Multiview object recognition using Bag of Words approach

Carlos Miguel Correia da Costa

***Abstract*—Multiview object detection and classification plays a critical role in robust image recognition systems, and can be applied in a multitude of applications, ranging from simple monitoring to advanced tracking. 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 scales, orientations and even from different perspective views. This approach relies in image analysis techniques, such as feature detection, description and clustering, in order to be able to recognize the target object even if it is present in cluttered environments. For supporting the recognition in different perspective views, machine learning techniques are used to build a model of the target objects. This model can then be employed to successfully recognize if an instance of the target object is present in an image. For pinpointing the location of the target object, a sliding window method is used in conjunction with dynamic thresholding. The recognition system was tested with several configurations of feature detectors, descriptors and classifiers, and achieved an accuracy of 87% when recognizing cars from 177 test images.**

***Index Terms*—Object recognition, multiview detection, image analysis, feature detection, feature extraction, clustering**

I. Introduction

Multiview object detection is a critical component in recognition systems and is very useful to improve the robustness of automation and assembly tasks. It also plays 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]–[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.

The system recognition setup comprises 3 main steps. Initially a visual vocabulary is built using the feature descriptor clusters of the training images. This vocabulary represents characteristic structures of the target object, and will be used as the n-dimensional descriptor space to describe an image. Using this vocabulary, a database of samples is built for training a machine learning classifier. This classifier creates a descriptor model that later can be employed to detect the target object in test images.

After the setup of the recognition system, the detection of the target objects along with their location in the image, employs a sliding window technique [6]. This technique uses the trained classifier to scan the image with windows of different size. In the end, a voting mask with the probable locations of the targets is retrieved, and in conjunction with a dynamic thresholding method, the locations are extracted.

This approach achieve promising results and can be used to recognize objects in different perspective views, even if they are in cluttered environments.

To allow the fine tuning of the system configuration, several feature detector and descriptors can be selected in conjunction with a range of machine learning classifiers.

In the following section it will be presented an overview of other approaches that can be used to perform multiview object recognition, and later it will be provided a detailed description of the implemented system. In the end, the results with the respective analysis will be discussed.

II. Related work

There are numerous approaches for detecting objects that can appear in several perspective views. Ranging from the very simple template matching to highly advanced systems with point clouds.

The most basic method to perform multiview object detection is template matching [7]. In this method, a database of images taken from several points of view is used to scan the test image and try to detect the target object. The problem of this approach is that it requires the image to be scanned with this database in several scales and orientations, which causes it to be very inefficient.

To solve the scale and orientation problem [8], feature detection and description algorithms can be used [9]. In this case, the database is only scanned once. Moreover, since the feature detection describes the image as feature points, the size of data to be compared is reduced drastically, and as a result, is orders of magnitude more efficient than template matching. In this approach, it is critical that the matching of descriptors is filtered in order to remove outliers, using for example a ratio test [8] or a homography [10] computed using RANSAC [11].

Other approaches suggest the construction of an Implicit Shape Model [5], that takes in consideration the relative position of interesting structures in the target object, in order to build a 3D representation that can then be used to recognize the intended objects.

Other methods use image strip features [12] to speed up recognition by focusing in structural parts of the target object or even Haar wavelets and edge orientation histograms [13].

For recognition of specific 3D objects, a more advanced approach using point clouds matching can be used. In this technique, 3D point clouds are matched using algorithms such as the Iterative Closest Point [14]. Besides recognizing the object, this method also allows the identification of the position of the camera in relation to the target object. However, this approach may not be suitable for general classification of objects, because it was designed to search for a particular 3D geometry. Moreover it takes considerable computation time to extract 3D point clouds from images, unless the point clouds are retrieved directly from the environment using 3D sensors such as LIDAR.

After reviewing the existing approaches, it was clear that an efficient and general multiview recognition system should employ machine learning algorithms in order to be able to successful recognize the intended category of objects without overfitting to the training data. One way to implement such system is by employing the bag of words model in conjunction with classifiers. As shown in [3], [15], this approach has promising results and good efficiency.

III. Implementation

The recognition system is comprised with a setup phase, in which a classifier is trained with samples built with a visual vocabulary, and a recognition phase, in which the classifier is used to identify new instances of the target object.

In the next sections it will be provided a detailed description of the main steps required to successful recognize categories of objects without overfitting to the training data.

The implementation of the system is available at [16]. In order to speed up development, the OpenCV library was used.

1) Preprocessing

To improve the detection of good features and ensure that the system has robust recognition even when the images have considerable noise, a preprocessing step is applied.

In a first phase, most of noise is removed using a bilateral filter [17]. This filter was chosen because it preserves the edges of the image blobs, which are very valuable structures in the detection of feature points.

After the noise is reduced, a CLAHE (Contrast Limited Adaptive Histogram Equalization) [18] is applied to increase the contrast. This can improve the recognition of the system when the images are taken in low light environments. This technique has better results over the simple histogram equalization because it can be applied to images that have areas with high and low contrast, and also limits the spread of the noise.

