A 3D Teacher for Car Detection in Aerial Images

Stefan Kluckner, Georg Pacher, Helmut Grabner, Horst Bischof
Institute for Computer Graphics and Vision
Graz University of Technology
{kluckner, pacher, hgrabner, bischof}@icg.tugraz.at

Joachim Bauer
Microsoft Photogrammetry
Graz, Austria
jbauer@microsoft.com

Abstract

This paper demonstrates how to reduce the hand labeling effort considerably by 3D information in an object detection task. In particular, we demonstrate how an efficient car detector for aerial images with minimal hand labeling effort can be build. We use an on-line boosting algorithm to incrementally improve the detection results. Initially, we train the classifier with a single positive (car) example, randomly drawn from a fixed number of given samples. When applying this detector to an image we obtain many false positive detections. We use information from a stereo matcher to detect some of these false positives (e.g. detected cars on a facade) and feed back this information to the classifier as negative updates. This improves the detector considerably, thus reducing the number of false positives. We show that we obtain similar results to hand labeling by iteratively applying this strategy. The performance of our algorithm is demonstrated on digital aerial images of urban environments.

1. Introduction

Building an efficient and robust framework for object detection from aerial images has drawn the attention of the vision community for years, e.g. [7, 20, 26]. In particular the problem of car detection from aerial images has a variety of civil and military applications, for example transportation control, road verification, land use classification for urban planning or military reconnaissance, etc. With the availability of 3D virtual worlds on the World Wide Web, car detection in aerial images has an additional utilization: In order to provide an adequate undisturbed texture for visualization, it is crucial to remove cars from the images and the 3D model.

In recent years, boosting [21] has become a popular algorithm for the detection of objects (e.g., faces, persons, cars etc.). A variety of boosting algorithms have been developed for solving machine learning problems [4, 5, 21]. Following the remarkable success of the face detector, introduced by Viola and Jones in [23], boosting techniques have been widely used for solving different recognition problems in the computer vision community.

For classification based car detection the car model is usually created from image exemplars, which consist of gray values or texture features. Appearance models are created by collecting statistics over those features. The classifier architecture can be a single classifier, a classifier combination or a hierarchical model. The detection is done exhaustively for each image region by computing the feature vectors and classifying them against the model features. Although these approaches have certain advantages (most notably are the high recognition rates) there are also drawbacks. Notably, the feature calculation and classification is computationally expensive. Moreover and more severe, there is a need of an enormous amount of labeled data for training the detector. The training set should provide a good coverage over the space of possible appearance variations of the data. This is highly time consuming and needs human interaction to build the training data in advance and limits the possibility to diversify the variances of training samples during the training phase.

The goal of this paper is to facilitate the problem of obtaining a large number of labeled samples by using 3D information as a teacher. The basic idea is to use 3D information (obtained from a stereo matcher) to provide labels for an on-line boosting algorithm [6]. We start with a classifier that is trained from a single randomly selected car image out of a database of positive samples. Of course this classifier delivers a lot of false positives, when classifying a whole image. By using the 3D information we can select obvious negative examples (e.g. detections that lie on facades). Since we use an on-line boosting strategy the classifier can be immediately updated and applied to the image, thereby always selecting the most informative negative samples. In fact, it has been shown in the active learning community [18], that it is more effective to sample the current estimate of the decision boundary than the unknown true boundary. This is exactly achieved by our approach.

The outlined approach is similar to the work of Nair and

Clark [14] and Levin et al. [10]. Nair and Clark propose to use motion detection to obtain the initial training set and then use *Winnow* as a final classifier. Levin et al. use the so called co-training framework. The idea is to start with a small training set and to increase it by using the co-training of two classifiers operating on different features. Our approach is in spirit very similar to the conservative learning method proposed in [19]. They have demonstrated that by using a combination of generative and discriminative classifiers and a conservative update strategy a person detector can be learned in an unsupervised manner. In our work we also use such a conservative update strategy.

The paper is structured as follows: In the next section we briefly describe the data and the 3D algorithm. We review the on-line boosting method. In Section 3 we describe our approach in detail. Section 4 presents extensive experimental results comparing our algorithm to results obtained by hand labeling. Finally we conclude and present some ideas for further research.

2. Preliminaries

Before we introduce our approach, we need to discuss the two main components of the system. These are the use of 3D information, which is provided by a height model and the on-line learning of the car detector.

