples to: Building (25), Intersection (17), Cross-Walk (18), Roadway (23), Sidewalk (50), and Doorway (11). Figure 4(a) shows all of these annotations as color coded polygons. Figure 4(b) shows the belief probability for the Intersection class as a heat map, where darker red indicates more likely regions and darker blue less likely. The belief probability maps provide more insight into the detection capabilities of the algorithms. That is, one can visually see that a region can have nonzero probabilities for the class of interest. On the other hand, the decision labels will only show the most probable class for that region, Figure 5.

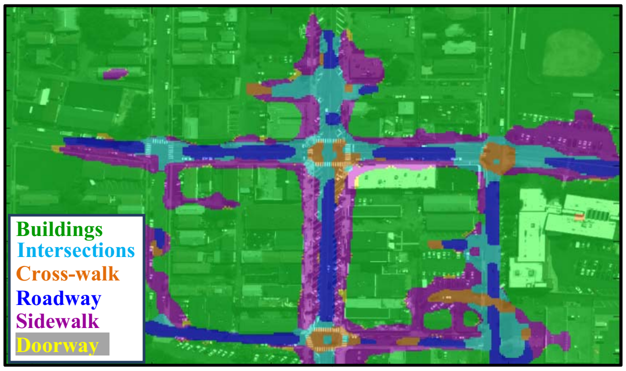


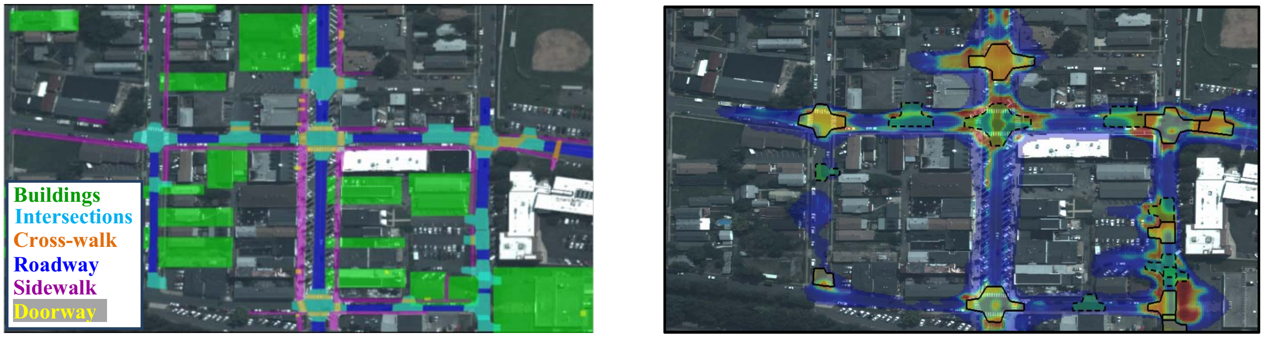
Figure 5, Pyramid Coding recognition results on the WAMI data (344 × 256 grid).

Our pyramid coding approach uses an arbitrarily large number of maximum clusters (35), relative to the number of classes (6). This results in 68 unique clusters in the sparse-dense pyramid. The spread parameter in the Laplace kernel, Equation (5), is set to one, $\sigma = 1.0$, and 50 iterations of belief-propagation inference are used.

Figure 5 shows the decision labels made by the MRF for our approach. Notice, the entire scene has label assignments this is because our approach uses the local behavioral context to form the data compatibility observations instead of single grid cell statistics.

The PCCs and mAP metrics are generated using the MRF labels. Our pyramid coding approach has overall PCC and mAP values of 68.0% and 0.72, respectively. As a comparison to the flat codebook used in the Direct-Clustering approach in [2] we reran our algorithm, but only used the leaf clusters in the codebook, with all other aspects being equal. The overall PCC in this case is 51.1% and the mAP is 0.55 indicating that pyramid coding reduces the error by 34.6% and 37.8%, respectively.