Finally, the brightness is adjusted to correct images that are too dark or too bright.

2) Visual vocabulary

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 feature descriptors.

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 grouped using the k-means clustering algorithm, in order to obtain the visual words of the vocabulary.

There are several algorithms to select features from images. The supported feature detectors are SIFT [8], SURF [19], GFTT [20], FAST [21], ORB [22], BRISK [23], STAR [24] and MSER [25].

For describing these features there is also several algorithms that aim to be scale and rotation invariant. The supported feature descriptors are SIFT, SURF, FREAK [26], BRIEF [27], ORB and BRISK.

The matching of these descriptors can be performed using either a brute force or a heuristic approach.

In the brute force approach, each descriptor in the image is compared with all descriptors in the reference image to find the best correspondence.

In the heuristic approach, such as the FLANN library [28], several optimizations are employed to speed up the computations. These optimizations can be related to the appropriate selection of which descriptors to match, and to the use of efficient data structures to speed up the search (such as k-d trees).

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.

4) Classifier training

After having the training samples, a classifier is trained in order to build a model of the distribution of the target object visual words descriptors. This model can then be used to predict with acceptable accuracy if the target objects are in an image or not.

There are several machine learning classifiers to perform object recognition. The included classifiers are Support Vector Machines [29], [30], Artificial Neural Networks [31], Normal Bayes Classifier [32], Decision Trees [33], Boosting [34], Gradient Boosting Trees [35], Random Trees and Extremely Randomized Trees [36].

5) Object recognition and localization 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 dynamic 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.

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 Fig. 1 is an example of a ground truth mask).

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 test image, the accuracy, precision and recall is computed.

To allow fast testing of the system, it was implemented an automatic evaluation option that analyzes all test images and collects both intermediate and final results.

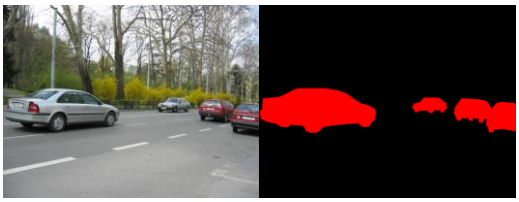

Fig. 1. Target objects ground truth masks.

IV. 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).

V. Results

Below are some representative results of the recognition of the target objects in several perspective views and in different types of environments.

The test images along with the recognition voting mask are provided. The masks are initially initialized to 0, and every time the classifier predicts that a sliding window contains the object, the mask votes in the area of the window are incremented. The computation of the bounding box of the object is achieved with a dynamic threshold based in the number of sliding windows and votes in a given mask position.

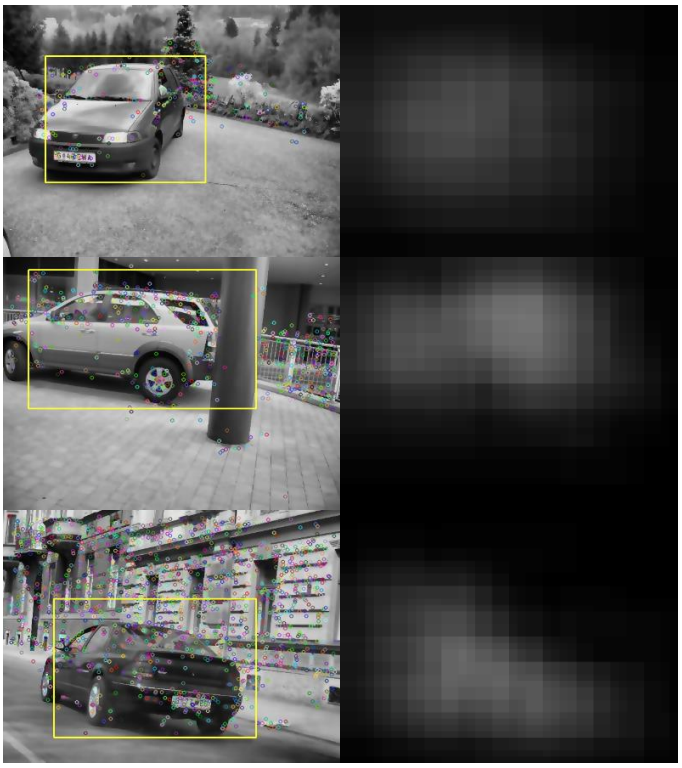

Fig. 2. Results obtained with STAR detector, SIFT extractor, FLANN matcher and ANN classifier.

Fig. 3. Results with partially occluded objects obtained with STAR detector, SURF extractor, FLANN matcher and SVM classifier.

Fig. 4. Results obtained with STAR detector, SIFT extractor, FLANN matcher and SVM classifier.

Fig. 5. Results obtained with SURF detector, SURF extractor, FLANN matcher and ANN classifier.