2.1. 3D Height Model from Aerial Images

The 3D data, that is used for learning of the image based recognition, is derived from high resolution aerial images such as ones are produced by the $UltraCam_D$ camera[1] from *Microsoft Photogrammetry*. Each image has a resolution of 11500×7500 pixels. In order to enable robust and fully automatic processing of the data, a high inter-image redundancy is ensured by capturing images at 80% along-track overlap and 60% across-track overlap. The ground sampling distance (GSD) for the digital aerial images is approximately 8 cm. The exterior orientation of the images is achieved by a fully automatic process as described in [25]. Using the camera orientation parameters an area based matching algorithm produces a dense range image for each input image. The range images are computed from three input images (a reference image and it's two immediate neighbors) using a plane sweeping approach. The plane sweeping uses the normalized cross correlation as similarity measure and produces a so-called 3D depth space which contains the depth hypotheses and their associated correlation values. The final range image is computed using a semi-global optimization approach proposed by [8]. These range images have a radiometric resolution of 16 bit. In Figure 1(a) a part of the final range image and its corresponding RGB image can be seen.

[1]http://www.vexcel.com/products/photogram/ultracam/index.html

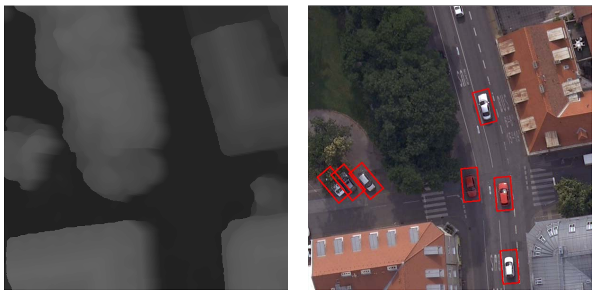

(a) (b)

Figure 1. A small part of a final range image from 3D reconstruction with a radiometric resolution of 16 bit (a), and the corresponding RGB image with ground truth overlaid as red rectangles (b).

2.2. On-line Learning for Object Detection

The basic idea is to train a binary classifier which can distinguish the class of objects from the background. Viola and Jones [23] use boosting to select a small subset from an enormous pool of simple image features to form such a classifier. Grabner and Bischof [6] use an on-line version of boosting in order to perform feature selection. This allows to continuously update the classifier when new examples are available. Thus, the classifier is able to adapt to a local scene. In the following we summarize the overall ideas of boosting, boosting for feature selection and finally the on-line variants.

Off-line Boosting for Feature Selection

Boosting is a method, which combines several weak learning algorithms to form a strong one. Many researchers have analyzed and applied boosting for different tasks. There are several variants of boosting which have been proposed (e.g. Real-Boost [5] and LP-Boost [4]). Our focus lies on the discrete AdaBoost algorithm, which was proposed by Freund and Schapire [5]. It adaptively re-weights the training samples instead of re-sampling them.

The algorithm basically works as follows: We have a training set $\mathcal{X} = \{\langle \mathbf{x_1}, y_1\rangle, ..., \langle \mathbf{x_L}, y_L\rangle \mid \mathbf{x_i} \in \mathbf{R}^m,\ y_i \in \{-1, +1\}\}$ with positive and negative labeled samples and an initial uniform distribution $p(\mathbf{x_i}) = \frac{1}{L}$ over the examples. A weak classifier h^{weak} is trained using $\mathcal{X}$ and $p(\mathbf{x})$. The weak classifier has to perform only slightly better than random guessing. Therefore, the error rate of a classifier for a binary decision task must be less than 50%. The classifier is obtained by applying a common learning algorithm. The weak classifier h_n^{weak} then gets a weight assigned $\alpha_n = \frac{1}{2} \cdot \ln\left(\frac{1-e_n}{e_n}\right)$, where e_n denotes the error of the classifier. Depending on the performance of the weak classifier, the probability $p(\mathbf{x})$ is updated. For misclassified samples, the corresponding weight is increased, while for correctly classified samples the weight is decreased. Therefore the algorithm focuses on those examples, which are difficult to learn. This process is iteratively repeated. A new weak classifier is added at each boosting iteration, until a certain stopping criterion is met.

From the obtained set of N weak classifiers $h_n^{weak}(\mathbf{x})$, a strong classifier $h^{strong}(\mathbf{x})$ is generated, by a linear combination:

$$conf(\mathbf{x}) = \frac{\sum_{n=1}^{N} \alpha_n \cdot h_n^{weak}(\mathbf{x})}{\sum_{n=1}^{N} \alpha_n} \tag{1}$$

$$h^{strong}(\mathbf{x}) = \text{sign}(conf(\mathbf{x})) \tag{2}$$

As $conf(\mathbf{x})$ is bounded by $[-1, 1]$, it can be interpreted as a confidence measure. The higher the absolute value, the more reliable is the result.