Figure 6 shows the PR curves for the six scene elements and an overall curve, which is color-coded according to the legend in Table 2. The lower PR curve for the Cross-walk class is due to confusion with the Intersection class. The poor Doorway detections are due to the small number of enter/exit examples (1-12 instances). Table 2 shows the



(a) Truth Scene Elements

(b) Belief heat map for Intersection (344 × 256 grid)

Figure 4, (a) Truth functional scene elements overlaid on WAMI background image. Legend in lower left shows the color coding. (b) Belief heat maps showing the probability of spatial regions for the Intersection scene element. Training examples are overlaid as yellow polygons with black solid outlines, while the test examples are shaded green with black dashed outlines.

PCCs and mAP values per scene element.

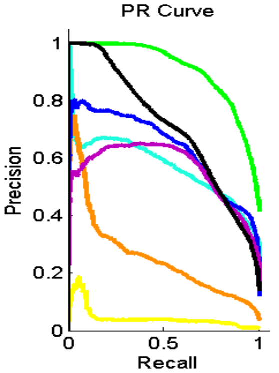


Figure 6, PR curves for the pyramid coding approach.

The Functional-Category approach required a resolution of 197 × 148 to achieve reasonable results, which is much coarser than the 344 × 256 grid used for the other two approaches. This is because the Functional-Category approach needs larger grid cells to accumulate statistics per grid cell. As in [3] we used a mean shift clustering algorithm for implementation with bandwidths of 2.3 for the descriptors and of 0.25 for clustering the histograms. Figure 7(a) shows the final clusters (Functional Categories) as they are assigned to grid cells. These results are quantified in Table 2 by manually assigning the clusters to the most likely scene element type. The lower performance is due to the algorithm's sensitivity to the grid cell size, mean shift parameter, and because it does not model local context.

The supervised MRF based approach [4] was implemented using the same MRF adjacency constraints as defined in Section 7. The Roadway and Sidewalk are detected well, as seen in Figure 7(b) and by the PCCs in Table 2, while the others appear to have very low performance. We believe the lower overall performance here is also due to the fact that the data compatibility matrix does not capture the local behavioral context and because the statistics are calculated from grid cells.

Experiments with discriminative approaches were also conducted. Using a Support Vector Machine with a nonlinear Laplace kernel (mAP=0.68) or discriminatively down-selecting the histogram bins using Adaboost feature selection (mAP=0.67) lead to slightly lower performance. Pyramid coding experiments were also performed that replaced our coding approach with the KM or HKM approaches. Our pyramid coding approach has a 24.3% and 51.1% reduction in mAP error when compared to the KM and HKM, respectively. The lower performance is because both approaches rely entirely or heavily on the leaf nodes. That is, the former approach assumes that you know the number of clusters and the latter assumes that the relevant information for classification is in the sparse and under-represented leaf clusters, when it is the denser high content clusters that are more informative here.

