Vol.2014-CVIM-192 No.31
2014/5/16

Robust Object Detection by Voting in Multiple Feature Spaces

Zhipeng Wang, Matasaka Kagesawa, Shintaro Ono, and Katsushi Ikeuchi

Institute of Industrial Science, The University of Tokyo, Japan
Email: {wangzp,kagesawa,onoshin,ki}@cvl.iis.u-tokyo.ac.jp

In this work, the performance of visual detection methods is improved from an aspect of combining information from different channels. The efforts are two-fold: 1) combining motion information with appearance information, and 2) combining visual and spatial information encoded among the local image features of the same object. Three detection methods are proposed, and the most important component is a voting system in each method. The first detection method is developed for real-time applications. By making time-consuming steps deal with fewer instances, the method combines motion information with appearance information efficiently, and gives promising results in real time. The second method extends the Implicit Shape Model to incorporate motion information, and outperforms the state-of-the-art method on two datasets. The third method does pyramid matching during training and detection for efficiency, makes full use of the visual and spatial information of local image features, and gives robust detection results efficiently.

Keywords: *Object detection, Voting*

1 Introduction

Visually detecting objects of interest from a complex scene is a basic perceptual skill in human beings and other animals. And successful object detection methods play fundamental roles in many application areas, which include video surveillance, driving assistance, image retrieval, etc.

Most modern detection methods fall into two categories. Some follow the sliding-window schema, and detect objects by consider whether each of the sub-images contains an instance of the target object. The other methods infer object centers based on local image features in a bottom-up manner. These methods start with detection of object parts, and then make inferences about the target objects' states, like position, or label. In this paper, efforts are also made to improve performance of detection methods. These efforts try to explore how to use the information which previous methods do not make full use of. Roughly, the efforts belong to two categories, the first category is exploring approaches of efficiently and effectively combing of motion information with appearance information, and the second category is exploring how to combine visual and spatial information encoded in local image features of the same object.

For fusion of information from different channels, voting systems are employed. Voting is preferred for its robustness in using local information, and its inference procedure's capability to use global information. Many methods for detection mainly use either appearance information or motion information. Following some previous work [11], the first two methods will address that when well combining the information from these two channels, detection performance will be better.

The first method is developed mainly for real-time applications under limited computational power. This method can be considered as a three-step method. The first step deals with keypoints. It takes original data as input, and outputs keypoint clusters as detection hypotheses. This step detects, verifies, and clusters keypoints. The second step takes these keypoint clusters as input, verifies them by their appearance and motion information, and outputs the ones which pass verifications as detection results. The last step feeds the detection results from step two into a voting system. Since detection results are connected by their belonging trajectories, voting along the temporal dimension is responsible for giving the final decision of each object, when it disappears from the scene. Motion information plays a very important role in the method. The target objects are considered as possessing both particular appearance patterns and motion patterns. When the second step verifies the detection hypotheses using appearance information, a biased classifier is used. This classifier produces more false alarms to pursue higher detection rate. Then motion information is used to filter out the false alarms. Motion information in the form of trajectories also connects weak inferences and feeds the weak inferences into a voting system for the final results. In addition, the pipeline of this method is optimized in a hierarchical way. In the pipeline, the later one step is, the more time-consuming it is, and the fewer instances it will deal with. The method performs well under simple scene, i.e., data collected by infrared cameras in a tunnel environment, and gives promising detection results in the experiments. However, the performance of this method under complicated scene is not promising. And then we propose the second method.

The second method belongs to methods based on Hough transform. It extend the Implicit Shape Model [16] to combine motion information. For training, image features together with labels and offsets to object centers of sample images are considered as codes, and inserted into a codebook. For detection, image features are detected on the target image, and then matched against the codebook using image feature as key. The matched codes will indicate the labels and object centers. During the detection step, this method firstly do motion analysis, which results in grouping results of the image features on the target image. The grouping results are used during the inference for labels and centers of the target objects. It is assumed that image features with the same motion pattern, here in the same motion group, should belong to the same object. The inference procedure then prefers the label and position inferences with more consistence in the same motion group. On two datasets, the proposed method outperforms the state-of-the-art method.

While the second method performs well under complicated scene, it is relatively slow. This is due to the time-consuming property of methods based on Hough transform. The third method aims at improving the efficiency of the second method. Also it tries to flatten the gap of appearance and positional information. This method does not use motion information. In methods based on Hough transform, image features are used as key to query similar codes from the codebook, and in the third method, both appearance and position are used as key. The bottom-up property of Hough transform also ignore the relationship between different image features. Actually, the mutual information encoded in the image features of the same object is very informational. The third method considers objects as point sets of, i.e., of 12-dimensional, while the first 10 dimensions are appearance information, and the last 2 dimensions are positional information. The training step is almost the same with Hough-transform methods, except for how a few parameters are trained. At the detection step, instead of using the appearance information of one single feature for querying, the point set of a sub-image is used for querying. Pyramid Matching is used for accelerating the querying. The procedure ensures the full use of the visual and spatial information encoded in the image features of the same object. While giving promising detection results on two datasets, this method is confirmed to be much more efficient than the second method.

The paper is organised as follows. 2 reviews related work. 3 introduces the method aimed at efficient detection by combining motion and appearance information. 4 proposes the method that extends the Implicit Shape Model to incorporate motion information, and the method groups object parts for detection. 5 presents the method which detects by Pyramid Match Score. 6 concludes.

2 Related Work

Detection is drawing a lot of attention [4, 6, 8, 13, 18, 22, 23, 26, 3, 5, 7, 15, 16, 17, 19, 20, 21], and it will continue to. While some methods are unique and very heuristic.

Instead of proposing class-specific methods, [2] tries to evaluate how like a sub-image contains an object of any class. In the method objects are defined by very general properties, which include having closed boundary, being different from surroundings, and sometimes being unique and salient in the image. By combining saliency detection, color contrasting, edge detection, and image segmentation methods in a Bayesian framework, they give convincing performance in general-purpose object detection. Bag of image features [10] is an important advance for object detection. Before the method, potential objects are described using feature extracted from raw pixels, while the method describes objects using object components. In the work following, [25] added a biased sampling component for describing each object. Instead of being described by one group of features, the objects are described by several groups of features, and then decisions are made using multi-label multi-class classification. Some pioneer methods also detect or recognize objects in 3D space. The method proposed in [12], tracks keypoints of the same object, generate features which include 3D information of the object accordingly, and feed the features to decision step. And the results are very appealing. Also excellent performance of deep learning inspires new methods to reconsider object representations. The discover of invariants and learning of a detector from unlabeled data are explored by [14].

Three methods are proposed in the paper. The methods contribute to decision making in object detection. These methods try to improve existing voting systems to easily employ information in multiple channels or try to accelerate the decision making procedure by employing good mechanisms. The methods proposed in this paper try to contribute to object detection by proposing effective and efficient mechanisms to combining information from multiple channels in robust voting systems. These efforts are encouraged by the human beings' amazing visual capabilities, these efforts try to act at a high abstraction level, and these efforts belong to the very challenging topic of decision making in object detection.