Boosting can also be applied for feature selection, as introduced by Tieu and Viola [22]. The basic idea is that features correspond to weak classifiers. By boosting, an informative subset from these features is selected.

Training for feature selection proceeds similar to the described algorithm above. From a set of possible features $\mathcal{F} = \{f_1, ..., f_k\}$. In each iteration n a weak hypothesis is built from the weighted training samples. The best one forms the weak hypothesis h_n^{weak} which corresponds to the selected feature f_n. The weights of the training samples are updated with respect to the error of the chosen hypothesis.

On-line Boosting for Feature Selection

Selecting features by boosting, as mentioned before, needs all training samples in advance, since it works off-line. In our approach an on-line feature selection algorithm [6], which is based on an on-line version of AdaBoost [17] is used. This means that each boosting step of the off-line algorithm has to be performed on-line. Therefore, the weak classifiers have to be updated every time a new training sample is available. This allows the training samples to be adapted to the current performance of the classifier, resulting in an efficient way to generate the training set.

The basic idea of on-line boosting is that the difficulty of a sample can be estimated by propagating it through the set of weak classifiers. One can think of this, as modeling the information gain with respect to the first n classifiers and code it by an importance factor for doing the update of the $n+1$-th weak classifier. As proved in [17], when given the same training set, the result of the classifier using on-line boosting converges statistically to the one obtained by off-line boosting as the number of iterations $N \rightarrow \infty$. When presented the same training set multiple times, on-line and off-line boosting achieve the same results.

For selecting features by on-line boosting, "selectors" are introduced. The on-line boosting is then not directly performed on the weak classifiers, but on the selectors. For that purpose, a selector $h^{sel}(\mathbf{x})$ consists of a set of M weak classifiers $\{h_1^{weak}(\mathbf{x}), \ldots, h_M^{weak}(\mathbf{x})\}$ and selects the one with minimal error.

$$h^{sel}(\mathbf{x}) = \arg\min_m e\left(h_m^{weak}(\mathbf{x})\right) \tag{3}$$

When training a selector, its M weak classifiers are trained and the one with the lowest estimated error is selected. Therefore, a selector can also be seen as a classifier which switches between the weak classifiers. As in the off-line case, each weak classifier corresponds to a single feature, i.e. the hypothesis generated by the weak classifier is based on the response of the feature.

The workflow of the AdaBoost on-line training framework used for feature selection is as follows: A fixed number of N selectors $h_1^{sel}, .., h_N^{sel}$ are initialized with random features. The selectors are updated, as soon as a new training sample $\langle \mathbf{x}, y \rangle$ is available, and the weak classifier with the smallest estimated error is selected. For the updating process of the weak classifier any on-line learning algorithm is applicable. Finally, the weight α_n of the sample is updated and passed to the next selector h_{n+1}^{sel}. The weight is increased if the sample is misclassified by the current selector or decreased otherwise.

A linear combination of the N selectors gives the strong classifier:

$$h^{strong}(\mathbf{x}) = \text{sign}\left(\frac{\sum_{n=1}^{N} \alpha_n \cdot h_n^{sel}(\mathbf{x})}{\sum_{n=1}^{N} \alpha_n}\right) \tag{4}$$

For more details see [6]. Contrary to the off-line version, the on-line classifier is available at any time.

Image Features

By using features instead of pixel values as input to a learning algorithm, the inter-class variability can be reduced. On the other hand the out-of-class variability can be increased. In our work we use Haar-like features [23], orientation histograms [9] and a simple version of local binary patterns [16]. By using integral images [23] as a representation, the computation of these feature types can be done very efficiently. This makes fast exhaustive template matching possible when scanning over the whole image.

Since we know the ground sampling distance of the image, the search for cars at different scales is not necessary, but cars can appear at any orientation. Instead of training the classifier with different orientations we train it at one canonical orientation, and evaluate it by rotating the image with increments of 15 degrees. Also the detector could be rotated by computing the features at different angles for

