COMBINING MOTION INFORMATION WITH APPEARANCE INFORMATION AND UTILIZING OF MUTUAL INFORMATION ENCODED AMONG IMAGE FEATURES OF THE SAME OBJECT FOR BETTER DETECTION PERFORMANCE

運動情報を外観情報との結びつけおよび同一物体の画像特徴によりエンコードされた相互情報の利用ーー検出性能の向上のため

by

Zhipeng Wang
王志鵬

A Senior Thesis
卒業論文

Submitted to
the Department of Information Science
the Faculty of Science, the University of Tokyo
on February 05, 2013
in Partial Fulfillment of the Requirements
for the Degree of PhD of Science

Thesis Supervisor: Katsushi Ikeuchi　池内　克史
Professor of Information Science

ABSTRACT

Object detection is a fundamental perceptual skill in human, and plays an important role in the machine vision area. There are two main categories of detection methods: methods using the sliding-window schema and methods based on Hough transforms. Among the methods, recent research improves detection performance by proposing better representative model, better classifiers, and better solution space search methods. We, in our work, propose four methods, of which two make use of motion information by combining it effectively and efficiently with appearance information, and two make use of the mutual information encoded among the image features of the same object.

論文要旨

物体検出は人間の中の基本の知覚能力で、マシン・ビジョン・エリアに重要な役割を果たします。2 つの主なカテゴリーの検知方法があります: スライディングウィンドウを使用する方法、およびハフ変換に基づいた方法。最近の研究は、提案する、よりよい代表的なモデル、よりよいクラシファイヤーおよびよりよい解空間探索方法によって検出性能を改善します。私たちは 4 つの方法を提案します。2 つの方法が運動情報を外観情報との結び。2 つの方法が同一物体の画像特徴によりエンコードされた相互情報の利用。

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Chapter 2

Background and Related Work

Chapter 3

Efficient Detection by Combining of Appearance and Temporal Information

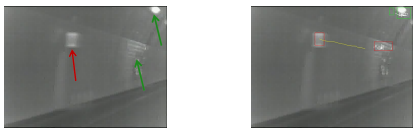
3.1 Introduction

Positioning of automobiles acts a fundamental role in autonomous driving, and it is also of great importance for driving assistance, vehicle navigation, etc. When GPS sensors function properly, it is usually easy for an automobile to estimate the position of itself. While in tunnel environment, for most of the time, there is no GPS signals available. A new positioning system which functions properly in tunnel environment is very necessary [7]. We have tackled this issue as a part of automated driving system in a NEDO project, "Development of Energy-saving ITS Technologies".

In most tunnels which appear on express ways in Japan, there are lots of signs which appear at equal intervals. We focus on the emergency telephone indicators, which appear every 200 meters in tunnels. The absolute coordinates of the emergency telephone indicators can be obtained by the method of [40]. While travelling in tunnels, if the emergency telephone indicators can be sensed, and the distance from the automobile to the indicators can be estimated, then the absolute position of the automobile can be inferred. Detection methods i.e. [39] based on ordinary cameras fail due to darkness. Here we use infrared cameras, which are usually equipped by recent intelligent vehicles, and which are suitable in dark environment. Moreover, far-infrared stereo system is developping in our automated driving system. Inspired by a previous work [37], we propose an approach to detecting emergency telephone indicators by using infrared cameras.

Detection performance and efficiency are the two important aspects of our method.

In tunnel environment, besides the target objects, 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 3.1. The clutter property of the sensed data makes the detection challenging. Our method meets this challenge by making use of both appearance and temporal information of the target objects. There are two main steps in the method. The first step deals with keypoints, and it takes original data as input, and outputs keypoint clusters as detection hypotheses. In this step, keypoints are detected, verified and then clustered. To detect keypoints, all points on each frame are uniformly sampled and filtered with preset intensity thresholds. Then the keypoints are verified by a simple keypoint appearance model built by k-means. At the end of the first step, the keypoints are clustered based on the Euclidean distance. The second step takes the keypoint clusters as input, and verifies them by appearance and temporal information, and outputs the ones pass verifications as detection results. In the second step, the keypoint clusters are labeled based on appearance by an adaboost machine, which is trained using intensity histograms of keypoint clusters from target objects and keypoint clusters from noisy objects. The keypoint clusters are also tracked by temporal association through frames. Motion information



(a) (b)

Figure 3.1: 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.

encoded in the trajectories are used to further verify the keypoint clusters. Finally, the keypoint clusters which pass both appearance and temporal verifications are decided as emergency telephone indicators.

This pipeline is designed also with consideration of the requirement for efficiency. The method deals with the large amount information contained on one frame following a hierarchical manner. The later one step is, the more time-consuming it is and the fewer instances it deals with. From an image containing 10^5 pixels, 10^4 points go through keypoint detection step of testing by intensity thresholds. Then in average, 10^3 keypoints are detected, and verified, leaving about 10^2 to be clustered. Afterwards, fewer than 10 keypoint clusters are left, which are dealt with by the very time-consuming steps of generating image features and tracking.

The advantage of the method is its ability to give promising detection results from cluttered data in real time. Besides, the method is also a successful attempt to combine bottom-up and classification methods, and a successful attempt to combine both appearance and temporal information.

The paper is organized as follows: section 2 reviews related work, section 3 proposes pipeline of the method, section 4 gives experimental results, and section 5 concludes.

3.2 Related Work

Most modern detection methods fall into two categories. Some [6, 8, 11, 17, 25, 32, 36, 43] follow the sliding-window schema, and they detect objects by consider whether each of the sub-images contains an instance of the target object. Classifiers are usually employed by these methods. The other methods [2, 9, 10, 19, 20, 21, 24, 26, 28] infer object centers based on local image features in a bottom-up manner. The proposed method makes advantages of both frameworks. Following the bottom-up manner, keypoints are detected, verified, and clustered. After these steps, the keypoint clusters are considered as detection hypotheses. Then following the sliding-window schema, the keypoint clusters are verified by their appearance and temporal information using discriminative methods.

Previous methods [31] also consider the combination of the two frameworks. Detection hypotheses are gained using Hough transform and then verified by support vector machines in [24, 42]. The methods in [12, 29], use randomized decision trees for both decisions whether local features belonging to foreground objects or not and decisions of their Hough votes. The method proposed in [18] describes both frameworks in the same manner. While giving state-of-the-art detection performance, they can't meet the requirement for efficiency as our method does.

(a) (b)

Figure 3.2: Keypoint detection.

Our work is also related to feature grouping methods [42], methods detecting using trajectories [4, 5], tracking methods [22, 15], and methods integrating appearance and temporal information [39].

Especially, compared with the method proposed in [37], our method employs a more effective classifying machine by setting biased weights for positive and negative training examples, and far over-perform the method.

3.3 Emergency Telephone Indicator Detection

The method can be considered as a two-step method. The first step deals with keypoints. It takes original data as input, and outputs keypoint clusters as detection hypotheses. 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.