3 Efficient Voting along Time Axis

The pipeline of the first method is designed for efficiency. The method deals with the large amounts of information contained in one image, following a hierarchical manner. The later a step is, the more time-consuming it is, and the fewer instances it deals with. The advantage of this method is its ability to give promising detection results from cluttered data in real time. In addition, this method successfully combines bottom-up and classification methods, as

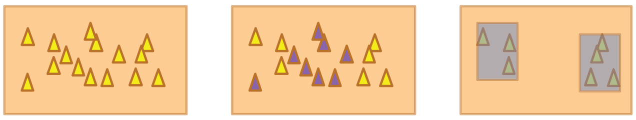


Figure 1: Keypoint detection, verification, and clustering. On the left, keypoints represented in yellow are detected on images. In the middle, yellow represents the keypoints pass verifications, while purple represents the keypoints fail to pass verifications. On the right, the keypoints which pass verifications are clustered, and blue rectangles represent candidate objects.

well as combines both appearance and temporal information.

3.1 Method Outline

The method is a three-step method. The first step of candidate object localization deals with keypoints. It takes original data as input, and outputs keypoint clusters as detection hypotheses. The second step of candidate object verification takes these keypoint clusters as input, verifies them by their appearance and motion information, and labels them as positive or negative. In the third step of voting along time axis, keypoint clusters connected by the same trajectory will vote for whether the trajectory is positive or negative when the trajectory ends, and this gives the final detection results.

3.1.1 Candidate Object Localization

There are a huge amount of information contained in even one frame of a video collected by cameras. In time-critical tasks, not all pixels in each frame are processed, instead keypoints can be detected, and verified. The appearance of keypoints belong to local features. Still the number of keypoints is large, and association of keypoints along time dimension is not feasible due to the constrain of time consumptions. Clustering the keypoints on the frame is feasible and will indicate the locations of target objects. Thus in this step, keypoints are detected from the input images, verified using appearance model, and then clustered to indicate candidate object locations, as shown in Figure 1.

3.1.2 Candidate Object Verification

The number of candidate objects in the form of keypoint clusters is smaller compared with the number of keypoints. Expensive global appearance feature is feasible, and expensive association along time axis is also feasible. By assuming certain motion patterns of the target objects, the trajectories can also be used to verify candidate objects. Thus in this step, candidate objects are verified by appearance and motion, as shown in Figure 2.

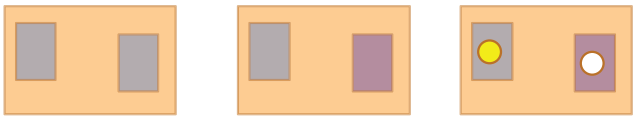
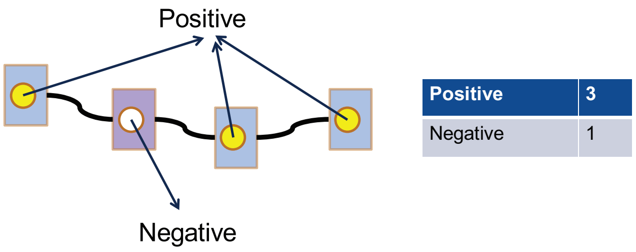


Figure 2: Candidate object verification by appearance and motion. On the left, blue rectangles represent the candidate objects given by the previous steps. In the middle, the blue rectangle represents that the candidate object is decided as positive by appearance, and purple rectangle represents negative by appearance. On the right, yellow circle represents that the candidate object is decided as positive by motion, and white circle represents negative by motion.



Positive	3
Negative	1

Figure 3: Voting along time axis. Local decisions vote along trajectories for positive and negative. In the example, there are 3 positive votes and 1 negative votes.

3.1.3 Voting along Time Axis

Trajectories not only connects the candidate objects, but also connects the local decisions made according to appearance and motion. To refine the results, when each trajectory ends, voting is employed, as shown in Figure 3.

3.2 Implementation on Data Collected by Infrared Cameras in Tunnels

The method is implemented to detect emergency telephone indicators in tunnels. The implementation aims to perform detection in real time, and serve as an effective unit in positioning automobiles in tunnel environment. In a tunnel environment, in addition to emergency telephone indicators, a lot of noisy objects also appear, e.g. ordinary lights, other vehicles, and other vehicles' shadows. And some of the noisy objects cannot even be distinguished from the target objects by appearance , as shown in Figure 4.

3.2.1 Candidate Object Localization

The method employs a simple yet useful method to detect keypoints. Firstly, points are uniformly sampled for an offset of 6 in width, and 7 in height (the length of an emergency telephone indicator is larger than its width). In this manner the magnitude of instances is reduced by nearly two orders. Then points that pass the test, which verifies them by setting intensity thresholds, are considered as keypoints. Here

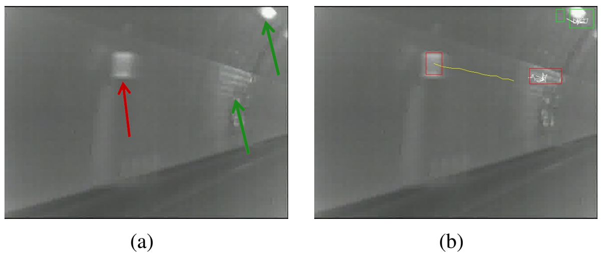


Figure 4: Original data and detection results. In (a), the red arrow points to the target object: emergency telephone indicator, and the green arrows point to noisy objects. In (b), red rectangles mark detection hypotheses labeled as positive using appearance information, and green rectangles mark negative ones. Yellow trajectories mark detection hypotheses labeled as positive using temporal information, and white trajectories mark negative ones.

a Gaussian distribution is assumed for the intensities of the points.

The detected keypoints don't just belong to emergency telephone indicators, but also belong to the background. To verify the keypoints, the appearance of the sub-image around each keypoint is used. Intensity histograms are used to describe the appearance. The k-means method is used to modeling the histograms.

After the verification step, the keypoints are used to build a minimum spanning tree (mst) using the pairwise Euclidean distance between two keypoints. Then the mst is split by cutting edges larger than a threshold. This results in a grouping of the keypoints.

3.2.2 Candidate Object Verification

For each keypoint cluster, the smallest bounding rectangle is considered a detection hypothesis.

The hypotheses are firstly verified by their appearance. An Adaboost machine is trained using intensity histograms. In this step, to emphasize the Adaboost machine's performance on the positive training examples, the initial weights of the positive training examples are set to be 7 times as large as the weights of the negative training examples. Since in practice, whether each keypoint cluster is a target object is decided by both appearance and motion information. The difficulties of excluding noisy objects can be left for later steps.