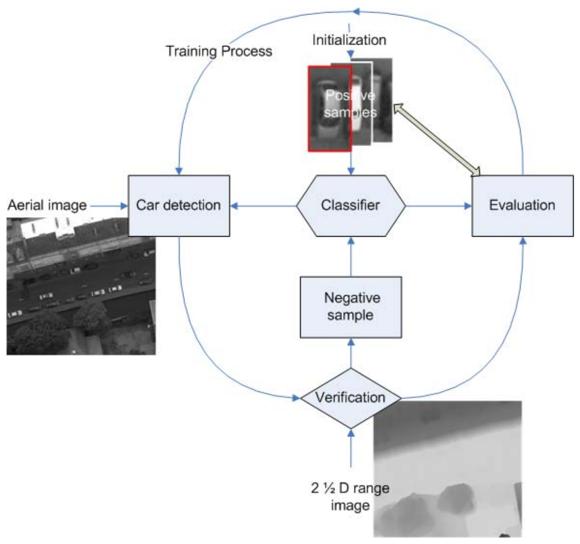


Figure 2. A schematic diagram of our proposed algorithm. After initialization with a single positive sample the iterative training process is started. The detected cars from the current classifier are verified using the range images. From this verification step, we generate negative samples, for which we are sure they are not cars, and update the classifier accordingly. After that, the classifier is evaluated on the positive samples and updated with the worst performing one, if it is not detected as car. This autonomous process is repeated until a stopping criterion is fulfilled (e.g. a maximum number of positive updates is reached).

the detection process. To speed up the process further in [11] and [12] the use of rotated Haar-like feature techniques were proposed. A trained classifier is converted to work at any angle, so rotated objects can be detected. A real-time version for the rotational invariant Viola-Jones detector has been reported in [24].

3. Autonomous Learning of a Car Detector

The main task is to train a classifier which is then used to detect objects (cars) in an image by exhaustively scanning the whole image. Our intention is to improve the classifier incrementally and to avoid hand labeling large amount of data. Therefore we use a variation of the previously described on-line boosting classifier [15], where the user labels informative samples for doing positive and negative updates.

The aim of our approach is to minimize this hand labeling effort and to make learning autonomous. Therefore, we propose to start with a set of labeled positive examples. Note, that these examples have to be labeled only once and have to cover the variances of the object, but can then be used for all new scenes. Our goal is to obtain a compact and efficient classifier for a particular scene.

A schematic diagram of our autonomous learning strategy is depicted in Figure 2. The main point is how we can autonomously generate negative samples from the image and do not introduce too much label noise (i.e. label patches which contain an object as negative sample). We provide the system with a relatively small set of 25 or 100 positive labeled images, respectively and continuously bootstrap negative samples directly from the aerial image. The classifier is initialized with a randomly drawn single positive sample. After that, the current classifier h_t^{strong} at time t is evaluated on a randomly rotated training image. This classifier provides a set $\mathcal{D}_t$ of locations where it detects cars in the image. But only a subset $\mathcal{D}_t{}^{corr} \in \mathcal{D}_t$ are correct detections, the others are false positives $\mathcal{D}_t^{false}$. We aim to identify these false positives robustly and use them as negative updates for the classifier. The verification is performed on the according 3D range data of the aerial image. We use the simple but powerful and robust assumption, that a car, if it is a car, has to be located on similar height values in the currently visited detection window. To take also important background information into account, the examined region is chosen larger than the dimension of the car detection window. Therefore, we analyze the height data in this enlarged patch window at the location of a detection and fit a plane to these values by using a robust alpha-trimming estimator [13]. From the robust estimate of the plane for the range image patch, we verify the detection on the basis of its slope. If the slope of this plane above a certain threshold, we consider this patch as a possible negative update. In each iteration, we try to find one false positive detection to retrain the classifier with a negative update. Note, if we consider a false positive as a correct detection, this does not perturb the on-line learning process. We only update false detections where we are confident that they do not correspond to cars. At the moment we cannot robustly generate new positive updates from the range image, we evaluate the classifier on all the hand labeled positive examples in each iteration of the learning process. To avoid drift, we perform a positive update of the classifier, if one sample is not correctly detected as a car. In each iteration, only one positive update is performed.
Summarizing, we have an iterative process, that can autonomously identify false positive detections and performs a negative update of our on-line learning classifier. This ensures a decrease of the false positive rate. In order to stabilize the true positive rate we evaluate the positive hand labeled samples. The set of positive samples is not updated yet. The sketch of the autonomous on-line training process is outlined in Algorithm 1.

After the training process is finished, the detection is performed by applying the trained classifier exhaustively on test images. To detect various possible orientations of the cars in the aerial images, we rotate the test images in steps of 15 degrees. If the classifier returns a confidence value for a patch above a certain activation threshold, this patch is

Algorithm 1 Autonomous On-line Training Process

```
Initialize parameters for the classifier;
while Non-Stop-Criteria do
    Evaluate the current classifier;
    3D Teacher: Verify the detections using 3D data;
    Perform a negative update if the verification fails;
    Evaluation of the positive hand labeled samples;
    if Positive sample is detected as false negative then
        Perform a positive update with this sample;
    end if
end while
```

considered as detected. The lower the threshold, the more likely an object is detected as a car but on the other hand the more likely a false positive occurs. For a higher activation threshold the false positives decrease at the expense of the detections. The detection process computes a high number of detections, some have overlap and various orientations. Therefore a post processing stage is needed to refine and combine these outputs. In [15] a mean shift based clustering is proposed. In our approach we apply a common non-maximum suppression technique to compute local confidence extrema that describe the detections and the according orientations.

