Multi-view object recognition using Bag of Words approach

Carlos Miguel Correia da Costa
http://paginas.fe.up.pt/~ei09097

Department of Computer Engineering,
Faculty of Engineering, University of Porto

Abstract

Multi-view object detection and classification plays a critical role in autonomous driving vehicles, and has been an area of intense research, with several approaches to solve this kind of problem. In this paper it is analyzed the usage of the Bag of Words model to efficiently detect and recognize objects that can appear in different poses. This approach relies in feature detection, extraction and clustering to create a visual vocabulary that then is used in conjunction with a classifier to recognize the objects present in the images. To test this approach, several configurations of feature detectors, extractors and classifiers were used, and an accuracy of 87% was achieved.

1 Introduction

Multi-view object detection and recognition systems are a critical component in autonomous driving vehicles and are very useful for automation and assembly tasks. They also play a pivotal role in extracting information from images by providing the classification of the objects and their position. Given its generalization properties, this kind of systems can be adapted to a multitude of tasks, and an efficient implementation could be used in real-time applications.

Several approaches were suggested during the years, ranging from the more computer intensive solutions that compares patches of the image to a database of objects in several poses, to the more efficient techniques that uses classifiers to try to detect several variations of the target object [1] [2] [3] [4] [5]. This paper focuses on the later and aims to provide an analysis of the application of the Bag of Words model to object detection and classification, and for that it was tested with different feature detectors (SIFT [6], SURF [7], GFTT [8], FAST [9], ORB [10], BRISK [11], STAR [12] and MSER [13]), feature descriptors (SIFT, SURF, FREAK [14], BRIEF [15], ORB and BRISK), descriptor matchers (FLANN and BFMatcher) and classifiers (Support Vector Machines [16] [17], Artificial Neural Networks [18], Normal Bayes Classifier [19], Decision Trees [20], Boosting [21], Gradient Boosting Trees [22], Random Trees and Extremely Randomized Trees [23]).

In the next sections it will be presented the main algorithms used to implement the recognition system, and it will be discussed the results achieved.

2 Related work

The Bag of Words model [1] had its inception in the document classification realm, but its concepts can be extended to image recognition by treating image features as words. For that a visual vocabulary must be built from the target objects features and a classifier must be trained with samples built upon this visual vocabulary.

To detect the regions in the image where the target object is located, a sliding window technique [24] can be employed, in which several regions of interest with different sizes are tested in order to retrieve an approximate location of the target object.

With a well-trained classifier, the Bag of Words approach can be used in real time applications with very good results [3] [25]. Such good results allows its use in monitoring traffic flow, or its application in autonomous driving vehicles.

However, since the Bag of Words approach disregards the relative position of each visual word, a post processing step may be required to segment with precision the regions of interest of the target objects.

Other approaches were suggested to solve this particular problem. In the Implicit Shape Model [5], an extension of the Hough Transform was suggested to obtain a more precise description of the target object parts and as a result, better recognition precision was achieved.

Other methods use image strip features [26] to speed up recognition by focusing in structural parts of the target object or even Haar wavelets and edge orientation histograms [27].

3 Recognition system implementation

The implementation of the recognition system used OpenCV to speed up development and is comprised of several stages that will be discussed in the following subsections.

3.1 Preprocessing

To remove noise from the images and improve the detection of good feature points, a preprocessing step is applied. In this stage a bilateral filter is used and is followed by a Contrast Limited Adaptive Histogram Equalization (CLAHE) and a correction of contrast and brightness.

3.2 Visual vocabulary

In order to be able to use the Bag of Words model, a visual vocabulary of the target objects is built.

In this stage, each image in the vocabulary image list set is preprocessed and for each ground truth mask of the target objects, it is computed the feature points and their associated descriptors. These extracted descriptors are then clustered using the kmeans clustering algorithm, in order to aggregate the results and obtain the visual words of the vocabulary.

3.3 Training samples

Before a classifier can be used, it must be trained with several samples of the target objects. As such, a training database is built using the vocabulary of the visual words computed earlier.