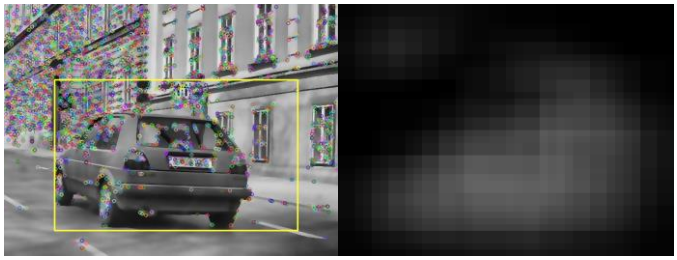

Fig. 6. Results obtained with FAST detector, SURF extractor, FLANN matcher and ANN classifier.

Fig. 7. Results obtained with ORB detector, ORB extractor, FLANN matcher and ANN classifier.

VI. Analysis of 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 Fig. 8, in which the mud in the car was reduced and the pavement was smoothed.

Fig. 8. Effect of preprocessing (right) in the original image (left).

VII. 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 for real-time applications, and its flexibility allows it to be adapted to other areas of object recognition.

The clustering of descriptors obtained with scale and rotation invariance significant contributed to the accuracy and robustness of the recognition and in conjunction with the versatility of the bag of words model, allowed the system to avoid overfitting to the training data.

These results can be further improved if a more advanced and precise location detector is used instead of the sliding window technique. This can be achieved by either replacing this method, or by considering the result as an initial step in identifying the target objects. For example, the peak in the voting mask could be used has the centroid of a more advanced segmentation technique, in order to retrieved the real location and contour of the target objects.

References

[1] G. Csurka and C. Dance, "Visual categorization with bags of keypoints," *Workshop on statistical …*, 2004.

[2] A. Collet and S. Srinivasa, "Efficient multi-view object recognition and full pose estimation," *Robotics and Automation (ICRA), 2010 …*, 2010.

[3] D. Jang and M. Turk, "Car-Rec: A real time car recognition system," *… of Computer Vision (WACV), 2011 IEEE …*, 2011.

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

[5] A. Thomas and V. Ferrar, "Towards multi-view object class detection," *Computer Vision and …*, 2006.

[6] C. Lampert, "Beyond sliding windows: Object localization by efficient subwindow search," *Computer Vision and …*, 2008.

[7] L. Cole and D. Austin, "Visual object recognition using template matching," *Australasian Conference on Robotics and …*, 2004.

[8] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, 2004.

[9] J. Ponce and S. Lazebnik, "Toward true 3D object recognition," *… de Formes et Intelligence …*, 2004.

[10] D. Baggio, S. Emami, and D. Escrivá, *Mastering OpenCV with Practical Computer Vision Projects*. 2012.

[11] M. Fischler and R. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, 1981.

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

[13] D. Gerónimo, A. Sappa, A. López, and D. Ponsa, "Adaptive image sampling and windows classification for on-board pedestrian detection," *Proceedings of the …*, 2007.

[14] P. J. Besl and H. D. Mckay, "A method for registration of 3-D shapes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 14, no. 2, pp. 239–256, 1992.

[15] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," *Computer Vision, 2003. Proceedings. …*, 2003.

[16] C. Costa, "Car-Detection." [Online]. Available: http://carlosmccosta.github.io/Car-Detection/.

[17] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pp. 839–846, 1998.

[18] P. Heckbert, *Graphics Gems: IV*. Morgan Kaufmann, 1994, p. 575.

[19] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," *Computer Vision–ECCV 2006*, 2006.

[20] J. Shi and C. Tomasi, "Good features to track," *… , 1994. Proceedings CVPR'94., 1994 IEEE …*, 1994.

[21] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," *Computer Vision–ECCV 2006*, 2006.

[22] E. Rublee and V. Rabaud, "ORB: an efficient alternative to SIFT or SURF," *Computer Vision (ICCV …*, 2011.

[23] S. Leutenegger, "BRISK: Binary robust invariant scalable keypoints," *Computer Vision (ICCV), …*, 2011.

[24] M. Agrawal, K. Konolige, and M. Blas, "Censure: Center surround extremas for realtime feature detection and matching," *Computer Vision–ECCV 2008*, 2008.

[25] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and vision computing*, 2004.

[26] A. Alahi, R. Ortiz, and P. Vandergheynst, "Freak: Fast retina keypoint," *Computer Vision and …*, 2012.

[27] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," *Computer Vision–ECCV 2010*, 2010.

[28] M. Muja and D. Lowe, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration.," *VISAPP (1)*, 2009.

[29] C. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, 1998.

[30] C. Chang and C. Lin, "LIBSVM: a library for support vector machines," *ACM Transactions on Intelligent Systems and …*, 2011.

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

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

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

[34] 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.

[35] J. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of Statistics*, 2001.

[36] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine learning*, 2006.

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