4. Experiments

The aim of our experiments is to demonstrate the robustness of our framework for car detection from aerial images. Therefore we compare our autonomously trained classifier with the interactive approach presented in [15]. For a quantitative evaluation, we use recall-precision curves (RPC) [1]. $\#TP$ describes the number of true positives, $\#FP$ is the number of false positives. $\#nP$ defines the total number of cars extracted from the ground truth. The precision rate (*PR*) shows the accuracy of the prediction of the positive class. The recall rate (*RR*) shows how many of the total number of positive samples we are able to identify. The F-Measure (*Fm*) is the harmonic mean. It can be considered as trade-off between recall rate and precision rate. For evaluation of our detector, we plot the recall rate against $1 - precision$.

$$PR = \frac{\#TP}{\#TP + \#FP} \quad (5)$$

$$RR = \frac{\#TP}{\#nP} \quad (6)$$

$$Fm = \frac{2 \cdot RR \cdot PR}{RR + PR} \quad (7)$$

Note that we define a correct detection, if and only if the center of the detection corresponds to the annotated ground truth car with a maximum city block distance of approximately $1.8m$ (22 pixels in the examples). The computation of the non-maximum suppression is performed with the same value for the local neighborhood. In addition we require that the orientation of the detection has to match within 16 degrees. In contrast to [15], we use the detected orientation additionally to the location as a criterion for the evaluation in our results. For all experiments these values are kept constant.

4.1. Data Set

For evaluation of our proposed car detector, we use aerial images of the city center of Graz, Austria. The images were acquired by the $UltraCam_D$ camera developed by *Microsoft Photogrammetry*.
As we know the ground sampling distance of the aerial images (see Section 2.1), we can specify a fixed sized rectangle which reflects the size of a typical car. The size of the patch has to be carefully chosen to cover the area, which contains a car in the middle and four small surrounding bands. This is done in order to include some context information of a car so that the car is considered together with its surrounding background. Usually the boundary of a car is a rectangle with the length twice its width. In our case, we have chosen the patch size to be 35×70 pixels or 2.8×5.6 meters, respectively.
In Figure 1(b) a typical test image with the ground truth data overlaid as red rectangles is shown.
For the training and testing process, we use two non-overlapping aerial images. The positive hand labeled samples are extracted from the hand labeling learning process.

4.2. Improving the Detector

We start with an untrained classifier, which is initialized with a single positive sample. The classifier improves online after evaluation and verification by using our proposed approach. During this training process we autonomously generate negative training samples. The positive hand labeled data sets contain 25 and 100 samples, respectively. These positive images samples should cover the main appearance of cars. In Figure 4 subsets of the positive and negative samples that are used for updating are shown. The hand labeling training process takes about 2 hours of permanent human interaction. The learning process is stopped after 220 positive updates.
Both, the interactive and autonomous learning strategy result in an increasing number of positive and negative updates. In case of autonomous learning the number of possible negative updates is slightly decreasing during the improvement of the detector, because the number of false positive detections is decreasing over training time.
In Figure 3 the performance of the autonomous training process at different stages of the learning process is given.

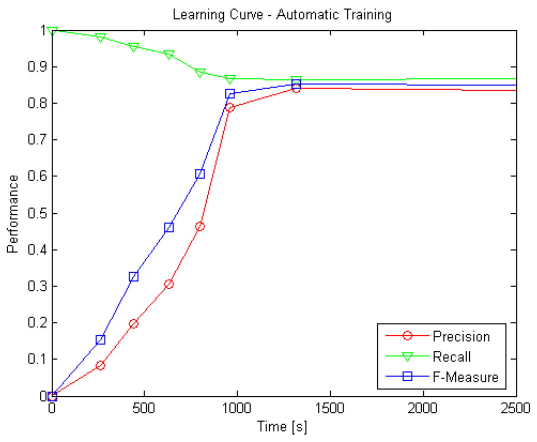


Figure 3. Performance of the autonomously trained classifier versus training time.