3.3.1 Keypoint Detection

In data collected using ordinary cameras, keypoints [3, 23] invariant to rotations, affine changes, and illumination changes are preferable. In our case, keypoint detection intends to provide hypotheses for emergency telephone indicators. Thus intensity is of great importance. Our method employs a simple yet useful method to detect keypoints. Firstly, points are uniformly sampled with width step 6, and height step 7 (the length of emergency telephone indicator is larger than its width). In this manner the magnitude of instances is reduced by nearly two orders. Then the points pass the test which verifies the points by setting intensity thresholds are considered as keypoints. Here Gaussian distribution is assumed for the intensities of the points.

let$\{\mathbf{x}\}$ denote all the sampled points, $I_{\mathbf{x}}$ the intensity of each point, and $l_{\mathbf{x}}$ the label. If the point is considered as belonging to emergency telephone indicators, $l_{\mathbf{x}} = 1$, otherwise, $l_{\mathbf{x}} = 0$. By setting lower threshold, $I_{\mathbf{x}}^{th1}$, and higher threshold, $I_{\mathbf{x}}^{th2}$, the probability that points belongs to emergency telephone indicators based on their falling into this interval is given by,

$$P(l_{\mathbf{x}} = 1 | I_{\mathbf{x}}^{th1} \leq I_x \leq I_{\mathbf{x}}^{th2}) = \frac{P(l_{\mathbf{x}} = 1, I_{\mathbf{x}}^{th1} \leq I_x \leq I_{\mathbf{x}}^{th2})}{P(I_{\mathbf{x}}^{th1} \leq I_x \leq I_{\mathbf{x}}^{th2})} \ . \tag{3.1}$$

At this step, that as few points belonging to the emergency telephone indicators as possible are excluded is also considered. The probability of ont point falling into the defined interval based on its belonging to emergency telephone indicators is given by,

$$P(I_{\mathbf{x}}^{th1} \leq I_x \leq I_{\mathbf{x}}^{th2} | l_{\mathbf{x}} = 1) = \frac{P(l_{\mathbf{x}} = 1, I_{\mathbf{x}}^{th1} \leq I_x \leq I_{\mathbf{x}}^{th2})}{P(l_{\mathbf{x}} = 1)} \ . \tag{3.2}$$

And points of which the intensities fall in the pre-set thresholds are detected as keypoints.

(a) (b) (c) (d)

Figure 3.3: Keypoint verification and clustering. Red circles mark keypoints which pass the verification, while blue marks failed ones. Rectangles mark keypoint clustering results.

3.3.2 Keypoint Verification

As shown in Figure 3.2(b), the detected keypoints don't just belong to emergency telephone indicators, but also belong to background. And for training, keypoints belonging to emergency telephone indicators are considered as positive ones, otherwise negative.

To verify the keypoints, the appearance of the sub-image around each keypoint is used. Intensity histograms are used to describe the appearance. Noisy keypoints not only come from the wall of the tunnel, but also from ordinary lights, other vehicles, and other vehicles' shadows. Thus robust linear classifiers are not suitable for the verification. Here, a general model in the form of a simple mixture is used. The *k*-means method is used to cluster the intensity histograms, $\{A_{\mathbf{x}}, l_{\mathbf{x}} = 1\}$, of the positive keypoints, and, $\{A_{\mathbf{x}}, l_{\mathbf{x}} = 0\}$, of the negative keypoints.

Let $\{C_1^i, i = 1, 2, ..., n_1\}$ denote the intensity histogram centers of the positive keypoints, and $\{C_0^i, i = 1, 2, ..., n_2\}$ the negative. For each C_1, the average Euclidean distance between $\{C_0^i, i = 1, 2, ..., n_2\}$ is calculated as,

$$Eu(C_1^i) = \frac{1}{n_2} \sum_{j=1}^{n_2} Euclid(C_1^i, C_0^j) \ . \tag{3.3}$$

Here, $Euclid(\cdot)$ calculates the Euclidean distance, and $Eu(\cdot)$ is a evaluation function of the positive feature centers. The positive feature centers are ranked by $Eu(\cdot)$, and the 10 positive feature centers with largest $Eu(\cdot)$ are chosen and used for verification.

For verification, the intensity histogram of each keypoint's surrounding sub-image is extracted. Then the Euclidean distance between the extracted intensity histogram and its nearest positive feature center is calculated. If this distance exceeds a threshold, $D_{A_{\mathbf{x}}}^{th}$, it is considered as negative, else it is considered as positive. Here, for simplicity, unlike [33], the same threshold is used for all components of the mixture.

3.3.3 Keypoint Clustering

After the keypoint verification step, on some frames, the result is pretty good, while on other frames, appearance of the keypoints is not enough for decisions of whether the keypoints belonging to emergency telephone indicators or not. Here generation of keypoint trajectories is not feasible, since nearby keypoints are similar in appearance and the time complexity of associating such a large number of keypoints along time dimension is high. So the keypoints are clustered, and then data association in time dimension only need to deal with a small number of keypoint clusters.

To cluster the keypoints, a minimum spanning tree (mst) is built using the pairwise Euclidean distance between two keypoints. And the mst is split by cutting edges larger a threshold. This results in a grouping results of the keypoints, denoted by, $\gamma = \{\mathbf{g}\}$.

3.3.4 Keypoint Cluster Verification by Appearance

For each keypoint cluster, the smallest bounding rectangle is considered as detection hypothesis, as shown in Figure 3.3(c) and Figure 3.3(d). There are two main sources of noise. Ordinary lights are the first, and other vehicles with their shadows are the second. The global appearance of ordinary lights is different from that of the emergency telephone indicators. As ordinary lights get further from the infrared camera, the intensity of its corresponding sub-image in the collected data gets lower. At a certain distance, the intensity of the ordinary lights is almost the same with emergency telephone indicators'. And for ordinary lights of which the intensity is higher than the emergency telephone indicators', the transition regions from them to tunnel walls will have similar intensity with the emergency telephone indicators. This means though locally the emergency telephone indicators share the same appearance with ordinary lights, they can still be distinguished globally by appearance. As for other vehicles and their shadows, the intensity range of them is very close to the emergency telephone indicators', and they can hardly be distinguished just by appearance.

At this step, the keypoint clusters are verified by their appearance, aiming at excluding keypoint clusters belonging to the ordinary lights. An Adaboost machine is trained using intensity histograms of the emergency telephone indicators and ordinary lights. The appearance of other vehicles is close to the emergency telephone indicators', and they are not used for training the machine. For training of the machine, labeled 32-dimensional intensity histograms are firstly normalized. Then each weak classifier of the machine makes decision on one dimension of the intensity histograms. After this step, each keypoint cluster is either labeled as positive or negative.

In this step, to emphasize the Adaboost machine's performance on the positive training examples, we set the initial weights of the positive training examples 7 times as large as the weights of the negative training examples. Since in practice, whether each keypoint cluster is a target object or not is decided by both appearance and motion information. And the difficulties of exclude noisy objects can be left to the later steps.

3.3.5 Keypoint Cluster Tracking

Not all noisy detection hypotheses can be excluded by using appearance, as shown in Figure 3.4. To distinguish keypoint clusters belonging to other vehicles and their shadows, the keypoint clusters are tracked through frames to generate trajectories.