Not all noisy detection hypotheses can be excluded by using appearance. To futher verify keypoint clusters, the keypoint clusters are tracked through frames to generate trajectories. In this case of keypoint cluster tracking, the problem is relatively simple, since no occlusion occurs. The problem of tracking is modeled by finding the best data association hypothesis between the trajectory set and detection response set.

The temporal information encoded in the trajectories is used to further verify the keypoint clusters. A linear model is used to fit each trajectory, and the Pearson Correlation Coefficient(PCC) of the fitting is the criteria for the decision.

For each keypoint cluster on the current frame, there exists a label given by the Adaboost machine according to its appearance, and the likelihood of fitting its trajectory to a straight line. For each keypoint cluster, it is considered an emergency telephone indicator if and only if its label which is given by the Adaboost machine is positive, its trajectory is long enough, and the likelihood of fitting its trajectory to a straight line is large enough.

3.2.3 Voting along Time Axis

Each trajectory not only connects the detection responses, but also connects the decisions for detection responses made by their appearance and motion patterns. The target objects and noisy objects actually appear in successive frames, and even if we make a wrong decision on one frame, we can expect to recover from this mistake based on the results of other frames. The final results are based on the trajectories of decisions. When one trajectory ends, if more than 80% of the decisions it connects are positive, then this trajectory is considered positive.

The procedure is as follows: 1) each detection result along a trajectory which encodes local appearance and motion patterns votes for whether the trajectory is positive or not, and 2) if the voting percentage is larger than a threshold, a final decision is made that the object is positive.

3.3 Experimental Results

The method is tested based on detection performance and efficiency. Two experiments and their results are reported.

3.3.1 Dataset One

To collect data, infrared cameras are mounted on top of the experimental vehicle, and then we take several tours of the Awagatake tunnel. All models are trained using data from one tour, while evaluated on data from another tour. Firstly, all emergency telephone indicators are marked in the form of rectangles on all frames from the training tour. On a laptop with Intel Core2 Duo 2.8GHz processors, the method deals with real data at a frame rate of 34 frames per second, and this fulfills real-time requirements. The detection rate and false alarm rate is evaluated on 250 frames, as shown in Table 1.

Total number	113
Correctly labeled	102
Miss detections	11
False alarms	21
Detection rate	90%
False alarm rate	19%

Table 1: Detection rate and false alarm rate.

3.3.2 Dataset Two

The results of the first experiment is not satisfactory, and then the second experiment is carried out. A better far infrared camera is used, and the zoom of the camera is adjusted for better images. Then with the new camera mounted on top, the experimental vehicle took several tours of the Awagatake tunnel. Being the same with experiment one, all models are trained using data from one tour, and evaluated on data on another tour. The detection rate and false alarm rate are evaluated on the keypoint clusters, as shown in Table 2.

Total number	472
Correctly labeled	468
Miss detections	4
False alarms	22
Detection rate	99.2%
False alarm rate	4.4%

Table 2: Detection rate and false alarm rate.

The results on the trajectories of decisions are also evaluated. When one trajectory ends, if its length is larger than 15, and over 80% of the last 15 decisions it connects are positive, it is considered as positive. The method correctly detects all the 22 emergency telephone indicators with no false alarms. The detection rate is 100%, and the false alarm rate is 0%.

3.4 Section Conclusion

This section proposes an object detection method, which performs well in simple scenarios by combining appearance and motion information in a very efficient way. The method makes use of appearance and motion information of the target objects in a hierarchical manner. With careful optimization of detection pipeline, the method gives promising results in real time.

4 Common Fate Hough Voting

The second method is inspired by the common fate principle, which is a mechanism of visual perception in human beings, and which states tokens moving or functioning in a similar manner tend to be perceived as one unit. The method embeds the principle in an Implicit Shape Model (ISM). In the method, keypoint-based object parts are firstly detected and then grouped by their motion patterns. Based on the grouping results, when the object parts vote for object centers and labels, each vote belonging to the same object part is assigned a weight according to its consistency with the votes of other object parts in the same motion group. Afterwards, the peaks, which correspond to detection hypotheses on the Hough image formed by summing up all weighted votes, become easier to find. Thus our method performs better in both position and label estimations.

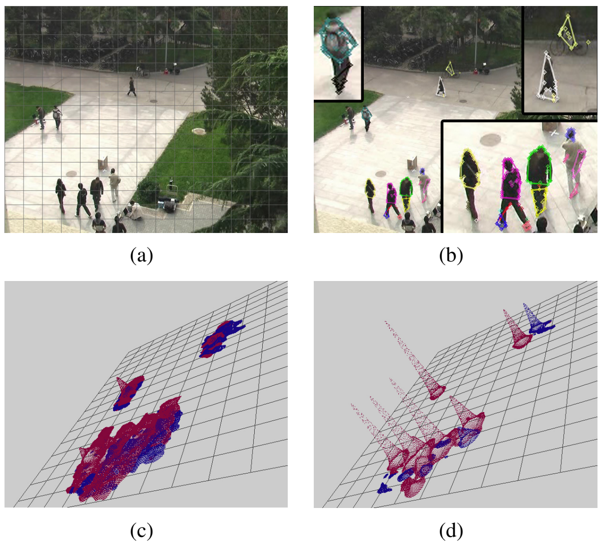


Figure 5: Merit of the proposed method. (a) Original image. (b) Motion grouping results. Some parts are enlarged to show details. (c) Original Hough image. (d) Hough image formed using this method. The grids in (c) and (d) correspond to the grids in(a).

Experiments show the effectiveness of the method in terms of detection accuracy.

4.1 Common Fate Hough Transform

A Hough transform can be simply considered as the transformation from a set of object parts, $\{\mathbf{e}\}$, to a confidence space of object hypotheses, $C(\mathbf{x}, l)$. Where $\mathbf{x}$ is the coordinate of the object center, and l the label. Let $\mathbf{e}$ denote an object part observed on the current image. The appearance of $\mathbf{e}$ is matched against the codebook, and $\mathbf{e}$ activates N best matched codes from the trained codebook. Each code contains the appearance, its offset to the object center, and the class label. According to the N matched codes, $\mathbf{e}$ casts N votes. Each vote $V_{\mathbf{e}}$ is about the object center that generates $\mathbf{e}$. The position of the object center casted by a vote, V, is denoted by $\mathbf{x}_V$, while the class label is l_V. Based on the N votes of $\mathbf{e}$, the confidence that a position $\tilde{\mathbf{x}}$ is the center of an object with class label $\tilde{l}$ is given by,

$$C(\tilde{\mathbf{x}}, \tilde{l}; \mathbf{e}) = \sum_{i=1}^{N} B(\tilde{\mathbf{x}}, \tilde{l}; V_{\mathbf{e}}^i) w(V_{\mathbf{e}}^i) \,. \tag{1}$$

Here $B(\tilde{\mathbf{x}}, \tilde{l}; V_{\mathbf{e}}^i)$ is the blurring function. And $w(V_{\mathbf{e}}^i)$ is the weight of $V_{\mathbf{e}}^i$.