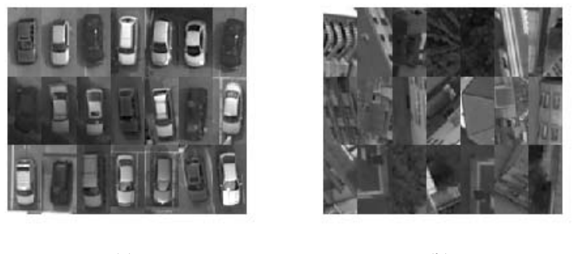


Figure 4. The positive (a) and negative (b) samples used for updating the classifier. The positive samples are hand labeled and given in advance, while the negative samples are generated autonomously.

The performance improvement over training time of the trained classifier can be seen. In contrast to manual training we can train our classifier without any human interaction apart from the construction of the set of positive samples. Comparing our learning curve to the results given in [15], we obtain a similar performance in a fractional amount of time. Note that for a negative update of the classifier the slope of the fitted plane to the height values is set to be higher than 10 degrees. This threshold gives a trade-off between finding negative samples and the probability that a car is wrongly used as a negative update. Furthermore it guarantees a conservative update strategy, since we rather make no update than using a wrong one.

4.3. Performance Evaluation

In this section we show quantitative results of our approach. The classifier is evaluated on sub-images of a size 4500×4500 pixels. These sub-images are extracted from the data set described in Section 4.1. In a previous hand labeling phase, we marked all cars in each sub-image to generate a ground truth. The ground truth set includes the locations and orientations of 423 cars. As shown in Figure 1(b) and 7, the images contain complex backgrounds, cars with low contrast and cars occluded by buildings or vegetation and there are objects which look like cars. We marked a car as positive if more than 50% of its area is visible.

For detection we use a 90% overlapping patch grid on each sub-image. Additionally these sub-images are rotated in steps of 15 degrees to get results for each location and orientation. After that, we apply a non-maximum suppression as a post processing step on the obtained detection results.

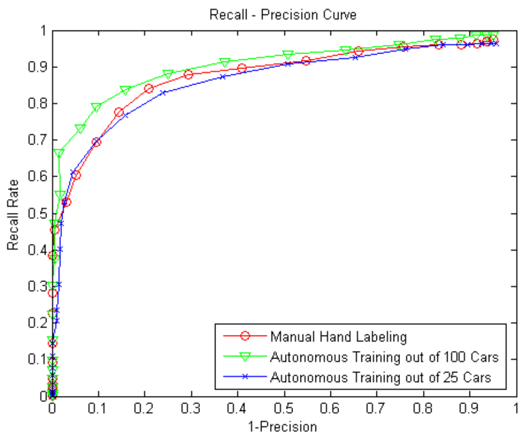


Figure 5. Recall-Precision Curves comparing the results of a manually trained classifier with our autonomously trained one after a constant number of positive updates. For the autonomous learning 25 and 100 various positive hand labeled samples were used.

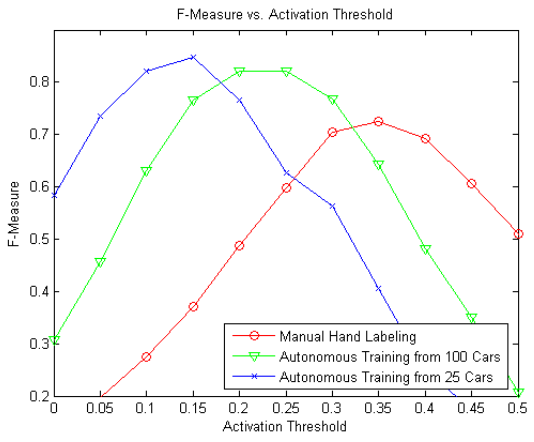
Figure 5 shows the resulting recall-precision curves of the trained classifiers with a constant number of positive updates. For comparison with the manually learned classifier we stopped our autonomous learning process after 220 positive updates.

We compare our detection results to the results of the hand labeling approach using sets of 25 and 100 positive labeled cars. The plots show that our autonomous learning strategy reaches similar performance compared to hand labeled training. This is explained by the fact that the autonomous approach uses important negative updates to improve the classifier at each iteration.

Again, note that learning the classifier by hand labeling needs approximately 2 hours of human interaction. It is obvious that the negative update is drawn more or less in a random manner. In this case the improvement of the detector is decelerated.

The autonomous car detector improves its ability to detect cars fast and without any human interaction. In terms of the detection rate, we achieve slightly worse results than the presented rates in [15], because we additionally verify the correct orientation for each detection.