In our case of keypoint cluster tracking, the problem is relatively simple, since no occlusion occurs. To keep the method on-line and maintain efficiency, a pool of trajectories are kept, $\tau = \{T^i_{\mathbf{g}}, i = 1, 2, ..., n\}$, and new detection hypotheses act as detection responses, $\nu = \{n^i_{\mathbf{g}}, i = 1, 2, ...m\}$, in tracking. The problem of tracking is modeled as finding best data association hypothesis, H^*, between the trajectory set and detection

Figure 3.4: Keypoint cluster verification by appearance. Red rectangles: positive detection hypotheses, and green: negative detection hypotheses.

response set as,

$$\begin{aligned} H^* &= \arg\max_{H\in\eta}(P(H|\tau,\nu)) \\ &= \arg\max_{H\in\eta}(\prod_{(T^i_{\mathbf{g}},n^j_{\mathbf{g}})\in H} P_{link}(n^j_{\mathbf{g}}|T^i_{\mathbf{g}}))\ . \end{aligned} \tag{3.4}$$

Let $u_{ij} = 1$ or 0 indicates $n^j_{\mathbf{g}}$ is linked to $T^i_{\mathbf{g}}$ or not, and assuming each trajectory can link once and each detection response can only be linked once, the problem can be modeled as,

$$\begin{aligned} &\arg\max_{u_{ij}} \sum_{i=1}^{n}\sum_{j=1}^{m} u_{ij} \ln P_{link}(n^j_{\mathbf{g}}|T^i_{\mathbf{g}}) \\ &s.t.:\ u_{ij} = 0 \text{ or } u_{ij} = 1, \forall\, i, \forall\, j; \\ &\sum_{i=1}^{n} u_{ij} \le 1\ ; \sum_{j=1}^{m} u_{ij} \le 1\ . \end{aligned}$$

Here, $P_{link}(n^j_{\mathbf{g}}|T^i_{\mathbf{g}})$ is defined by the appearance difference, the scale difference, and the time gap between the last detection response contained in $T^i_{\mathbf{g}}$ and $n^j_{\mathbf{g}}$. While Hungarian algorithm [16] gives near-optimal solution, we follow a very simple manner for the solution by finding the best matched pairs and excluding them until no matching pairs can be found.

3.3.6 Keypoint Cluster Verification by Motion

As shown in Figure 3.5, the trajectories from keypoint clusters belonging to emergency telephone indicators are different from other objects'. In this step, the temporal information encoded in the trajectories are used to further verify the keypoint clusters. A linear model is used to fit each trajectory, and the significance of the fitting is the criteria for decisions. Let $(x^i_{\mathbf{g}}, y^i_{\mathbf{g}})$ denote the coordinate of the ith element belonging to a trajectory. The linear assumption is that $y^i_{\mathbf{g}} = a_0 + a_1 x^i_{\mathbf{g}}$. The significance of the fitting is defined as,

$$r = |\frac{\sum_i \left(x^i_{\mathbf{g}} - \bar{x}_{\mathbf{g}}\right)\left(y^i_{\mathbf{g}} - \bar{y}_{\mathbf{g}}\right)}{\left[\sum_i \left(x^i_{\mathbf{g}} - \bar{x}_{\mathbf{g}}\right)^2 \cdot \sum_i \left(y^i_{\mathbf{g}} - \bar{y}_{\mathbf{g}}\right)^2\right]^{1/2}}| \ . \tag{3.5}$$

And r is used to decide the trajectories of the keypoint clusters as belonging to emergency telephone indicators or not.

3.3.7 Object Detection

For each keypoint cluster on the current frame, there exists label given by the Adaboost machine according to its appearance, and the significance of fitting its trajectory as a straight line. For each keypoint cluster, it is considered as an emergency telephone indicator if and only if its label given by the Adaboost machine is positive, its trajectory is longer than l^{th}, and the significance of fitting its trajectory into a straight line is larger than r^{th}.

Each trajectory not only connects the detection responses, but also connects the decisions for each 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 on other frames. The final results is 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 as positive.

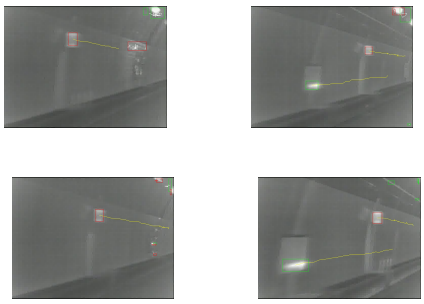


Figure 3.5: Detection results.

3.4 Experimental Results

We test our method on detection performance and efficiency.

Data To collect data, we mount an infrared camera on top of the experimental vehicle, and then take several tours of the Awagatake tunnel. About 7,000 frames are collected for each tour. The frame size is 640×480, the intensity range is [0,255], and the frame rate is 30 frames per second.

Implementation Settings All models are trained using data from the same tour, while evaluated on data from another tour.

To set intensity thresholds for keypoint detection, Gaussian distribution is assumed for the points belonging to emergency telephone indicators. Following the 3σ principle, $I_{\mathbf{x}}^{th1}$ is set to 160 and $I_{\mathbf{x}}^{th2}$, 190.

The approximate sub-images of the emergency telephone indicators are manually marked, and used for training the mixture model of keypoint verification. The detected keypoints falling into the sub-image are marked as positive, and otherwise negative. Note this model and the training is not very accurate, since more accurate marking means more manual efforts. About 30,000 intensity histograms of the positive keypoints are sampled, and about 3,000,000 of the negative. When using of k-means for clustering the positive intensity histograms, k is set to 40, and 400 for negative. The k values oversegment both feature sets. The threshold to verify keypoints, $D_{A_{\mathbf{x}}}^{th}$ is set to 0.14 for the normalized histograms.

For keypoint clustering, the threshold to split the mst is set to 40, which is half the largest height of the emergency telephone indicators.

The Adaboost machine to distinguish other vehicles and their shadows is trained by intensity histograms of positive keypoint clusters and negative keypoint clusters. We manually mark positive and negative keypoint clusters. If the Adaboost machine is trained by averagely weighted training examples, its correct rate on the training examples is overall 84%. When trained using our bias weighted training examples, its correct rate on the positive training examples is 94%, and 77% on the negative training examples.

During keypoint cluster tracking, whether a detection response can be linked to a trajectory or not is constrained by position and scale changes. Here scale change limit is set to 4. When the trajectories are fitted as lines, the linear model is also used in associating new detection responses.

Detection Results

On an ordinary desktop computer with Intel Core2 Quad 2.6GHz processors, the method deals with real data at a frame rate of 41 frames per second, and this fulfills real-time requirements.

The detection rate and false alarm rate are evaluated on the keypoint clusters, as

shown in TABLE 3.1. More detection results are shown in Figure 3.5.

total number	472 + 3304
correctly labeled	468
miss detections	4
false alarms	22
detection rate	99.2%
false alarm rate	0.7%

Table 3.1: Detection rate and false alarm rate.

The detection rate and false alarm rate of [37] are 90% and 19%, while evaluated on a much smaller dataset. We outperforms [37], because our sensed images are much clearer, and also because our more effective training of the Adaboost machine.

The results on the trajectories of decisions are also evaluated. 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%.

3.5 Conclusion

We propose an object detection method to detect emergency telephone indicators in tunnel environment. 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. Based on the detection results, a positioning system in tunnel environment can be expected.

Chapter 4

Grouping of object parts by motion for detection

4.1 Introduction

~~In ITS areas, detection methods using cameras can be used for navigation, safe driving, surveillance, and sustaining results from other sensors. These methods can be used to detect pedestrians, bicycle riders, and automobiles, and here we focus on techniques from the area of computer vision for detection of such objects.~~ In ITS areas, detection methods using cameras can be used for navigation, safe driving, surveillance, and sustaining results from other sensors. In traditional ITS applications, vehicles are main targets. Currently pedestrians are also considered as important subjects of ITS applications, and bicycles also become very popular for environmental and economical reasons. In Japan, the number of traffic accidents among bicycles and pedestrians is very large. Thus we tackle an issue of detecting freely moving bicycle riders and pedestrians from the data collected by a camera which keeps them under surveillance from the top. These situations can be observed in parks, university campuses, station squares, tourist spots, etc. Here we focus on techniques from the area of computer vision for detection under surveillance scenarios.