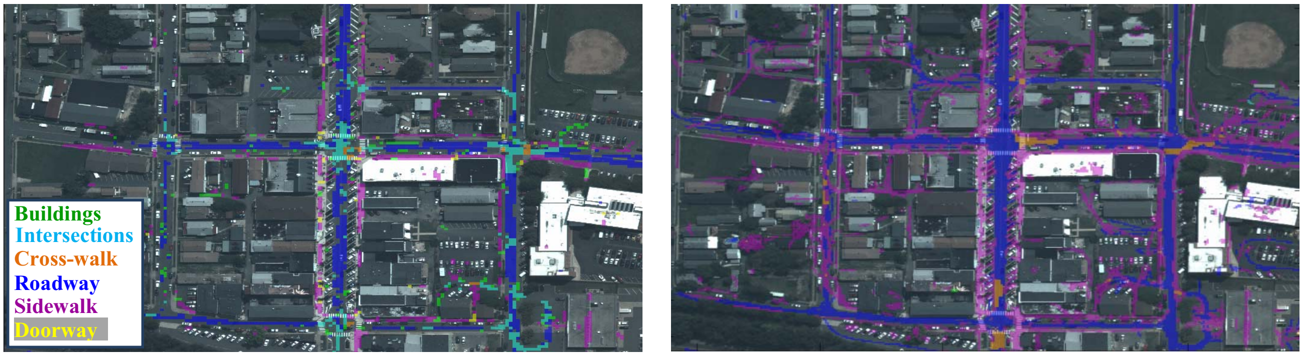
Table 2, mAPs and PCCs on the WAMI data

Legend	Functional Category [3]		Supervised MRF[4]		Pyramid Coding	
FE Names	mAP / PCC%		mAP / PCC%		mAP / PCC%	
Building	NA	0.6	0.06	0.02	0.92	79.6
Intersection	NA	18.3	0.50	0	0.58	52.2
Cross-walk	NA	4.5	0.15	18.7	0.21	32.4
Roadway	NA	50.3	0.42	66.1	0.63	44.3
Sidewalk	NA	34.9	0.46	66.0	0.53	65.0
Doorway	NA	5.9	0.01	0	0.03	0
Overall	NA	9.0	0.19	16.8	0.72	68.0

8.4. Ocean City Webcam Results

The decoded OC scene results are shown in Figure 8 for the Functional-Category [3], Supervised-MRF [4], and our pyramid coding approach along with the truth assignments. Five scene elements are analyzed here: Building, Roadway, Sidewalk, Doorway, and Parking-spot, with 5, 13, 21, 5, and 24 examples respectively. This dataset uses features 1, 2, 9, and 10 from Table 1. The event descriptors are not used because the very high levels of track fragmentation and large number of false tracks make them unreliable. Table 3 shows the PCC and mAP scores for all three approaches.

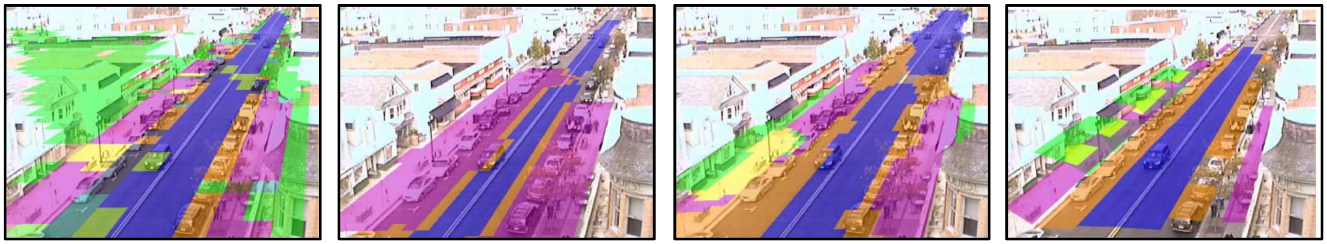
The results shown in Figure 8(a) for the Functional-Category approach are slightly better than those shown in [3], particularly for the Doorway class. This is mostly because of the use of the doorway normalcy maps as descriptors. Also notice that this approach appears to detect the Building class. While this is the correct label, it is for the wrong reason. That is, the buildings are detected because of the many short false tracks that are clustered together and not because of pedestrian or vehicle behaviors. These results were obtained using mean shift bandwidth parameters of 1.5 for the descriptors and of 0.25 for clustering the histograms with a 49 × 80 grid.



(a) Functional-Category [3]

(b) Supervised-MRF [4]

Figure 7, Functional scene element labels assigned to grid cells (a) Functional-Category approach (197 × 148 grid) (b) Supervised-MRF approach (344 × 256 grid).



(a) Functional-Category [3] (49 × 80) (b) Supervised-MRF [4] (121 × 199) (c) Pyramid-Coding (121 × 199) (d) Truth

Figure 8, Qualitative results on OC dataset for the three approaches using five scene element types. Legend in Table 3 shows color code.