In this stage, each image of the training set list is preprocessed, its feature points are computed and separated into the corresponding classes according to the ground truth masks and then the descriptors are built using the visual vocabulary. The results are a set of normalized histograms of the visual words present in each training image, associated with the corresponding labels, that will inform the classifier to which class the training samples belongs.

3.4 Classifier training

After having the training samples, a classifier is trained, and the result is a model that can predict with acceptable accuracy if the target objects are in an image or not.

3.5 Object recognition and location estimation

To detect the regions in the image where the target objects are, a sliding window technique was applied. In this stage, the classifier is used to analyze several patches of different sizes and locations in the image, and the result is a voting mask, that contains the locations where the classifier predicted that the target objects are likely to be.

To improve the precision of the identification of the targets regions, a threshold was applied to the voting mask, in order to discard zones that receive few votes from the classifier.

Then a blob detection algorithm was used to retrieve the bounding boxes of the targets regions.

3.6 Evaluation of results

To evaluate the results of the object recognition system, an image test set was used, in which the resulting voting masks were compared with the target objects ground truth masks.

In this stage, each pixel in the voting masks was compared to the ground truth masks, in order to see if the result was a true positive, true negative, false positive or false negative. With each of these measures acquired for each image, the accuracy, precision and recall was computed.

4 Methods used to calculate the results

The results were collected using a Clevo P370EM, with an i7-720QM CPU, NVIDIA GeForce GTX 680M GPU and 16 GB of RAM DDR3 (1600 MHz), running a Windows 8.1 x64 operating system.

It was used the Graz-02 dataset of car images, from which was retrieved 177 images to build the vocabulary and the training samples, and another 177 images for testing the recognition system.

The visual vocabularies were built with a 1000 word size, and all the intermediate results (vocabulary, training samples, and classifiers) are saved to xml files to speedup future uses of the system.

The OpenCV algorithms were used with the default parameters except the SVM classifier, in which the maximum number of iterations was set to 100000 and the Artificial Neural Networks, which were configured to have 20 neurons in the intermediate layer. Also, for binary descriptors the FLANN matcher was modified to use the multi probe LSH index search, and the BFMatcher to use Hamming distances.

The sliding window technique used 482 regions of interest per image. These patches start at 20% of the image size, and after each scan of the image, (in which the patch moves at 25% increments of its own size), the patch grows 10% (in relation to the image size).

5 Results

In appendix 1 is the detailed results that were obtained in the testing of the recognition system.

From the analysis of the results, the best accuracy (87.4%) was achieved by combining the STAR feature detector, the SIFT feature extractor, the FLANN matcher and the Artificial Neural Network classifier. This can be attributed to the superior feature description of the SIFT algorithm due to its scale and orientation invariance and to the fact that the Neural Network classifier can achieve better generalization of models.

Nevertheless, the second best accuracy result (85.5%), which was achieved with the STAR feature detector, the SURF feature extractor, the FLANN matcher and the Support Vector Machine classifier, was 5 times faster to analyze all the test images. This is greatly due to the application of a faster feature extractor (SURF) and the use of the more efficient SVM classifier (that shifted the computation time to the training stage, in which it was more than 1300 times slower than the best result, but since this is computed only once, it was an acceptable cost in the overall use of the system).

From the output of the system it can also be seen that the preprocessing stage helped in the selection of better feature points by reducing the noise and correcting the contrast and brightness. This can be seen in the figures below, in which the mud in the car was reduced and the pavement was smoothed.

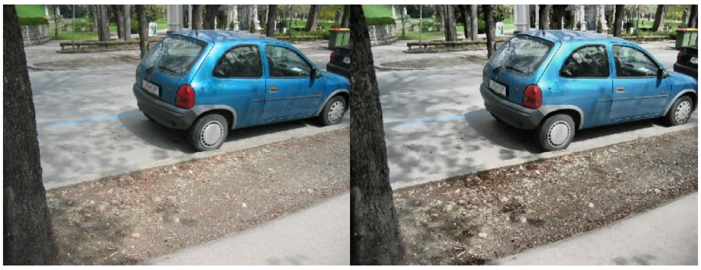

Figure 1: Effect of preprocessing (right) in the original image (left)

The final result of the recognition system also had very acceptable results. The precision of the objects bounding boxes and their voting masks can be seen in the figures below.