The idea of the proposed method is that, the weight term, $w(V_{\mathbf{e}}^i)$, is defined by the motion grouping results of all the object parts. The blurring function is defined as,

$$B(\tilde{\mathbf{x}},\tilde{l};V)=\begin{cases}0 & \text{if } l_V\neq\tilde{l} \text{ or } |\tilde{\mathbf{x}}-\mathbf{x}_V|>d \\ G(\tilde{\mathbf{x}};\mathbf{x}_V,\sigma) & \text{otherwise}\end{cases}. \quad (2)$$

Here $G(\tilde{\mathbf{x}};\mathbf{x}_V,\sigma)$ is a Gaussian function that fixes the spatial gap between $\tilde{\mathbf{x}}$ and $\mathbf{x_V}$. Let M be the total number of object parts on the image, then by summing up over all the object parts, the confidence of $\tilde{\mathbf{x}}$ being the center of an $\tilde{l}$-class object is given by,

$$\begin{aligned}C(\tilde{\mathbf{x}},\tilde{l})&=\sum_{j=1}^{M}C(\tilde{\mathbf{x}},\tilde{l};\mathbf{e}_j)w(\mathbf{e}_j)\\&=\sum_{j=1}^{M}\sum_{i=1}^{N}B(\tilde{\mathbf{x}},\tilde{l};V^i_{\mathbf{e}_j})w(V^i_{\mathbf{e}_j})w(\mathbf{e}_j)\,.\end{aligned} \quad (3)$$

A uniform weight is assumed for each object part, and $w(\mathbf{e}_j)=\frac{1}{M}$. By considering $C(\tilde{\mathbf{x}},\tilde{l})$ as the evaluation score of the Hough space $(\tilde{\mathbf{x}},\tilde{l})$, the task of estimating object centers and labels converts to finding, and then validating, the local maxima of the Hough image.

4.1.1 Common Fate Weights

To meet the challenges of separating near objects, separating similar different-class objects, and using a noisy codebook, different weights are assigned to the votes of each object part by considering the motion grouping results of the object parts. In this subsection, when given some grouping results, how the results are combined into a Hough transform framework is introduced.

Let $\gamma=\{\mathbf{g}\}$ denote the grouping results, where $\mathbf{g}$ is a group of object parts. Assume $\mathbf{e}_m\in\mathbf{g}$ and $\mathbf{e}_n\in\mathbf{g}$. Those votes of $\mathbf{e_m}$ which are more "agreeable" than the votes of the other objects in $\mathbf{g}$ are assigned larger weights. Towards this end, the relationship between the votes of $\mathbf{e_m}$ and the votes of $\mathbf{e_n}$ needs to be given in advance. This relationship is named support. The support from $V_{\mathbf{e}_n}$ to $V_{\mathbf{e}_m}$ is defined based on $V_{\mathbf{e}_n}$ and the confidence that $V_{\mathbf{e}_m}$'s voted center is correct, as,

$$S(V_{\mathbf{e}_n}\rightarrow V_{\mathbf{e}_m})=B(\mathbf{x}_{V_{\mathbf{e}_m}},l_{V_{\mathbf{e}_m}};V_{\mathbf{e}_n})\,,n\neq m\,.$$

Here $B(\mathbf{x}_{V_{\mathbf{e}_m}},l_{V_{\mathbf{e}_m}};V_{\mathbf{e}_n})$ is defined in (2). This measures the coherence of the two votes from different object parts. Then, the support from $\mathbf{e}_n$ to $V_{\mathbf{e}_m}$ is defined based on $\mathbf{e}_n$, and the confidence that $V_{\mathbf{e}_m}$'s voted center is correct, as,

$$\begin{aligned}S(\mathbf{e}_n\rightarrow V_{\mathbf{e}_m})&=C(\mathbf{x}_{V_{\mathbf{e}_m}},l_{V_{\mathbf{e}_m}};\mathbf{e}_n)\\&=\sum_{i=1}^{N}S(V^i_{\mathbf{e}_n}\rightarrow V_{\mathbf{e}_m})w(V^i_{\mathbf{e}_n})\,,n\neq m\,.\end{aligned}$$

And the support from $\mathbf{g}$ to $V_{\mathbf{e}_m}$ is defined by the confidence that $V_{\mathbf{e}_m}$'s voted center is correct based on the votes of all the other object parts excluding its belonging object part in $\mathbf{g}$, as,

$$\begin{aligned}S(\mathbf{g}\rightarrow V_{\mathbf{e}_m})&=\sum_{\mathbf{e}_i\in\mathbf{g}-\{\mathbf{e}_m\}}C(\mathbf{x}_{V_{\mathbf{e}_i}},l_{V_{\mathbf{e}_m}};\mathbf{e}_i)w(\mathbf{e}_i)\\&=\frac{1}{M}\sum_{\mathbf{e}_i\in\mathbf{g}-\{\mathbf{e}_m\}}S(\mathbf{e}_i\rightarrow V_{\mathbf{e}_m})\,.\end{aligned}$$

By assuming all object parts in the same motion group are from the same object, which means motion grouping gives good results, the estimations for center position and class label given by every object part should be consistent with that given by the motion group. Thus for a particular vote of $\mathbf{e}_m$, i.e., $\tilde{V}_{\mathbf{e}_m}$, a weight is assigned to it by considering its consistence with $\mathbf{g}$ and the consistence of $\mathbf{e}_m$'s other votes with $\mathbf{g}$, as:

$$\begin{aligned}w(\tilde{V}_{\mathbf{e}_m})&=\frac{S(\mathbf{g}\rightarrow\tilde{V}_{\mathbf{e}_m})+\frac{\Delta}{N}}{\sum_{i=1}^{N}S(\mathbf{g}\rightarrow V^i_{\mathbf{e}_m})+\Delta}\\&=\frac{\sum_{\mathbf{e}_j\in\mathbf{g}-\{\mathbf{e}_m\}}\sum_{k=1}^{N}S(V^k_{\mathbf{e}_j}\rightarrow\tilde{V}_{\mathbf{e}_m})w(V^k_{\mathbf{e}_j})+\frac{M\Delta}{N}}{\sum_{i=1}^{N}\sum_{\mathbf{e}_j\in\mathbf{g}-\{\mathbf{e}_m\}}\sum_{k=1}^{N}S(V^k_{\mathbf{e}_j}\rightarrow V^i_{\mathbf{e}_m})w(V^k_{\mathbf{e}_j})+M\Delta}\,.\end{aligned} \quad (4)$$