Table 3, mAPs and PCCs on the OC data

Legend	Functional Category [3]		Supervised MRF[4]		Pyramid Coding	
FE Names	mAP / PCC%		mAP / PCC%		mAP / PCC%	
Building	NA	54.0	0.24	0.3	0.81	63.6
Roadway	NA	75.8	0.96	50.6	0.95	81.4
Sidewalk	NA	80.7	0.64	1	0.68	52.9
Doorway	NA	21.4	0.19	0	0.39	18.7
Parking Spot	NA	56.1	0.27	0.2	0.76	73.4
Overall	NA	54.0	0.50	33.0	0.81	63.6

The Supervised-MRF approach performs very well on the Sidewalk class, but at the expense of the Doorway and Building detections. A class adjacency matrix similar to Figure 3 was used here with $\alpha = 0.95$ and $\beta = 0.25$.

Overall our pyramid-coding method shows significant improvements compared to the other two approaches, particularly on the Parking-spot, Roadway, and Building classes. Our higher mAP performance on the Building class is due to the incorporation of local behavioral context, which enables it to detect not just the building but the activity regions associated with them.

9. Conclusion

We presented two contributions to current state-of-the-art functional scene element recognition approaches that lead to significant recognition improvements. The first is the incorporation of sparse-dense pyramid of codebooks to better characterize multi-model scene elements. The second is the incorporation of local behavioral context through pooling behaviors in and around scene elements. This enables the detection of elements with low and/or indirect evidence. Comparisons are made to the most relevant functional scene element recognition and coding approaches [2,3,4,14] on ground surveillance video and for the first time on WAMI data. Significant improvement in recognition rates were shown for both datasets.

10. Acknowledgments/Disclaimer

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under contract no. W91CRB-10-C-0098. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressly or implied, of DARPA or the U.S. Government. Approved for public release; distribution unlimited.

11. References

[1] E. Swears and A. Hoogs, "Functional Scene Element Recognition for Video Scene Analysis," WMVC, 2009

[2] E. Swears, M. Turek, R. Collins, A. Perera, and A. Hoogs, "Automatic Activity Profile Generation from Detected Functional Regions for Video Scene Analysis," Video Analytics for Business Intelligence, Studies in Computational Intelligence, Springer-Verlag, vol. 409, 241-269, 2012

[3] M. Turek, A. Hoogs, and R. Collins, "Unsupervised Learning of Functional Categories in Video Scenes", ECCV, 2010

[4] C. Fernandez, J. Gonzalez, and X. Roca, "Automatic Learning of Background Semantics in Generic Surveilled Scenes," ECCV, 2010

[5] A. Kembhavi, T. Yeh, L. Davis, "Why Did the Person Cross the Road (There)? Scene Understanding Using Probabilistic Logic Models and Common Sense Reasoning," ECCV, 2010

[6] K. Grauman and T. Darrell, "The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features," ICCV, 2005

[7] O. Veksler, "Fast Variable Window for Stereo Correspondence using Integral Images," CVPR, 2003

[8] G. Schwarz, "Estimating the Dimension of a Model," Annals of Statistics, 6(2), 461-464, 1978.

[9] C. Stauffer, W. Grimson, "Adaptive Background Mixture Models for real-time tracking," CVPR, 1999

[10] A. Perera, C. Srinivas, A. Hoogs, G. Brooksby, and W. Hu, "Multi-object tracking through simultaneous long occlusions and split-merge conditions," CVPR, 2006

[11] A. Guenoche, P. Hansen, and B. Jaumard, "Efficient algorithms for divisive hierarchical clustering with diameter criterion," Journal of Classification, 8(1):05-30, 1991

[12] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond Bag of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories," CVPR, 2006

[13] M. Jaur, J. Jaur, J. Kaur, "Survey of Contrast Enhancment Techniques based on Histogram Equalization," IJACSA, vol. 2, no. 7, 2011

[14] D. Nister and H. Stewenius," Scalable Recognition with a Vocabulary Tree," CVPR, 2006

[15] A. Hyvärinen and E. Oja, "Independent Component Analysis: Algorithms and Applications," Neural Networks, 13(4-5),411-430, 2000

[16] P. Felzenszwalb and D. Huttenlocher, "Efficient belief propagation for early vision," IJCV, 70, 2006, 41–54