Figure 2: Results obtained with STAR detector, SIFT extractor, FLANN matcher and ANN classifier

Figure 3: Results obtained with STAR detector, SIFT extractor, FLANN matcher and ANN classifier

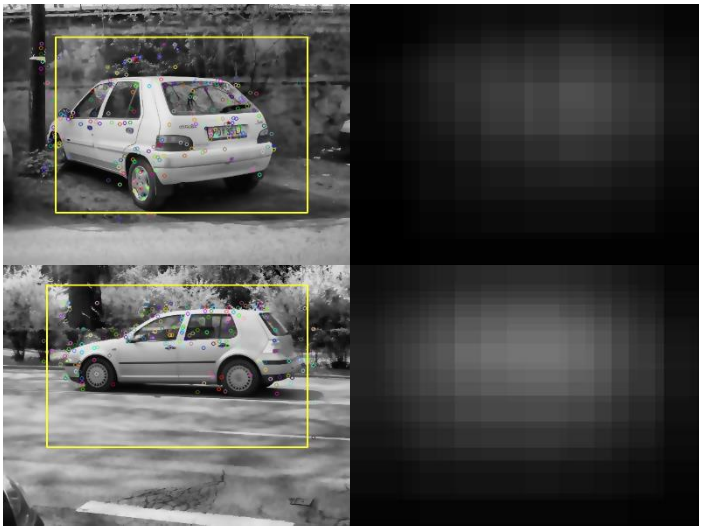

Figure 4: Results obtained with STAR detector, SIFT extractor, FLANN matcher and SVM classifier

Figure 5: Results with partially occluded objects obtained with STAR detector, SURF extractor, FLANN matcher and SVM classifier

6 Conclusions

The presented Bag of Words approach to multi-view object recognition has shown promising results and good versatility to handle different shapes of cars in different views. Its efficiency and accuracy make it a viable solution to real-time applications and its flexibility allows it to be adapted to other areas of object recognition.

References

[1] G. Csurka and C. Dance, “Visual categorization with bags of keypoints,” *Work. Stat. …*, 2004.

[2] A. Collet and S. Srinivasa, “Efficient multi-view object recognition and full pose estimation,” *Robot. Autom. (ICRA), 2010 …*, 2010.

[3] D. Jang and M. Turk, “Car-Rec: A real time car recognition system,” *… Comput. Vis. (WACV), 2011 IEEE …*, 2011.

[4] E. Nowak, F. Jurie, and B. Triggs, “Sampling strategies for bag-of-features image classification,” *Comput. Vision–ECCV 2006*, 2006.

[5] A. Thomas and V. Ferrar, “Towards multi-view object class detection,” *Comput. Vis. …*, 2006.

[6] D. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. Comput. Vis.*, 2004.

[7] H. Bay, T. Tuytelaars, and L. Van Gool, “Surf: Speeded up robust features,” *Comput. Vision–ECCV 2006*, 2006.

[8] J. Shi and C. Tomasi, “Good features to track,” *… , 1994. Proc. CVPR’94., 1994 IEEE …*, 1994.

[9] E. Rosten and T. Drummond, “Machine learning for high-speed corner detection,” *Comput. Vision–ECCV 2006*, 2006.

[10] E. Rublee and V. Rabaud, “ORB: an efficient alternative to SIFT or SURF,” *Comput. Vis. (ICCV …*, 2011.

[11] S. Leutenegger, “BRISK: Binary robust invariant scalable keypoints,” *Comput. Vis. (ICCV), …*, 2011.

[12] M. Agrawal, K. Konolige, and M. Blas, “Censure: Center surround extremas for realtime feature detection and matching,” *Comput. Vision–ECCV 2008*, 2008.