Most state-of-the-art visual detection methods fall into two main categories: sliding-window methods and Hough transform based methods. The methods [17, 43] based on a sliding window schema perform detection in a typical machinery way. In these method, decisions of whether a target object exists or not are made for part of or all the sub-images in a test image. Beside the attractive performance and the extendibility of combining various kernels, these methods are favorable also because they consider each object as a whole during detection. However, they share limited aspects with visual perception in human beings, and the efficiency heavily relies on the size of the test images.

The other methods [20, 9, 10, 28] detect objects based on the generalized Hough transform [1]. Object parts are detected, and the object parts provides confidence of locations being potential objects' centers. Locations of objects are decided according to the converged confidence. They are favorable for the robustness to partial deformation and easiness of training. To human beings, this kind of methods seems to be more natural. And in our work, we combine a mechanism of visual perception in human with the ISM [20] to demonstrate this natural property.

A typical Hough transform based method contains two steps: training and detection. During training, a codebook of object parts is built from a set of well annotated images. Each code in the codebook contains information about the appearance of the object part, the relative position to the object center, and the class label. Each object part's appearance is given in the form keypoint descriptors [20], image patches [12, 29], or image regions [13]. Each code not only encode one object part's appearance, but also its offset to the ~~the~~ object center and the class label. while during detection, on each test image,

(a) (b) (c) (d)

Figure 4.1: Merit of the proposed method. (a) Original image. (b) Motion ~~group~~grouping results. Some parts are enlarged to show details. (c) Original Hough image. (d) Hough image formed using our method. The grids in (c) and (d) correspond to the grids in(a).

object parts are detected. Then every object part is matched against the codebook, and activates several nearest codes in appearance. The offset and class label encoded in each activated code will act as a vote. All the votes from the object parts are added up to form a Hough image. The peaks of the Hough image are considered as detection hypotheses with the height of each peak as the confidence for the corresponding hypothesis.

Two challenging issues for detection methods are how to separate near objects and how to separate similar different-class objects. The target objects, in the case of ITS applications, are pedestrians, bicycle riders, and automobiles. In the schema of sliding window, usually ~~maximum~~non-maximum suppression is needed for post-processing, and a mechanism in [17] works by excluding from the feature pool the features which belong to each successive found detection response. In Hough transform based methods, a similar mechanism is also employed in [2], however, this effort is after the forming of Hough image. During the forming of a Hough image, two kinds of votes make detection challenging: (1) votes casted by object parts from near objects make the peaks corresponding to different objects mixed up, and (2) votes casted by similar different-class object parts lead to tough decisions on the class label of the peaks. See Figure 4.2(d). Before the forming of Hough images, problems also arise from the pollution of training images' background part to the codebook. During training a very clean codebook can be built with foreground marked, which needs manual efforts. Otherwise, large amount of training examples are needed for the effectiveness of the codebook, and this harms efficiency.

In videos, motion information is also available by simple tracking of object parts. Thus we propose a method for detection which utilizes both appearance and motion information. ~~the~~The method is based on the common fate principle [38]. The principle is one of the visual perception principles as theorized by gestalt psychologists, and it states for human beings, tokens moving coherently are perceptually grouped. This provides an intuition to group the object parts by their motion patterns, and let them vote afterwards. In our work, the object parts are represented using keypoint descriptors, which are tracked to generate trajectories. The object parts are grouped by the pairwise similarities of their corresponding trajectories. Having the assumption that object parts in the same motion group probably belong to the same object, for each object part, we assign higher weights for the votes of this object parts which are more "agreeable" within the motion group. This results in votes corresponding to true detection responses are more likely to be assigned higher weights. And on a Hough image formed by summing up these weighted votes, the peaks are easier to find as shown in Figure 4.1(d).

By the combination of motion analysis results with the Hough transform framework through assigning different weights to each object part's votes, the proposed method has several appealing properties:

- The method's ability to estimate object position and label of multiple objects from

different classes. The existence of three types of objects makes the task challenging: near objects, similar different-class objects, and multi-pose same-class objects.

- Its ability to use a codebook trained by images with cluttered backgrounds.
- The framework to combine grouping results of object parts is very general, and has a good expandability.

The remaining paper is organized as follows. Section 2 reviews related work. Section 3 gives formalism of the common fate Hough transform. Section 4 describes inference on the formed Hough images. Section 5 gives experimental results, and section 6 concludes.

4.2 Related Work

Our work is most related to object detection methods [2, 19, 20, 21, 24, 26] based on the Hough transform framework. Recently, such methods make a lot of ~~process~~progress. The ISM [20, 21] is extended by notifying correspondences between the object parts and the hypotheses [2] for the detection of multiple near objects. While in ~~[24]~~[12, 24, 29] the Hough transform is placed in a discriminative framework for object detection in a way that the codes are assigned different weights by the co-occurrence frequency of their appearance and offset to the object center. Two Hough transform methods consider the grouping of object parts [30, 42]. The method in [30] deals with scale change. Instead of estimating the scale by local features trained from different scaled examples, the votes are considered as voting lines. By considering the difference of the voted centers, local features are firstly grouped and then vote more consistently for the object center. In [42], the grouping of object parts, the correspondence between object parts and object, and the decisions on detection hypotheses are optimized in the same energy function. For this method, the problem is that the grouping results don't have meaning or correspond to and real entities.

The work is also related to object detection methods by trajectories [4, 5], methods weighting features [41], methods dealing with codebook noise [27], and methods which integrate temporal information [39].

4.3 Common Fate Hough Transform

Probabilistic standpoints are very appealing, because of inference easiness. However, as pointed in [18], placing the ISM in a probabilistic framework is not satisfactory. Especially, describing weights of the votes as priors does not make sense. Hough transform can be simply considered as transformation from a set of object parts, $\{\mathbf{e}\}$, to a confidence space of object hypotheses, $C(\mathbf{x}, l)$. And $\mathbf{x}$ is the coordinate of the object center, while l the label. Terms described as priors of the votes in the ISM are actually weights, and the likelihood terms are actually blurring functions to convert discrete votes into continuous space. Then this section describes how a Hough image for estimation of object centers and labels is formed from object parts observed on an image.

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 by 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,

$$C(\tilde{\mathbf{x}}, \tilde{l}; \mathbf{e}) = \sum_{i=1}^{N} B(\tilde{\mathbf{x}}, \tilde{l}; V_{\mathbf{e}}^i) w(V_{\mathbf{e}}^i) \,. \tag{4.1}$$

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,

$$B(\tilde{\mathbf{x}}, \tilde{l}; V) = \begin{cases} 0 & \text{if } l_V \neq \tilde{l} \text{ or } |\tilde{\mathbf{x}} - \mathbf{x}_V| > d \\ G(\tilde{\mathbf{x}}; \mathbf{x}_V, \sigma) & \text{otherwise} \end{cases} \,. \tag{4.2}$$

Here $G(\tilde{\mathbf{x}}; \mathbf{x}_V, \sigma)$ is a Gaussian function that fixes the ~~spacial~~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 a $\tilde{l}$-class object is given by,

$$\begin{aligned} C(\tilde{\mathbf{x}}, \tilde{l}) &= \sum_{j=1}^{M} C(\tilde{\mathbf{x}}, \tilde{l}; \mathbf{e}_j) w(\mathbf{e}_j) \\ &= \sum_{j=1}^{M} \sum_{i=1}^{N} B(\tilde{\mathbf{x}}, \tilde{l}; V_{\mathbf{e}_j}^i) w(V_{\mathbf{e}_j}^i) w(\mathbf{e}_j) \,. \end{aligned} \tag{4.3}$$

Here, a uniform weight is assumed for each object part, and $w(\mathbf{e}_j) = \frac{1}{M}$. Then 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.