In Figure 6 the F-Measure versus the activation threshold is given. By using only 25 cars as possible positive updates,



(a)

Figure 6. F-Measure versus the activation threshold for manual and autonomous learning.

we obtain the lowest activation threshold. The hand labeling process results in a high activation threshold. The lower the number of samples in the positive car data set, the lower the activation threshold. This results from the relatively low variance of the positive hand labeled set. To obtain the detection rates of hand labeled training, the number and the variance of samples respectively for positive updates have to be increased significantly.
By using context information such as street layers, the detection rates of our system could certainly be improved.
As mentioned before, cars are undesired for large scale 3D reconstruction of urban environments. Therefore we remove the detected cars from the aerial images with an in-painting approach based on Total Variation models [3]. The in-painting results of the test images are shown in Figure 7.

5. Conclusion and future work

We have proposed a framework for learning a car detector for aerial images autonomously by making use of additional 3D data. The on-line boosting technique offers a fast update and an evaluation of the current classifier at any time.
By iteratively updating the on-line classifier with previously labeled positive samples and automatically generated negative samples using range images, the amount of human interaction can be minimized drastically.
We have also shown, that only a small set (25 samples) of hand labeled positive examples is sufficient for training. Our approach obtains similar results as the hand labeling process, which suffers from an enormous effort of human interaction. For future work, our approach will be extended to use additional context information like street layer or various area classification results. An on-line boosting approach based on rotational invariant features would improve

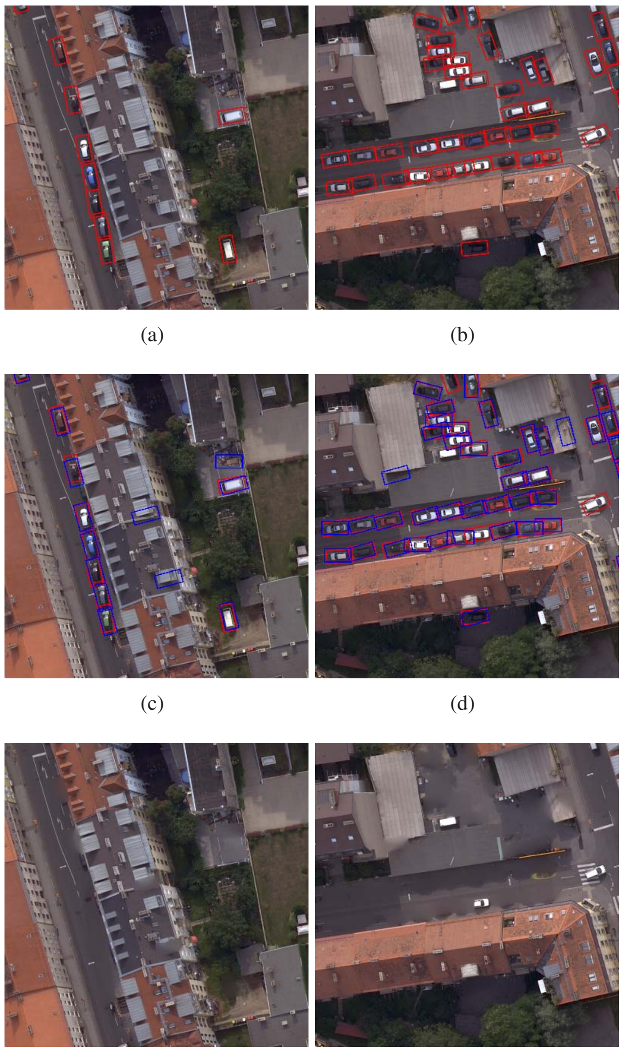


Figure 7. In (a) and (b) parts of our test images are shown. The ground truth is indicated by red rectangles around the cars. The detection results of our classifier (blue) are compared to the ground truth in (c) and (d). The classifier was autonomously trained with 220 positive and 217 negative updates. The in-painting results are shown in (e) and (f).

the training and detection time.
An elaborated in-painting strategy (e.g. [2]) will give improved results if cars are detected on complex backgrounds, such as road signs and curbs, or if the detections are partially occluded by trees and facades.
Moreover, the redundancy due to high overlap in the aerial images can be exploited. A major next step is to include also positive updates. Following venues are currently utilized: Starting from a small set of hand labeled samples and using the available multi-view images, we can generate positive update samples by using new detected cars in other images. We are also exploring co-training strategies by us-

ing extracted height field features to construct a shape based car model out of the proposed appearance driven detection process. Using both models would achieve a better performance and generalization on various aerial image data sets.