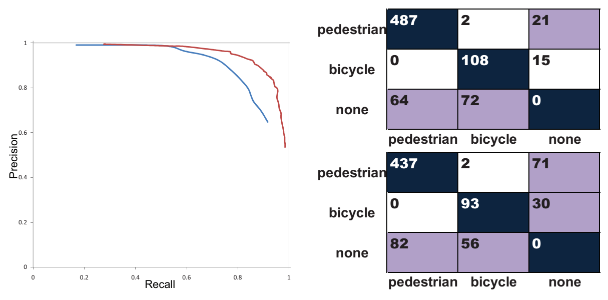
Here, Δ is a small constant for preventing zeros. Notice $w(\tilde{V}_{\mathbf{e}_m})$ is defined using $w(V^k_{\mathbf{e}_j})$ - the weights of the votes of the other object parts in $\mathbf{g}$. In order to determine $w(\tilde{V}_{\mathbf{e}_m})$, uniform weights are firstly assigned to the votes of each object part in $\mathbf{g}$, i.e., $w(V^k_{\mathbf{e}_j})=\frac{1}{N}$. Then new weights are calculated based on the uniformly assigned weights. The weights of votes used to form the Hough image are the iteratively converged weights. The grouping result $\gamma=\{\mathbf{g}\}$, can be replaced by grouping results based on other information, for example our method utilizes motion to group the voting elements.

4.2 Detection

Following [3], the optimal result for the problem is given by greedy maximization. The largest local maximum of all the local maxima is chosen to be the center of a true object, and then the object parts belonging to the chosen object center are excluded from the object part set. A new Hough image, where new objects are found, is formed using the remaining object parts. And this procedure ends when the object part set is empty, or when the confidence of the chosen object is lower than a given threshold.

4.3 Experimental Results

In experiments, advantage of the method is verified in terms of detection accuracy. The method is tested on the P-campus dataset with [3] as a benchmark, and then tested on a dataset of several animals.



	pedestrian	bicycle	none
pedestrian	487	2	21
bicycle	0	108	15
none	64	72	0

	pedestrian	bicycle	none
pedestrian	437	2	71
bicycle	0	93	30
none	82	56	0

(a) (b)

Figure 6: (a) Precision-recall curves (red: the proposed method, blue: the benchmark method). (b) Confusion matrices (upper: the proposed method, down: the benchmark method).

4.3.1 P-campus

For comparison, detection is done on the Hough images formed with and without motion grouping results. The same codebook and the same parameter settings are used for forming and searching over both Hough images. The votes of each object part are assigned uniform weights in the benchmark method, while weights defined in (4) are assigned in the proposed method.

4.3.2 Wild-scene

In order to show that our method can be used for general purposes, we test our method on complicated scenes, especially, complicated background. Even in these cases, our method works well, which shows robustness of our method. A mini dataset is built upon leopards and tigers of the family Felidae. The proposed method successfully localizes and labels all the leopards and tigers, while the benchmark method miss-detects three leopards.

4.4 Section Conclusion

The computational ability of human beings is limited, while the ability to detect is far beyond machines. Thus, it is very possible that this detection ability benefits from multiple perceptual mechanisms. By using one of these mechanisms, we propose a detection method. By embedding motion grouping results into the voting schema of Hough transform, the method is capable to distinguish near objects' positions, to distinguish similar objects' labels, and to maintain detection rate with a noisy codebook. The success of this method further demonstrate the advancement of perceptual mechanisms in human beings. And the success of this method will help with detection methods in ITS areas.

5 Fast Voting by Pyramid Match

The key to the third method is how to define a match score for two point sets. Here pyramid matching procedure is employed, not only for efficiency, but also for combining visual and spatial information from local features in an effective manner. The visual-spatial space is divided from fine to coarse. Under a certain dividing parameter, points from the two matching point sets are considered as match if they fall into the same grid, and they are excluded from the respective point sets. The procedure continues till one point set is empty. Then the numbers of matched pairs under each dividing method is counted, and a weighted sum of all these numbers are considered as the match score for two point set, which will be referred to as Pyramid Match Score, or PMS for short. The weights under all dividing methods are learned during training, and how to divide the visual-spatial space is of great importance.

5.1 Pyramid Match Score

The typical procedure of pyramid matching is firstly reviewed, and how a match score between two point sets by using pyramid matching is defined. Then based on the defined matric, how from the training examples, a super template can be learnt and how the super template can be used for object detection in a test image is proposed. In the definition of the matching score, there are parameters very important, finally how these parameters are estimated is introduced.

5.1.1 Pyramid Matching

The Pyramid Matching method is designed to find the best one-one match between two point sets in a heuristic manner. Given two point sets, $S_1 = \{u_1, u_2, ..., u_m\}, u_i \in R^d$ and $S_2 = \{v_1, v_2, ..., v_n\}, v_i \in R^d$, there exists a best one-one matching π^* that minimizes the sum of $L1$ -distances between matched pairs,

$$\pi^* = \arg\min_{\pi} \sum_{u_i \in S_1} ||u_i - v_{\pi(i)}||_1 \ .$$

Here $m \leq n$, and π maps each feature u_i in S_1 to a unique feature $v_{\pi(i)}$ in S_2.

The best matching exists, and can be found by simple brute-force enumeration. Sub-optimal solution can be found by heuristic methods. The Pyramid Matching method is straightforward. Divide the point space from fine to coarse, find pairs of points from different point sets in the same grid under the current dividing parameter, exclude the matched pairs, and continue this procedure until the smaller point set is empty. The Pyramid Matching method is very efficient, and its time complexity is bounded by $O(dmL)$ [9]. Here d is the number of dimensions in each point set, m is size of the smaller point set, and L is number of dividing methods. In the example of Figure 7, L is 3.

In [9], pyramid matching helps to define kernel functions for SVMs. The meaning of pyramid matching is that, it changes how the way to define similarity

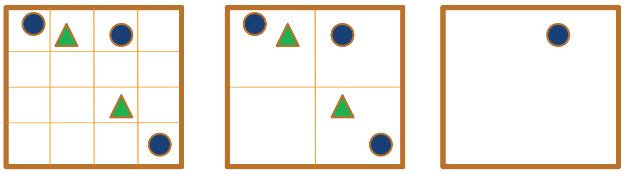


Figure 7: Pyramid matching procedure. The pyramid matching method divides the 2D space from fine to coarse in the 2D space. Notice, the triangle points belong to point set one, and the circle points belong to point set two. In the left, each dimension is divided into 4, results in totally 16 grids, and 0 matched point pairs are found. In the middle, each dimension is divided into 2, results in totally 4 grids, and two pairs of points belonging to different point sets are found and excluded. In the right, since the matched points belonging to matched point pairs are excluded, then only one point from the second point set is left. So the number of pairs found under all dividing methods are, 0, 2, and 0. The pyramid match score is calculated as a weighted sum of these 0, 2, and 0.

between two objects. Originally two objects are considered as similar if they both contain certain number of certain object parts, while the idea of one-one match will only favor the objects parts which have corresponding counterparts.