4.3.1 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, and assume $\mathbf{e}_m \in \mathbf{g}$ and $\mathbf{e}_n \in \mathbf{g}$. Those votes of $\mathbf{e_m}$ which are more "agreeable" by the votes of the other objects in $\mathbf{g}$ are assigned larger weights.

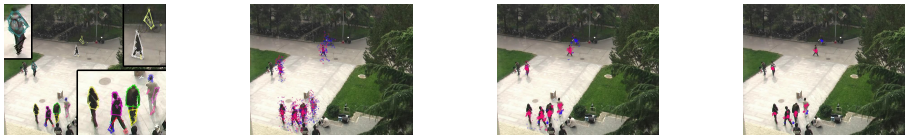
Towards this end, the relationship between the votes of $\mathbf{e_m}$ and the votes of $\mathbf{e_n}$ needs to be given in advance. This relationship is named support. The support from $V_{\mathbf{e}_n}$ to $V_{\mathbf{e}_m}$ is defined by that based on $V_{\mathbf{e}_n}$, the confidence $V_{\mathbf{e}_m}$'s voted center is correct, as,

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

Here $B(\mathbf{x}_{V_{\mathbf{e}_m}}, l_{V_{\mathbf{e}_m}}; V_{\mathbf{e}_n})$ is defined in (4.2). 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 by that based on $\mathbf{e}_n$, the confidence that $V_{\mathbf{e}_m}$'s voted center is correct, as,

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



(a) (b) (c) (d)

Figure 4.2: Effect of the proposed weight. (a) Motion groups, different colors mark different motion groups. (b) Voted centers given by the 7 best matched codes. (c) Voted centers with the highest defined weights. (d) Voted centers with weights higher than a threshold.

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 but its belonging object part in $\mathbf{g}$, as,

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

~~By assuming all object parts in the same motion group are from the same object, which means motion grouping gives good results. The center position and the class label given by one vote shall be consistent with that given by the motion group.~~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 shall 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:

$$\begin{aligned} w(\tilde{V}_{\mathbf{e}_m}) &= \frac{S(\mathbf{g} \to \tilde{V}_{\mathbf{e}_m}) + \frac{\Delta}{N}}{\sum_{i=1}^{N} S(\mathbf{g} \to V^i_{\mathbf{e}_m}) + \Delta} \\ &= \frac{\sum_{\mathbf{e}_j \in \mathbf{g} - \{\mathbf{e}_m\}} \sum_{k=1}^{N} S(V^k_{\mathbf{e}_j} \to \tilde{V}_{\mathbf{e}_m}) w(V^k_{\mathbf{e}_j}) + \frac{M\Delta}{N}}{\sum_{i=1}^{N} \sum_{\mathbf{e}_j \in \mathbf{g} - \{\mathbf{e}_m\}} \sum_{k=1}^{N} S(V^k_{\mathbf{e}_j} \to V^i_{\mathbf{e}_m}) w(V^k_{\mathbf{e}_j}) + M\Delta} \,. \end{aligned} \tag{4.4}$$

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 give $w(\tilde{V}_{\mathbf{e}_m})$, 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 to form the Hough image are weights converged in iterations.

The grouping result $\gamma = \{\mathbf{g}\}$, can be replaced by grouping results based on other information, while our method utilizes motion to group the voting elements. The manner of extending the Hough transform is very general, and the extended Hough transform with motion grouping results is called the common fate Hough transform. The votes given by the best matched codes and the votes with higher defined weights are shown in Figure 4.2.

4.3.2 Motion Grouping

In this subsection how to group the object parts by their motion patterns is introduced. Basically, the object parts are tracked, and clustered by their motion patterns. The object parts are tracked through frames before and after the current frame to generate trajectories. Then the object parts are grouped by their corresponding trajectories' pairwise motion similarity.

The object parts in this method are in the form of keypoint descriptors. The Harris Corner [14] feature is chosen, for robustness, to represent each object part, while for appearance, the region covariance [35] feature of the image patch around each keypoint is used. The image feature is chosen because of its flexibility to combine multiple channels of information, and also for its capability of handling scale changes in a certain range. For each object part, a trajectory is generated by tracking its corresponding Harris Corner by the KLT tracker [34]. To group the trajectories, two pairwise similarities are defined.

Let $T_{\mathbf{e}_m}$ and $T_{\mathbf{e}_n}$ denote two trajectories corresponding to $\mathbf{e}_m$ and $\mathbf{e}_n$. The first similarity between two trajectories is defined as,

$$D_1(T_{\mathbf{e}_m}, T_{\mathbf{e}_n}) = \max_{i=1...L} (|\mathbf{x}^i_{T_{\mathbf{e}_m}} - \mathbf{x}^i_{T_{\mathbf{e}_n}}|) \, .$$

Here, i is the frame index, and L is ~~the length of the overlapping part of the two trajectories~~the number of frames which are crossed by both trajectories.

To define the second similarity, the ith directional vector of T is firstly defined as, $\mathbf{d}^i_T = \mathbf{x}^{i+3}_T - \mathbf{x}^i_T$. Let $\mathbf{a}_i = \mathbf{d}^i_{T_{\mathbf{e}_m}}$, $\mathbf{b}_i = \mathbf{d}^i_{T_{\mathbf{e}_n}}$, $a_i = \frac{\mathbf{a}_i \cdot \mathbf{b}_i}{\mathbf{a}_i \cdot \mathbf{a}_i}$, and $b_i = \frac{\mathbf{a}_i \cdot \mathbf{b}_i}{\mathbf{b}_i \cdot \mathbf{b}_i}$. Then the second similarity is defined as,

$$D_2(T_{\mathbf{e}_m}, T_{\mathbf{e}_n}) = \max_{i=1...L-3} (\max(|\mathbf{a}_i - a_i \mathbf{a}_i|, |\mathbf{b}_i - b_i \mathbf{b}_i|)) \, .$$

Before grouping the trajectories, the static points are excluded. ~~Inspired by [4], a minimal spanning tree of the trajectories is built upon D_1, and split by cutting edges larger than a threshold, D^1_{th}. For each element of the splitting results, a minimal spanning tree is built upon D_2 and split by cutting the edges larger than a threshold, D^2_{th}.~~The defined D_1 is calculated for all pairs of trajectories, and a minimal spanning tree is then built using the calculated distances. The built mst is split by cutting edges larger than a threshold, D^1_{th}, and this gives a grouping result of the trajectories. For each element in the clustering result, D_2 is used in the same procedure to generate even smaller clusters. This hierarchical procedure ensures that trajectories in the same group have both small D_1 and D_2. Each trajectory corresponds to an object part, and the grouping results of the trajectories correspond to grouping results of the object parts.