[13] J. Matas, O. Chum, M. Urban, and T. Pajdla, “Robust wide-baseline stereo from maximally stable extremal regions,” *Image Vis. Comput.*, 2004.

[14] A. Alahi, R. Ortiz, and P. Vandergheynst, “Freak: Fast retina keypoint,” *Comput. Vis. …*, 2012.

[15] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “Brief: Binary robust independent elementary features,” *Comput. Vision–ECCV 2010*, 2010.

[16] C. Burges, “A Tutorial on Support Vector Machines for Pattern.” 1998.

[17] C. Chang and C. Lin, “LIBSVM: a library for support vector machines,” *ACM Trans. Intell. Syst. …*, 2011.

[18] M. Riedmiller and H. Braun, “A direct adaptive method for faster backpropagation learning: The RPROP algorithm,” *Neural Networks, 1993., IEEE …*, 1993.

[19] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, vol. 22, no. 7. Academic Press, 1990, pp. 833–834.

[20] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, vol. 19. 1984, p. 368.

[21] J. Friedman, T. Hastie, and R. Tibshirani, “Additive logistic regression: a statistical view of boosting (With discussion and a rejoinder by the authors),” *The Annals of Statistics*, vol. 28, no. 2. pp. 337–407, 2000.

[22] J. Friedman, “Greedy function approximation: a gradient boosting machine,” *Ann. Stat.*, 2001.

[23] P. Geurts, D. Ernst, and L. Wehenkel, “Extremely randomized trees,” *Mach. Learn.*, 2006.

[24] C. Lampert, “Beyond sliding windows: Object localization by efficient subwindow search,” *Comput. Vis. …*, 2008.

[25] J. Sivic and A. Zisserman, “Video Google: A text retrieval approach to object matching in videos,” *Comput. Vision, 2003. Proceedings. …*, 2003.

[26] W. Zheng and L. Liang, “Fast car detection using image strip features,” *… Pattern Recognition, 2009. CVPR 2009 …*, 2009.

[27] D. Gerónimo, A. Sappa, A. López, and D. Ponsa, “Adaptive image sampling and windows classification for on-board pedestrian detection,” *Proc. …*, 2007.

