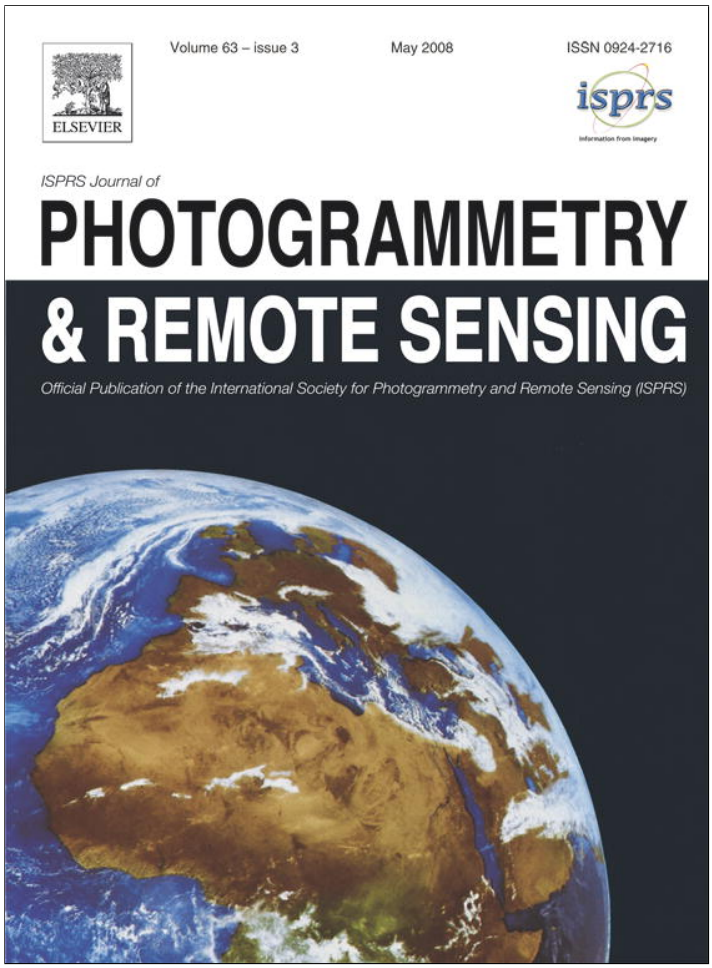

Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
ISPRS Journal of Photogrammetry & Remote Sensing 63 (2008) 382–396
PHOTOGRAMMETRY & REMOTE SENSING
www.elsevier.com/locate/isprsjprs

On-line boosting-based car detection from aerial images

Helmut Grabner [a], Thuy Thi Nguyen [a,*], Barbara Gruber [b], Horst Bischof [a]

[a] *Institute for Computer Graphics and Vision (ICG), Graz University of Technology, Inffeldgasse 16, A-8010 Graz, Austria*
[b] *VRVis Research Center for Virtual Reality and Visualization, Graz, Austria*

Received 23 November 2006; received in revised form 22 October 2007; accepted 26 October 2007
Available online 25 January 2008

Abstract

Car detection from aerial images has been studied for years. However, given a large-scale aerial image with typical car and background appearance variations, robust and efficient car detection is still a challenging problem. In this paper, we present a novel and robust framework for automatic car detection from aerial images. The main contribution is a new on-line boosting algorithm for efficient car detection from large-scale aerial images. Boosting with interactive on-line training allows the car detector to be trained and improved efficiently. After training, detection is performed by exhaustive search. For post processing, a mean shift clustering method is employed, improving the detection rate significantly. In contrast to related work, our framework does not rely on any priori knowledge of the image like a site-model or contextual information, but if necessary this information can be incorporated. An extensive set of experiments on high resolution aerial images using the new UltraCamD shows the superiority of our approach.