Let $\gamma = \{\mathbf{g}_1, \mathbf{g}_2, ..., \mathbf{g}_L\}$ be an ordered set, which contains all the dividing methods from fine to coarse. Let $N(S_1, S_2; \mathbf{g}_i)$ be the numbers of matched pairs of points under dividing method $\mathbf{g}_i$. Then the pyramid match score between S_1 and S_2 on γ is defined by,

$$\mathrm{P}(S_1, S_2; \gamma) = \frac{\sum_{i=1}^{L} \omega_i \times N(S_1^{(i-1)}, S_2^{(i-1)}; \mathbf{g}_i)}{m}. \quad (5)$$

Here, S_1^i and S_2^i represent the point set after excluding the points which are found match after the ith round matching respectively from $S_1^{(i-1)}$ and $S_2^{(i-1)}$. Actually, $S_1^0 = S_1$, and $S_2^0 = S_2$.

The procedure is as follows, 1) given the original S_1 and S_2, find the point pairs which fall into the same grid in the space defined by $\mathbf{g}_1$, 2) exclude the matched points respectively from S_1 and S_2 to give S_1^1 and S_2^1, and 3) continue until $i = L$ or one point set is empty.

Then how to construct the dividing methods in γ and how to define the corresponding weight, ω_i, for each $\mathbf{g}_i \in \gamma$ are left to be defined. And these also belong to the factors which distinguish the proposed method from [9].

5.1.2 Training and Detection

The Pyramid Match Score is a matric between two point sets. In [9], image features are considered as points, while in the proposed method, each point encodes both appearance and location information of each local feature. Each visual-spatial point is $d-$dimensional, and, the first $(d-2)$ dimensions are SIFT after PCA, while the last 2 dimensions are relative $x-$ and $y-$ coordinates after considering scale and width-height ratio changes.

Let $\mathbf{p}$ be a visual-spatial point in the point set of an image, I, and $F_\mathbf{p}$ be the image feature of $\mathbf{p}$, which is $(d-2)-$dimentional. Let $x_\mathbf{p}$ and $y_\mathbf{p}$ be the $x-$ and $y-$ coordinates of $\mathbf{p}$. Let w_I and h_I be the width and height of I. Then

$$\mathbf{p} = [F_\mathbf{p}^1, F_\mathbf{p}^2, ..., F_\mathbf{p}^{d-2}, \frac{x_\mathbf{p}}{w_I}, \frac{y_\mathbf{p}}{h_I}] \;.$$

Instead of following [9], PMS does not server as kernel functions for SVMs. And, a procedure similar to Hough transform is employed. Each training image is considered as a point set. From all the training images, the method generates a point set as a super template, $S_\mathbf{T}$, following Algorithm 1. This is just a procedure to collect all points from point sets generated from training images into one point set.

Algorithm 1 Template Generation

```
1: $S_\mathbf{T} \leftarrow \emptyset$
2: for $S_{I_{tr}} \in \{S_{I_{tr}}\}$ do
3:     $S_\mathbf{T} \leftarrow S_\mathbf{T} + S_{I_{tr}}$
4: end for
5: return $S_\mathbf{T}$
```

In Algorithm 1, each $S_{I_{tr}}$ in $\{S_{I_{tr}}\}$ is the point set generated from the corresponding training image I_{tr}, and the $+$ operator is defined on two sets. Actually, $S_\mathbf{T}$ plays a role similar to a codebook as in methods based on Hough transform.

For detection, a most popular pipeline is employed, as in Algorithm 2. All possible hypotheses are generated, given by $\{\eta\}$. Each hypothesis, η is a rectangle in the image where target objects will be detected, and

$$\eta = [x_\eta, y_\eta, w_\eta, h_\eta].$$

So each η is defined by its starting (x, y) coordinate, its width, and its height. To generate $\{\eta\}$, the sliding window schema is followed, and it works by enumerating all possible rectangles by considering subwindows' positions and sizes. In Algorithm 2, Ω is the set of final detection results, P_{th} is a threshold to accept hypotheses as detections, I_{te} is a test image, and S_η is the point set generated by local features contained in η.

5.1.3 Dividing Visual-spatial Space

What is very important in Algorithm 2 is how to define the set of dividing methods, γ. In the method of [9], $\mathbf{g}_i$ means dividing each dimension of the point space into 2^i intervals. However, the space in [9] is a pure feature space, while the space here is a visual-spatial space. And also, in [9], the two point sets both belong to objects, while here one point set belongs to the super template.

Algorithm 2 Detection Procedure

```
1: $\Omega \leftarrow \emptyset$, generate $\{\eta\}$ from $I_{te}$
2: for $\eta \in \{\eta\}$ do
3:   Calculate $\mathrm{P}(S_\eta, S_{\mathbf{T}}; \gamma)$
4: end for
5: Sort $\{\eta\}$ by $\mathrm{P}(S_\eta, S_{\mathbf{T}}; \gamma)$ in descending order
6: while $\mathrm{P}(S_{\eta_1}, S_{\mathbf{T}}; \gamma) >= \mathrm{P}_{th}$ do
7:   $\Omega \leftarrow \Omega + \eta_1$
8:   $\{\eta\} \leftarrow \{\eta\} - \eta_1$
9:   for $\eta \in \{\eta\}$ do
10:    for $\eta^{'} \in \Omega$ do
11:      for $\mathbf{p} \in \mathbf{S}_\eta$ do
12:        if $(\mathbf{p}^{(d-1)}, \mathbf{p}^d)$ is inside $\eta^{'}$ then
13:          $S_\eta \leftarrow S_\eta - \mathbf{p}$
14:        end if
15:      end for
16:    end for
17:    Calculate $\mathrm{P}(S_\eta, S_{\mathbf{T}}; \gamma)$
18:  end for
19:  Sort $\{\eta\}$ by $\mathrm{P}(S_\eta, S_{\mathbf{T}}; \gamma)$ in descending order
20: end while
21: return $\Omega$
```

The space-dividing method proposed here divides the dimensions of visual features and spatial coordinates at different grid sizes. Let

$$\mathbf{g} = g(i, j), i, j \in N.$$

Here $g(i, j)$ is a function which defines how to divide the visual-spatial space. And i means each dimension belonging to visual channel is divided in to 2^i intervals, and j means each dimension belonging to spatial channel is divided into 2^j intervals. Note, that for a point, $\mathbf{p}$, the first $(d-2)$ dimensions belong to visual channel, while the remaining 2 dimensions belong to spatial channel. For example, if $d = 3$, then $g(2, 3)$ will divide the whole space into $(2^i)^{(d-2)} \times (2^j)^2 = 256$ grids.

In Figure 8, an example is given by considering visual information as one dimension, and spatial information as the other dimension. Note, the total dimension of a point is actually d, while in the example it is 2.

About γ, not only its members, but also the order of its members is important.

Though (5) can be used to calculate a pyramid match score for two point sets, given any set, γ, still the dividing methods and the order of the dividing methods will affect performance. For a largest fineness level, $l_{max}, l_{max} \in N$, γ is defined in Algorithm 3.