Acknowledgements

This work has been supported by the APAFA Project No. 813397, financed by the Austrian Research Promotion Agency (http://www.ffg.at) and the Virtual Earth Academic Research Collaboration funded by Microsoft.

References

[1] S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1475–1490, 2004. 5

[2] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher. Simultaneous structure and texture image inpainting. *IEEE Transactions on Image Processing*, 12(8):882–889, August 2003. 7

[3] T. F. Chan and J. Shen. Mathematical models for local nontexture inpaintings. *SIAM Journal on Applied Mathematics*, 62(3):1019–1043, 2002. 7

[4] A. Demiriz, K. Bennett, and J. Shawe-Taylor. Linear programming boosting via column generation. *Machine Learning*, 46:225–254, 2002. 1, 2

[5] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997. 1, 2

[6] H. Grabner and H. Bischof. On-line boosting and vision. In *Proceedings IEEE Conference Computer Vision and Pattern Recognition*, volume 1, pages 260–267, 2006. 1, 2, 3

[7] S. Hinz. Detection and counting of cars in aerial images. In *International Conference on Image Processing*, volume 3, pages 997–1000, 2003. 1

[8] A. Klaus, M. Sormann, and K. Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. *Proceedings of International Conference on Pattern Recognition*, pages (III):15–18, 2006. 2

[9] K. Levi and Y. Weiss. Learning object detection from a small number of examples: The importance of good features. In *Proceedings IEEE Conference Computer Vision and Pattern Recognition*, pages 53–60, 2004. 3

[10] A. Levin, P. Viola, and Y. Freund. Unsupervised improvement of visual detectors using co-training. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 626–633, 2003. 2

[11] R. Lienhart and J. Maydt. An extended set of haar-like features for object detection. In *Proceedings International Conference on Image Processing*, pages 900–903, 2002. 4

[12] A. L. C. Marczack, M. J. Johnson, and C. H. Messom. Real-time computation of haar-like features at generic angles for detection algorithms. Research Letters in the Information and Mathematical Sciences - ISSN 1175-2777, Vol. 9, 2005. 4

[13] H. Myler and A. Weeks. *The Pocket Handbook of Image Processing Algorithms in C*. Prentice Hall, 1993. 4

[14] V. Nair and J. J. Clark. An unsupervised, online learning framework for moving object detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume II, pages 317–324, 2004. 2

[15] T. Nguyen, H. Grabner, B. Gruber, and H. Bischof. On-line boosting for car detection from aerial images. In *IEEE International Conference on Research, Innovation and Vision for the Future (RIVF07)*, pages 87–95, 2007. 4, 5, 6

[16] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002. 3

[17] N. Oza and S. Russell. Online bagging and boosting. In *Proceedings Artificial Intelligence and Statistics*, pages 105–112, 2001. 3

[18] J.-H. Park and Y.-K. Choi. An on-line PID control scheme for unknown nonlinear dynamic systems using evolution strategy. In *International Conference on Evolutionary Computation*, pages 759–763, 1996. 1

[19] P. Roth, H. Grabner, D. Skocaj, H. Bischof, and A. Leonardis. Conservative visual learning for object detection with minimal hand labeling effort. In W. Kropatsch, R. Sablatning, and A. Hanburry, editors, *Pattern Recognition 27th DAGM Symposium*, volume LNCS 3663, pages 293–300. Spinger, 2005. 2

[20] R. Ruskone, L. Guigues, S. Airault, and O. Jamet. Vehicle detection on aerial images: A structural approach. In *International Conference on Pattern Recognition*, volume 3, pages 900–904, 1996. 1

[21] R. Schapire. The boosting approach to machine learning: An overview. In *Proceedings MSRI Workshop on Nonlinear Estimation and Classification*, 2001. 1

[22] K. Tieu and P. Viola. Boosting image retrieval. In *Proceedings IEEE Conference Computer Vision and Pattern Recognition*, pages 228–235, 2000. 3

[23] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings IEEE Conference Computer Vision and Pattern Recognition*, volume I, pages 511–518, 2001. 1, 2, 3

[24] B. Wu, H. Ai, C. Huang, and S. Lao. Fast rotation invariant multi-view face detection based on real adaboost. In *Proceedings International Conference on Automatic Face and Gesture Recognition*, pages 79–84, 2004. 4

[25] L. Zebedin, A. Klaus, B. Gruber-Geymayer, and K. Karner. Towards 3d map generation from digital aerial images. *International Journal of Photogrammetry and Remote Sensing*, 60:413–427, Sept. 2006. 2

[26] T. Zhao and R. Nevatia. Car detection in low resolution aerial image. In *International Conference on Computer Vision*, 2001. 1