Keywords: Car detection; Aerial image; Adaboost; On-line learning; Pattern recognition; UltraCamD

1. Introduction

Building an efficient and robust framework for object detection from aerial images has drawn the attention of research community in computer vision for years (e.g., Ruskone et al., 1996; Rajagopalan et al., 1999; Zhao and Nevatia, 2003; Hinz, 2003; Alba-Flores, 2005). An aerial image contains a lot of objects with a complicated background of the urban scene. The UltraCamD camera from Microsoft-Vexcel can deliver large format panchromatic images as well as multi spectral images (Leberl et al., 2003). The high resolution images have a size of 11,500 pixels across-track and 7500 pixels along-track. Thus, a panchromatic image has a size of 84 MB and a RGB or NIR (near infrared) image has a size of 252 MB. These large images need automatic methods for efficient processing.

Car detection from aerial images has a variety of civil and military applications, such as transportation control, road verification to support land use classification for urban planning, military reconnaissance, etc.

Aerial images are usually taken from vertical direction. Although with some constraints on the viewpoint, the appearance of the cars in the image is varying widely. Cars appear as small objects, which vary in intensity and many details are not visible. Depending on the resolution a typical car has a size

* Corresponding author.
E-mail addresses: hgrabner@icg.tu-graz.ac.at (H. Grabner), thuy@icg.tu-graz.ac.at (T.T. Nguyen), gruber@vrvis.at (B. Gruber), bischof@icg.tu-graz.ac.at (H. Bischof).

0924-2716/$ - see front matter © 2007 International Society for Photogrammetry and Remote Sensing, Inc. (ISPRS). Published by Elsevier B.V. All rights reserved.
doi:10.1016/j.isprsjprs.2007.10.005

between 13 and 26 pixels (Zhao and Nevatia, 2003). The appearance of cars may have parts occluded by the shadow of buildings or trees, or may be dominated by the shadow of the car. Moreover, the urban scene comprises a complicated background with a variety of objects that look like cars such as windows, roofs, corners of streets, or buildings. All these issues make it difficult to characterize the features of a car and impose challenges in recognition of cars from aerial images. Therefore, although a lot of efforts have been made, it is still an open problem to build an efficient and robust algorithm for automatic car detection from aerial images.

In recent years, boosting, a machine learning method, has become popular. Referring to the overview given in Schapire (Schapire, 2003), boosting has been used for text recognition, routing, medical diagnostic, segmentation, etc. Various boosting frameworks have been developed for solving machine learning problems (Schapire, 2003; Demiriz et al., 2002; Freund and Schapire, 1997; Stojmenovic, 2006). Following the remarkable success of the face detector introduced by Viola and Jones (Viola and Jones, 2001), boosting techniques have been widely used for different problems in the computer vision community. The detection problem is formulated as a binary classification problem, discriminating the object from the background. The learned classifier is evaluated on the whole image. In order to speed up the exhaustive search, in the classical work of Viola and Jones (2001) integral images were employed, which allow very fast computation of simple image features for object representation. Additionally, a cascade structure makes the detector simultaneously fast and accurate. This framework allows to proceed efficiently on large image data and has been successfully applied for various object detection problems.

Most of the above work uses Adaboost for the detection of objects in terrestrial images. None of them (up to our knowledge) uses boosting methods for object (car) detection from aerial images. In this paper, we propose a robust boosting-based system for car detection from aerial images. The main goal is high quality detection by using novel machine learning methods with an efficient training mechanism.

First, we use boosting and particularly an efficient integral image representation for fast calculation of cars' features. In addition to the commonly used Haar wavelets (Viola and Jones, 2001), we employ local orientation histograms (Dalal and Triggs, 2005) and local binary patterns (Ojala et al., 2002) as features.

Second, we use a novel on-line version of Adaboost to train the detector. It performs on-line updating on the ensembles of features during the training process. By on-line training, we can update the classifier as new samples arrive, and therefore we can minimize the tedious work of hand labeling of training samples.

The developed framework results in a robust and automatic car detection system from aerial images achieving high performance. The system is flexible since it does not require any site-model or contextual knowledge or other information influencing the appearance of cars in images.