4.3.3 Codebook

For training, Harris corners are extracted from the training images with the object center and the class label annotated. In this method, region covariance is chosen to represent the appearance, which is defined as,

$$\mathbf{r} = \frac{1}{K-1} \sum_{i=1}^{K} (\mathbf{z}_i - \mu)(\mathbf{z}_i - \mu)^T \, .$$

Here, K is the number of pixels in the region, and $\mathbf{z}_i$ is a 7-dimensional vector regarding the (x, y) coordinate of the pixel, while μ is the mean of $\mathbf{z}_i$. And $\mathbf{z}(x, y)$ contains the RGB color of the pixel and the intensity gradients of the pixel, as: $r(x, y)$, $g(x, y)$, $b(x, y)$, $|\frac{\partial I(x,y)}{\partial x}|$, $|\frac{\partial I(x,y)}{\partial y}|$, $|\frac{\partial^2 I(x,y)}{\partial x^2}|$, and $|\frac{\partial^2 I(x,y)}{\partial y^2}|$.

The appearance similarity between $\mathbf{r}_m$ and $\mathbf{r}_n$ is given by,

$$\rho(\mathbf{r}_m, \mathbf{r}_n) = \sqrt{\sum_{i=1}^{7} \ln^2 \lambda_i} \ .$$

Here, λ_i is the generalized eigenvalue by solving the generalized eigenvalue problem, $\lambda_i \mathbf{r}_m \mathbf{u}_i = \mathbf{r}_n \mathbf{u}_i, \mathbf{u}_i \neq \mathbf{0}$, with $\mathbf{u}_i$ the eigenvector.

A square image patch around each keypoint is used to represent the appearance of an object part. Six region co- variances are generated for each image patch by using the pixels of the top-left, the top-right, the bottom-left, the bottom-right, the central, and all of the image patch. Then besides the offset and the class label, a code contains six region covariances. ~~When an object part is matched against the codebook, the similarity between the image patch of the object part and a code is defined by the smallest similarity of the corresponding region covariance. In this way, a codebook of object parts is built.~~All codes from all training images constitute the codebook. When an object part is matched against the codebook, the similarity between the image patch of the object part and a code is defined by the smallest similarity of the corresponding region covariance.

4.4 Detection

After forming the Hough image, the detection hypotheses are validated. Let $\mathbf{h} = \{H\}$ be the points in the Hough space which are evaluated by $C(\mathbf{x}_H, l_H)$ and have $C(\mathbf{x}_H, l_H) > 0$. Inspired by [2], the hypotheses are validated by an optimizing procedure. Let O be the number of the points in $\mathbf{h}$. ~~let~~Let $u_i = 1$ or 0 indicate H_i being a true object center or not. The problem is:

$$\arg\max_{u_i} \prod_{i=1}^{O} C^{u_i}(H_i) \Longleftrightarrow \arg\max_{u_i} \sum_{i=1}^{O} u_i \ln(C(H_i)) \ .$$

Let $v_{ij} = 1$ or 0 indicate ~~e_j~~$\mathbf{e}_j$ belongs to H_i or not, then

$$\begin{aligned} C(H_i) &= \sum_{j=1}^{M} C(\mathbf{x}_{H_i}, l_{H_i}; \mathbf{e}_j) w(\mathbf{e}_j) \\ &= \frac{1}{M} \sum_{j=1}^{M} v_{ij} C(\mathbf{x}_{H_i}, l_{H_i}; \mathbf{e}_j) \ , \end{aligned}$$

and by assuming one object part belongs to and only belongs to one hypothesis, the problem is,

$$\begin{aligned} &\arg\max_{u_i, v_{ij}} \sum_{i=1}^{O} u_i \ln(\sum_{j=1}^{M} v_{ij} C(\mathbf{x}_{H_i}, l_{H_i}; \mathbf{e}_j)) \\ &s.t. : \ u_i = 0 \text{ or } u_i = 1, \forall\ i; \\ &\quad v_{ij} = 0 \text{ or } v_{ij} = 1, \forall\ i, \forall\ j; \\ &\quad \sum_{i=1}^{O} v_{ij} = 1, \forall\ j; \\ &\quad \sum_{j=1}^{M} v_{ij} \leq u_i, \forall\ i \ . \end{aligned}$$

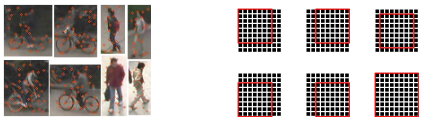
Following [2], the optimal result for the problem is given by greedy maximization. As described in Algorithm 1, the largest local maximum of all the local maxima is chosen to be the center of a true object and then the object parts belonging to the chosen object center are excluded from the object part 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 the confidence of the chosen object is lower than a threshold.

Algorithm 1 Greedy Maximization

Let ε be the set of object parts, C_{th} be the low confidence threshold to accept detection responses, and $\hat{\mathbf{h}}$ be the local maxima of $\mathbf{h}$

while $\varepsilon \neq \emptyset$ **do**
  Form $\mathbf{h}$ with ε
  Generate $\hat{\mathbf{h}}$ and select $H_i \in \hat{\mathbf{h}}, u_i = 0$ and $\forall H^{'} \in \hat{\mathbf{h}}, C(\mathbf{x}_{H_i}, l_{H_i}) >= C(\mathbf{x}_{H'}, l_{H'})$
  if $C(\mathbf{x}_{H_i}, l_{H_i}) >= C_{th}$ **then**
    $u_i \leftarrow 1$
    for $\mathbf{e}_j \in \varepsilon$ **do**
      if $\forall H^{'} \in \hat{\mathbf{h}}, C(\mathbf{x}_{H_i}, l_{H_i}|\mathbf{e}_j) >= C(\mathbf{x}_{H'}, l_{H'}|\mathbf{e}_j)$ **then**
        $v_{ij} \leftarrow 1$
        $\varepsilon \leftarrow \varepsilon - \{\mathbf{e}_j\}$
      end if
    end for
  else
    for $\mathbf{e}_j \in \varepsilon$ **do**
      $v_{1j} \leftarrow 1$
    end for
    $\varepsilon \leftarrow \emptyset$
  end if
end while
return $\{H_i, u_i = 1\}$

while $\varepsilon \neq \emptyset$ **do**
  Form $\mathbf{h}$ with ε
  Generate $\hat{\mathbf{h}}$ and select $H_i \in \hat{\mathbf{h}}$ with the largest $C(\mathbf{x}_{H_i}, l_{H_i})$
  if $C(\mathbf{x}_{H_i}, l_{H_i}) >= C_{th}$ **then**
    for $\mathbf{e}_j \in \varepsilon$ **do**
      if $\forall H^{'} \in \hat{\mathbf{h}}, C(\mathbf{x}_{H_i}, l_{H_i}|\mathbf{e}_j) >= C(\mathbf{x}_{H'}, l_{H'}|\mathbf{e}_j)$ **then**
        $\varepsilon \leftarrow \varepsilon - \{\mathbf{e}_j\}$
      end if
    end for
  else
    $\varepsilon \leftarrow \emptyset$
  end if
end while
return $\{H_i\}$



(a) (b)

Figure 4.3: (a) Training images. Note some keypoints fall on the background. (b) The manner how a 9×9 image patch is used to generate six region covariances, and red rectangles indicate the pixels used for each covariance.

Figure 4.4: Motion grouping results.

4.5 Experimental Results

In experiments, improvement of the method is verified in terms of detection accuracy. The method is tested on the P-campus dataset with [2] as benchmark, and then tested on a dataset of ~~sevral~~several animals.