The size of γ, $L = l_{max} \times l_{max}$. For two dividing methods $\mathbf{g}_i, i \in 1, 2, ..., L$ and $\mathbf{g}_j, j \in 1, 2, ..., L$, if $\mathbf{g}_i > \mathbf{g}_j$, then $i < j$, which means if one dividing method is finer than the other, it will appear earlier in the set of dividing methods. There are also dividing methods, of which the fineness level cannot be compared, i.e., $g(1, 2)$ and $g(2, 1)$ as shown in Figure 8.

Algorithm 3 Generation of Dividing Method Set

```
1: $\gamma \leftarrow \emptyset$, $r \leftarrow 2 \times (l_{max} - 1)$
2: while $r \geq 0$ do
3:   if $r \geq l_{max} - 1$ then
4:     $i \leftarrow l_{max} - 1$
5:   else
6:     $i \leftarrow r$
7:   end if
8:   $j \leftarrow r - i$
9:   while $i \leq (l_{max}-1)$ and $i \geq 0$ and $j \leq (l_{max}-1)$ and $j \geq 0$ do
10:    $\gamma \leftarrow \gamma + g(i, j)$, $i \leftarrow i - 1$, $j \leftarrow r - i$
11:  end while
12:  $r \leftarrow r - 1$
13: end while
14: return $\gamma$
```

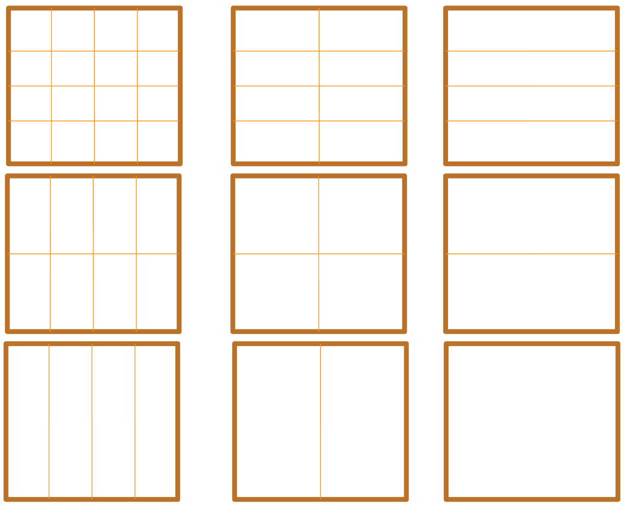


Figure 8: An example set of methods to divide the visual-spatial space. $x-$ and $y-$ coordinates represent visual and spatial information respectively. From left to right, the first line is $g(2, 2)$, $g(1, 2)$, and $g(0, 2)$. The second line is $g(2, 1)$, $g(1, 1)$, and $g(0, 1)$. And the third line is $g(2, 0)$, $g(1, 0)$, and $g(0, 0)$. And γ is defined as an ordered set of all the dividing methods with different parameters, i.e., $\gamma = \{g(2, 2), g(1, 2), g(2, 1), g(0, 2), g(1, 1), g(2, 0), g(0, 1), g(1, 0), g(0, 0)\}$.

5.1.4 Deciding Weights for Dividing Methods

After how to divide the visual-spatial space is decided, the remaining task is, for each dividing method $\mathbf{g}$, defining a corresponding weight. When talking about two points which are found in the same grid under $\mathbf{g} = g(i,j)$, there is an upper bound to their $L1$-distance, which is given by

$$D_{ub} = (d-2) \times \frac{1}{2^i} + 2 \times \frac{1}{2^j},$$

if unit length is assumed for all $(\mathbf{p}^k_{max} - \mathbf{p}^k_{min}), k \in \{1, 2, ..., d\}$. Since the first $(d-2)$ dimensions of each grid under $g(i,j)$ possess length of $\frac{1}{2^i}$, while the last 2 dimensions possess length of $\frac{1}{2^j}$.

Following [9], for two point from different point sets, if they are in the same grid under $g(i,j)$, which means $G(\mathbf{p}_{S_1}; g(i,j)) = G(\mathbf{p}_{S_2}; g(i,j))$, the visual difference between $\mathbf{p}_{S_1}$ and $\mathbf{p}_{S_2}$ is defined as $\frac{(d-2)}{2^i}$, and the spatial difference is defined as, $\frac{2}{2^j}$. A weight, ω, defined for a dividing method $g(i,j)$ shows the importance of two matched points, and measures how difficult it is to match under such dividing method.

$$\omega_{g(i,j)} = \sqrt{((d-2) \times 2^i) \times (2 \times 2^j)}. \tag{6}$$

As is seen in (6), the finer one grid is in $g(i,j)$, the larger a weight will be assigned for it. The weight is the confidence that the point set belong to a target object based on a point has corresponding evidence from the super template under the current $\mathbf{g}$.

5.2 Experimental Results

The key value of the method is the proposed matric between an object hypothesis and the super template trained from training examples. Accordingly, two experiments are carried on and the results are reported in this paper.

5.2.1 UIUC Cars

UIUC cars [1] can be considered as one of the most famous datasets, and it has been used as benchmarks in the area of detection.

Performance of the method upon different parameters will be evaluated on the UIUC cars, and compared with DPM [6].

In Figure 9, performance are evaluated between the proposed method, and a state-of-the-art method, DPM [6]. The proposed method performs no worse than DPM, while the training time between the two models are 1 minute vs 16 hours. In this experiment, the training speed of the proposed method is 1,000 times faster. Also during the training of the proposed method, since (6) is used, only the positive training examples are used, while DPM uses both positive and negative training examples.

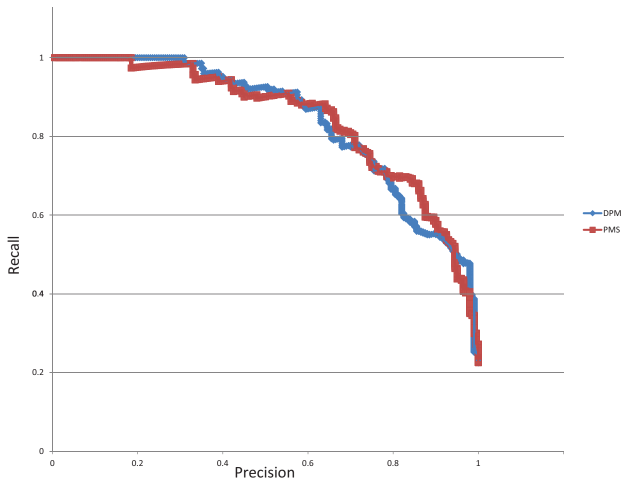


Figure 9: Result evaluation on UIUC cars between PMS and DPM.

5.2.2 P-campus

The method is also tested on a dataset of pedestrians [24]. For a fair comparison, both training images and test images are exactly same for method comparisons.