Appendix 1: Object Recognition Results

Object Recognition Results

Feature detector	Feature descriptor	Feature matcher	Classifier	Vocabulary build time (1000 word size - 177 images)	Training samples build time (177 images)	Training samples retrieved (from 177 images)	Classifier training time	Classifier test time - 177 images (sliding window with 482 ROIs per image)	Accuracy	Precision	Recall
STAR	SIFT	FLANN	Artificial Neural Network	00m31.204s	00m44.265s	403	00m00.028s	15m14.323s	0.874	0.234	0.162
STAR	SURF	FLANN	Support Vector Machine	00m21.251s	00m17.901s	403	00m38.217s	03m02.452s	0.855	0.271	0.214
STAR	SURF	BFMatcher	Support Vector Machine	00m20.932s	00m17.985s	403	00m37.934s	03m33.083s	0.854	0.299	0.234
STAR	SIFT	FLANN	Support Vector Machine	00m31.204s	00m44.265s	403	00m36.318s	09m43.652s	0.847	0.306	0.362
STAR	BRIEF	FLANN	Support Vector Machine	00m20.131s	00m20.105s	403	00m35.184s	03m46.283s	0.841	0.276	0.277
ORB	ORB	FLANN	Artificial Neural Network	01m25.694s	00m43.962s	447	00m00.188s	17m04.451s	0.839	0.206	0.195
STAR	FREAK	FLANN	Support Vector Machine	00m20.824s	00m24.739s	403	00m36.273s	05m22.562s	0.815	0.274	0.279
SURF	SURF	FLANN	Artificial Neural Network	00m37.574s	00m35.434s	457	00m00.201s	13m03.423s	0.815	0.168	0.202
SIFT	SIFT	BFMatcher	Artificial Neural Network	01m46.338s	01m32.902s	488	00m00.234s	43m00.362s	0.794	0.217	0.296
SIFT	SIFT	BFMatcher	Support Vector Machine	01m40.631s	01m30.025s	488	00m49.265s	41m43.748s	0.784	0.242	0.385
SIFT	SIFT	FLANN	Support Vector Machine	01m46.338s	01m32.902s	488	00m50.727s	42m32.801s	0.776	0.251	0.411
ORB	ORB	FLANN	Support Vector Machine	01m25.695s	00m43.966s	447	00m44.078s	16m56.037s	0.739	0.239	0.549
SIFT	SURF	FLANN	Support Vector Machine	01m17.674s	00m43.966s	488	00m51.802s	27m05.743s	0.714	0.219	0.543
SIFT	SURF	BFMatcher	Support Vector Machine	01m11.727s	00m39.477s	488	00m50.481s	26m21.736s	0.705	0.213	0.528
GFTT	FREAK	FLANN	Support Vector Machine	01m01.011s	00m40.011s	497	00m50.479s	40m07.149s	0.699	0.201	0.478
MSER	SURF	FLANN	Support Vector Machine	00m22.772s	00m20.369s	464	00m47.321s	07m41.181s	0.672	0.241	0.735
FAST	FREAK	FLANN	Support Vector Machine	00m56.567s	01m49.256s	514	00m54.863s	51m38.865s	0.666	0.204	0.596
BRISK	BRISK	FLANN	Support Vector Machine	00m21.704s	00m30.616s	464	00m47.038s	13m12.818s	0.661	0.213	0.682
SIFT	BRIEF	FLANN	Support Vector Machine	01m03.294s	00m47.819s	488	00m48.438s	29m37.773s	0.616	0.187	0.661
SIFT	FREAK	BFMatcher	Support Vector Machine	01m08.355s	00m38.618s	488	00m49.269s	25m22.225s	0.606	0.188	0.696
SIFT	FREAK	FLANN	Support Vector Machine	01m06.325s	01m00.102s	488	00m53.349s	35m35.147s	0.605	0.191	0.717
SIFT	BRIEF	BFMatcher	Support Vector Machine	01m05.877s	00m38.599s	488	00m50.382s	25m04.586s	0.601	0.191	0.732
BRISK	FREAK	FLANN	Support Vector Machine	00m30.058s	00m29.093s	464	00m45.131s	11m03.882s	0.579	0.191	0.801
SURF	SURF	FLANN	Decision Tree	00m37.188s	00m34.271s	457	00m00.064s	18m05.666s	0.578	0.175	0.648
SURF	SURF	FLANN	Random Tree	00m37.073s	00m43.967s	457	00m00.199s	16m17.609s	0.503	0.172	0.847
SURF	SURF	FLANN	Boosting Tree	00m37.495s	00m43.962s	457	00m00956s	15m41.621s	0.499	0.171	0.845
SURF	SURF	FLANN	Extremely Random Tree	00m35.759s	00m43.969s	457	00m00.491s	18m33.911s	0.469	0.167	0.864
ORB	ORB	FLANN	Normal Bayes Classifier	01m24.585s	00m26.650s	487	00m05.779s	27m22.274s	0.446	0.165	0.886
SURF	SURF	FLANN	Gradient Boosting Tree	00m37.207s	00m43.964s	457	00m04.295s	17m23.841s	0.423	0.161	0.897
SIFT	BRISK	FLANN	Support Vector Machine	01m08.126s	01m00.105s	488	00m49.559s	45m40.242s	0.421	0.159	0.889

Accuracy: (truePositives + trueNegatives) / (truePositives + trueNegatives + falsePositives + falseNegatives)

Precision: truePositives / (truePositives + falsePositives)

Recall: truePositives / (truePositives + falseNegatives)

Measures obtained comparing the objects ground truth masks with the recognition results
Recognition results were computed using a sliding window, that gathered the most probable image regions were the target object could be located
These regions were computed with a voting mask, and the final results were thresholded to remove regions with small number of votes