4.5.1 Campus-scene Detection

Dataset The P-campus dataset contains two primary classes of foreground objects: pedestrians and bicycle riders. The frame size is 720×576. Among all the 401 continuous frames, 633 different-class ground truth bounding boxes are annotated on 79 frames. In this dataset, pedestrians and bicycle riders have in common the upper human body, and pedestrians appear in front, back, and side views.

Implementation Settings For training, 52 images of bicycle riders and 171 images of pedestrians are randomly selected. Harris corners are generated on the image, examples are given in Figure 4.3(a). For appearance, six region covariances are generated for each keypoint using the 9×9 image patch around it as shown in 4.3(b). The appearance, the offset to the image (object) center, and the label of the training image are encoded into a code, and the code is inserted into a codebook. The final codebook contains 5502 codes.

For motion grouping, each keypoint is tracked through 10 frames before and through 10 frames after the current frame. The similarity of two 21-point trajectories is defined using only ~~the overlapping part~~the frames crossed by both trajectories. To set the two thresholds for motion grouping, D_1 and D_2 are measured for keypoint pairs of different objects. D_{th}^1 is set that it is larger than only 10% of the measured D_1s, and so is D_{th}^2. By doing so, keypoints belonging to different objects are not likely to be grouped together. So that in one motion group, the keypoints are very likely to belong to the same object, as shown in Figure 4.4.

To form the Hough image, 35 best matched codes are chosen from the codebook for each object part. In (4.3), d and σ need to be given. The precision-recall curves are based on σ, while d is set to be 10. Here σ is the most important parameter.

Comparisons For comparison, detection is done on the Hough images formed with and without motion grouping results. The same codebook and the same parameter set-

(a) (b)

Figure 4.5: (a) Precision-recall curves (red: the proposed method, blue: the benchmark method). (b) Confusion matrices (upper: the proposed method, down: the benchmark method).

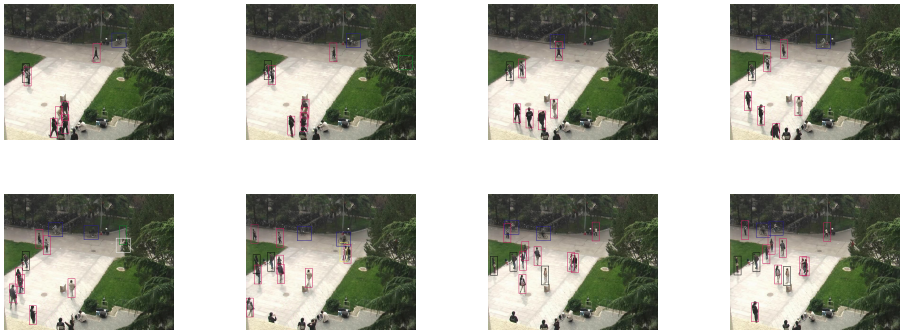


Figure 4.6: Results. Red rectangles and blue rectangles mark correctly detected pedestrians and bicycle riders. Yellow rectangles mark missed detections. White rectangles mark correctly detected but not correctly labeled objects. Green rectangles mark false alarms. Black rectangles mark static objects, which are beyond the verification for the method.

tings 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 (4.4) are assigned in the proposed method.

The precision-recall curves are shown in Figure 4.5(a). An object is considered as correctly detected only if the distance from the ground truth to it is less than 10 pixels. In Figure 4.5(a), the correctly positioned but wrongly labeled objects are considered as true positives, aiming at verifying the positioning ability of the proposed method.

The confusion matrices are given in Figure 4.5(b). For clarity of the comparisons, the proposed method is compared with the benchmark method when they have nearly equal number of false alarms. To evaluate the labeling ability, a class of "none" to represent missed detections and false alarms is manually added. For example, in Figure 4.5(b), 487 pedestrian instances are correctly positioned and labeled by the proposed method, 2 are wrongly labeled to be bicycle riders, and 21 are miss-detected. More results are shown in Figure 4.6.

4.5.2 Wild-scene Detection

Dataset ~~To show the effectiveness of our method in general cases, we prepared a more difficult dataset.~~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. Especially, the image

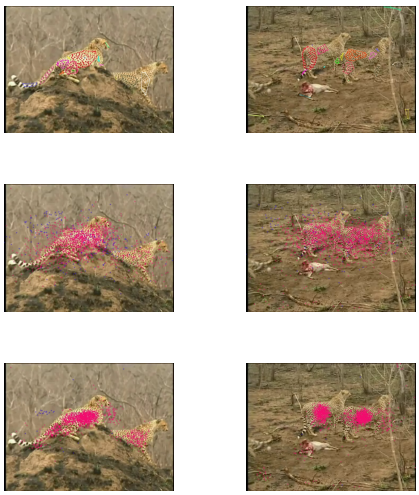


Figure 4.7: Effect of the proposed weight assignment. Red circles are voted center for leopards, while blue ones are voted centers for tigers. On the top are the motion grouping results. In the middle are the voted centers according to the best matched codes. On the bottom are the voted centers voted by votes with highest weights.

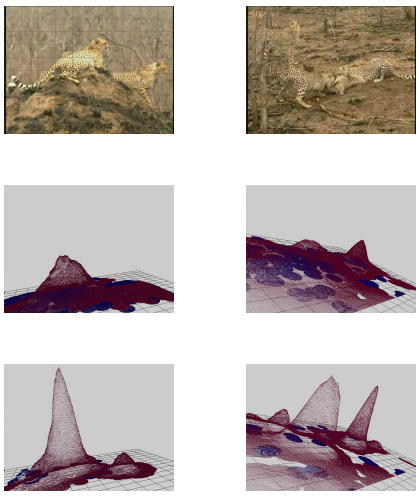


Figure 4.8: Example Hough images. On the top are the original images. In the middle are the Hough images formed by votes with uniform priors. On the bottom are the Hough images formed by votes with the proposed weights. Red indicates leopards, and blue indicates tigers. Note for the two leopards, there is no peak corresponding to the right one on the benchmark Hough image. For the three leopards, there is also no peak corresponding to the leopard in behind on the benchmark Hough image.

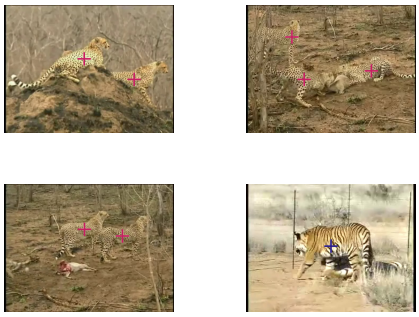


Figure 4.9: Results. Red crosses mark the centers for leopards and blue crosses mark the centers for tigers.

feature used by this method belongs to the type of texture, and texture from different ~~postions~~positions of the leopards are almost the same. The dataset contains 6 video clips of 9 leopards and 4 tigers. The frame size is 640×480. Both the animals are in the side view.

Implementation settings Most implementation settings are the same with the settings for campus object detection. For training, 5 leopards and 2 tigers are used. The size of the image patch around each keypoint is 27×27.

Comparisons In Figure 4.7, the motion grouping results and how the voted centers are affected are given. Since parts from different positions of the leopard are very similar, the true center of a leopard is difficult to find from the voted centers of the object parts. In Figure 4.8, example Hough images are given to show the merit of the proposed prior by the ability to detect leopards. In Figure 4.9, the detection results are given. The proposed method successfully localizes and labels all the leopards and tigers, while the benchmark method miss-detects three leopards.