In Figure 10, there are the results of comparing the proposed method with ordinary Hough transform and common fate Hough transform [24]. It can be seen that, common fate Hough transform performs the best since the method employs motion information. The proposed method almost perform as well as the Hough transform on this dataset.

The method detects at a frame rate of 8 frames per second with multi-thread accelerations, while common fate Hough transform needs two minutes to deal with one frame in its old implementation.

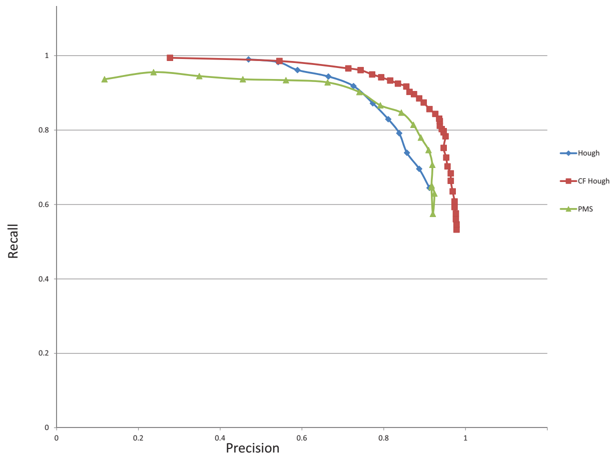


Figure 10: Precision-Recall curves for the method, Hough transform, and common fate Hough transform.

The training time of the three methods are also compared. Both Hough transform and common fate Hough transform spend 483 ms for training, while pyramid match score spend 978 ms for training. All methods spend less than one second for training.

5.3 Section Conclusion

This paper proposes a method to efficiently and effectively combine visual and spatial information of the local image features. Experiments show the method is comparable with state-of-the-art detection methods on benchmark datasets. What make the method different are: 1) pyramid matching between a codebook and an object hypothesis, 2) the set of methods to dividing the visual-spatial space, and 3) the way to define or learn weights for corresponding dividing method.

The underlying principle of PMS, which is defined by the order of dividing methods in Algorithm 3, is that use the template to explain every visual-spatial point of an object hypothesis at the best visual-spatial level.

To the best of the author's knowledge, this is the first attempt of pyramid matching between a codebook and an object hypothesis.

6 Conclusion

In 3, a method to efficiently combine motion and appearance information is proposed. The method gives promising results under simple scenarios. To improve the results of 3 on complex scenarios, in 4, the Hough transform framework is extended to incorporate motion information, and performs well on two datasets. Still the Hough transform framework itself is troublesome when considering about efficiency. This lead to the method of 5, which is a new voting system, and is very different in the using of visual and spatial information. The method's effectiveness is proven by experiments, and also it is theoretically promising.

Visual object detection by computers is still and, in near future, continues to be a very open problem. A very hopeful effort would be combining motion, appearance and location information of local features in a robust voting system.

This paper focuses on improving voting-based detection methods's detection performance by fusion of information of different channels. And in practice, the efficiency of voting is an obstacle, which lead to a new efficient voting system. The method of 3 uses voting to summarise local visual and motion patterns. The simple appearance model and the linear assumption for motion make it only suitable for particular application cases. The method performs well in detection thermal features in tunnels. The method of 4 extends the implicit shape model with motion. This method does not have assumptions for motion model, instead, online motion information are used for clustering local visual patterns, which results in better voting results. Though performing promisingly, the method's efficiency is prevented by the time consuming property of voting systems. In 5, a new voting system is proposed, by proposing new visual-spatial space dividing method for pyramid matching, the speed of voting is highly enhanced.

Thus detection performance is improved by combining motion information with appearance information in voting, and voting efficiency is improved by proposing new voting mechanisms in the paper.

References

[1] S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *PAMI*, 26(11):1475–1490, Nov. 2004.
[2] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *CVPR*, pages 73–80, 2010.
[3] O. Barinova, V. Lempitsky, and P. Kohli. On detection of multiple object instances using hough transforms. In *CVPR*, pages 2233–2240, 2010.
[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages 886–893, 2005.
[5] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61(1):55–79, January 2005.
[6] P. F. Felzenszwalb, D. A. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, pages 1–8, 2008.
[7] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, pages II: 264–271, 2003.
[8] V. Ferrari, F. Jurie, and C. Schmid. Accurate object detection with deformable shape models learnt from images. In *CVPR*, pages 1–8, 2007.
[9] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *ICCV*, pages 1458–1465, 2005.
[10] H. Jégou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *IJCV*, 87(3):316–336, February 2010.
[11] M. Jones, P. Viola, P. Viola, M. J. Jones, D. Snow, and D. Snow. Detecting pedestrians using patterns of motion and appearance. In *ICCV*, pages 734–741, 2003.
[12] K. Kise and T. Kashiwagi. 1.5 million subspaces of a local feature space for 3d object recognition. In *ACPR*, pages 672–676, 2011.
[13] C. Lampert, M. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR*, pages 1–8, 2008.
[14] Q. V. Le, M. Ranzato, R. Monga, M. Devin, G. Corrado, K. Chen, J. Dean, and A. Y. Ng. Building high-level features using large scale unsupervised learning. In *ICML*, 2012.
[15] B. Leibe, N. Cornelis, K. Cornelis, and L. Van Gool. Dynamic 3d scene analysis from a moving vehicle. In *CVPR*, pages 1–8, 2007.
[16] B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *IJCV*, 77(1-3):259–289, May 2008.
[17] B. Leibe and B. Schiele. Interleaved object categorization and segmentation. In *BMVC*, pages 759–768, 2003.

[18] S. Maji, A. C. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *CVPR*, pages 1–8, 2008.
[19] S. Maji and J. Malik. Object detection using a max-margin hough transform. In *CVPR*, pages 1038–1045, 2009.
[20] K. Mikolajczyk, B. Leibe, and B. Schiele. Multiple object class detection with a generative model. In *CVPR*, pages I: 26–36, 2006.
[21] K. Ohba and K. Ikeuchi. Detectability, uniqueness, and reliability of eigen windows for stable verification of partially occluded objects. *PAMI*, 19(9):1043–1047, September 1997.
[22] H. Schneiderman and T. Kanade. Object detection using the statistics of parts. *IJCV*, 56(3):151–177, 2004.
[23] P. A. Viola and M. J. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, 2004.
[24] Z. Wang, M. Kagesawa, S. Ono, A. Banno, and K. Ikeuchi. Emergency light detection in tunnel environment: An efficient method. In *ACPR*, pages 628 – 632, 2010.
[25] L. Yang, N. Zheng, M. Chen, Y. Yang, and J. Yang. Categorization of multiple objects in a scene without semantic segmentation. In *ACCV*, pages 303–312, 2009.
[26] T. Yeh, J. Lee, and T. Darrell. Fast concurrent object localization and recognition. In *CVPR*, pages 280–287, 2009.