4.6 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. ~~The~~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 our 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.

Chapter 5

Pyramid Match Score for Detection

5.1 Pyramid Match

The Pyramid Match method is designed to find the best one-one match between two point(feature) sets in a heuristic manner.

Given two point sets, $S_1 = \{u_1, u_2, ..., u_m\}$ and $S_2 = \{v_1, v_2, ..., v_n\}$, there exists a best one-one match 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 $m < n$, and π maps each feature u_i in S_1 to a unique feature $v_{\pi(i)}$ in S_2.

The best match exists, and be found by simple brute-force methods. In special cases, the Hungarian algorithm is also applicable.

Sub-optimal solution can be found by heuristic methods. A very intuitionistic method is to find matched pairs of nearest distance, exclude corresponding points from both point sets, and repeat until no pair can be found.

The methods mentioned above are not efficient enough. The Pyramid Match method is straightforward. Divide the space into very fine grids, and exclude all pairs of points, each of which exists in the same grid. Then divide the space into coarser grids, and continue to exclude matched pairs of points until no pair can be found.

However, in order to know the sum of distances between matched pairs, the distance between two matched points still needs to be calculated. In order to avoid such calculations, the "distance" is directly assigned by how fine the grids are when the two points are considered as match.

The Pyramid Match method is very efficient.

5.2 Pyramid Match Score

The Pyramid Match Score measures the confidence about one rectangle contains an object of a given type. One "point" set contains all the image features contained in the rectangle to be measured, the other "point" set is a "super template". The template is trained by inserting all features from training images.

When SIFT is used, PCA is used to do dimension reduction. And the position information of each feature can be used by adding them as two extra dimensions of the feature.

Chapter 6

Conclusion and outlook

References

[1] D.H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122, 1981.

[2] O. Barinova, V. Lempitsky, and P. Kohli. On detection of multiple object instances using hough transforms. In *CVPR*, pages 2233–2240, 2010.

[3] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *ECCV*, pages 404–417, 2006.

[4] G.J. Brostow and R. Cipolla. Unsupervised bayesian detection of independent motion in crowds. In *CVPR*, pages I: 594–601, 2006.

[5] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *ECCV*, pages V: 282–295, 2010.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In Cordelia Schmid, Stefano Soatto, and Carlo Tomasi, editors, *CVPR*, pages 886–893, June 2005.

[7] P. Davidson, J. Hautamaki, and J. Collin. Using low-cost mems 3d accelerometer and one gyro to assist gps based car navigation system. In *International Conference on Integrated Navigation Systems*, 2008.

[8] Pedro F. Felzenszwalb, David A. McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008.

[9] P.F. Felzenszwalb and D.P. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61(1):55–79, January 2005.

[10] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, pages II: 264–271, 2003.

[11] Vittorio Ferrari, Frédéric Jurie, and Cordelia Schmid. Accurate object detection with deformable shape models learnt from images. In *CVPR*, 2007.

[12] J. Gall and V. Lempitsky. Class-specific hough forests for object detection. In *CVPR*, pages 1022–1029, 2009.

[13] C.H. Gu, J.J. Lim, P. Arbelaez, and J. Malik. Recognition using regions. In *CVPR*, pages 1030–1037, 2009.

[14] C. Harris and M.J. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, pages 147–152, 1988.

[15] C. Huang, B. Wu, and R. Nevatia. Robust object tracking by hierarchical association of detection responses. In *ECCV*, pages II: 788–801, 2008.

[16] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, pages 83–97, 1955.

[17] C.H. Lampert, M.B. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR*, pages 1–8, 2008.

[18] A. Lehmann, B. Leibe, and L.J. Van Gool. Fast prism: Branch and bound hough transform for object class detection. *IJCV*, 94(2):175–197, September 2011.

[19] B. Leibe, N. Cornelis, K. Cornelis, and L.J. Van Gool. Dynamic 3d scene analysis from a moving vehicle. In *CVPR*, pages 1–8, 2007.

[20] B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *IJCV*, 77(1-3):259–289, May 2008.

[21] B. Leibe and B. Schiele. Interleaved object categorization and segmentation. In *BMVC*, pages 759–768, 2003.

[22] B. Leibe, K. Schindler, and L.J. Van Gool. Coupled detection and trajectory estimation for multi-object tracking. In *ICCV*, pages 1–8, 2007.

[23] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110, 2004.

[24] S. Maji and J. Malik. Object detection using a max-margin hough transform. In *CVPR*, pages 1038–1045, 2009.

[25] Subhransu Maji, Alexander C. Berg, and Jitendra Malik. Classification using intersection kernel support vector machines is efficient. In *CVPR*, 2008.

[26] K. Mikolajczyk, B. Leibe, and B. Schiele. Multiple object class detection with a generative model. In *CVPR*, pages I: 26–36, 2006.

[27] S. Mohottala, S. Ono, M. Kagesawa, and K. Ikeuchi. Fusion of a camera and a laser range sensor for vehicle recognition. In *OTCBVS*, pages 16–23, 2009.

[28] 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.

[29] R. Okada. Discriminative generalized hough transform for object dectection. In *ICCV*, pages 2000–2005, 2009.

[30] B. Ommer and J. Malik. Multi-scale object detection by clustering lines. In *ICCV*, pages 484–491, 2009.

[31] O. Russakovsky and A.Y. Ng. A steiner tree approach to efficient object detection. pages 1070–1077, 2010.

[32] Henry Schneiderman and Takeo Kanade. Object detection using the statistics of parts. *IJCV*, 56(3):151–177, 2004.

[33] Chris Stauffer and W. Eric L. Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, pages 2246–2252, 1999.

[34] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical report, Carnegie Mellon University, April 1991.

[35] O. Tuzel, F.M. Porikli, and P. Meer. Region covariance: A fast descriptor for detection and classification. In *ECCV*, pages II: 589–600, 2006.

[36] Paul A. Viola and Michael J. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, 2004.

[37] Z. Wang, M. Kagesawa, S. Ono, A. Banno, and K. Ikeuchi. Global 3d modeling and its evaluation for large-scale highway tunnel using laser range sensor. In *ITS World Congress*, 2012.

[38] M. Wertheimer. Laws of organization in perceptual forms (partial translation). In W. B. Ellis, editor, *A Sourcebook of Gestalt Psychology*, pages 71–88. Harcourt, Brace, 1938.

[39] C. Wojek, S. Roth, K. Schindler, and B. Schiele. Monocular 3d scene modeling and inference: Understanding multi-object traffic scenes. In *ECCV*, pages IV: 467–481, 2010.

[40] L. Xue, S. Ono, A. Banno, T. Oishi, and K. Ikeuchi. Global 3d modeling and its evaluation for large-scale highway tunnel using laser range sensor. In *ITS World Congress*, 2012.

[41] L. Yang, N. Zheng, M. Chen, Y. Yang, and J. Yang. Categorization of multiple objects in a scene without semantic segmentation. In *ACCV*, 2009.

[42] P. Yarlagadda, A. Monroy, and B. Ommer. Voting by grouping dependent parts. In *ECCV*, pages V: 197–210, 2010.

[43] T. Yeh, J.J. Lee, and T.J. Darrell. Fast concurrent object localization and recognition. In *CVPR*, pages 280–287, 2009.