The paper is organized as follows. Section 2 gives a brief review of related work. Section 3 presents our approach for car detection from aerial images. Section 4 is dedicated to experiments and results. It also discusses the suitability data delivered by UltraCamD to integrate our system with related applications. Finally, Section 5 ends up with discussion and future work.

2. Related work

Recently, a lot of research has been dedicated to object recognition using machine learning methods (e.g, Papageorgiou and Poggio, 2000; Schneiderman and Kanade, 2000; Heisele et al., 2006; Bernstein and Amit, 2005). Related work on car detection can be roughly divided into two groups of approaches according to the type of modeling: explicit and implicit (Hinz, 2003).

Explicit modeling uses a generic car model (Zhao and Nevatia, 2003; Moon et al., 2002; Hinz, 2003; Schlosser et al., 2003; Hinz and Stilla, 2006). A car is represented as a 2D or 3D model representing the shape of cars, e.g. by a box or wire-frame representation. Prominent geometric features of cars are used on different levels of detail. In the detection stage, image features are extracted and grouped to construct structures similar to the model. Mainly used features are rectangles of car boundaries or the front windshields. Additionally, radiometric features such as intensity of shadow or color can also be employed. Car detection is done by grouping extracted image features "bottom–up" or by matching the model "top–down" to the image. The car object is considered to be detected if there is sufficient evidence for the model in the image. This approach relies mainly on geometric features such as edges, lines and areas to construct a hierarchical structure. One to the ground resolution of aerial images, in the decimeter range, the models cannot be very detailed because the features would not be detectable. On the other hand, generic and simple models have the inherent danger of fitting to too many positions in the image, therefore not being discriminative enough.

In an implicit or appearance-based approach, the car model is created by example images of cars and consists

of gray value or texture features. Appearance models are generated by collecting statistics over these features. For car detection in terrestrial images, some part or component based models have been proposed (Bileschi et al., 2004; Heisele et al., 2006; Bernstein and Amit, 2005; Leibe et al., 2004). The classifier architecture can be a single classifier, a combination of classifiers or a hierarchical model for classification. Support vector machines were mainly used (Rajagopalan et al., 1999; Schneiderman and Kanade, 2000; Papageorgiou and Poggio, 2000; Heisele et al., 2006; Bernstein and Amit, 2005). For image regions, the detection is done by computing feature vectors and classifying them according to the model features. Although these approaches have certain advantages, there also exist drawbacks: Feature calculation and classification are computational expensive. Moreover, there is a need for a huge amount of labeled data for training the detector. The training set should provide a good coverage of the space of possible appearance variations of the data. This needs a lot of time and labor to build a representative training set.

Co-training versions of the classifier have been proposed to deal with this problem for object detection and classification (Javed et al., 2005; Levin et al., 2003; Nair and Clark, 2004; Roth et al., 2005; Abramsol and Freund, 2005). The on-line strategy aims at reducing the manual labeling effort and makes possible to increase variability of the training data, while progressively improving the classifier.

Most related approaches attempt to match the model with the images to detect/recognize appearances of cars. Additionally, some methods limit the search area by taking into account site-model or contextual knowledge (Zhao and Nevatia, 2003; Moon et al., 2002). For example, cars are only searched on known roads or parking lots. No method has yet explored the power of state-of-the-art machine learning methods, such as Adaboost, for adaptively and efficiently training a car detector for large-scale aerial images.

3. On-line boosting for car detection

We propose an on-line boosting-based framework for car detection from aerial images based on implicit appearance-based models. The main contribution of this paper is an on-line boosting algorithm for car detection in aerial images. On-line training avoids the need for a huge pre-labeled training set. Moreover, efficient data structure allows a fast feature calculation thereby enabling interactive training and classification on the large aerial images.

First, we summarize the boosting method which will be used for feature selection. The active training process, which allows efficient on-line learning, is described afterwards. Then, the features used for classification are discussed. Car detection is performed by applying the trained classifier over all possible locations and rotations of the image. Exhaustive search in an image is possible because we use efficient data structures. Finally, a post processing stage using the mean shift clustering technique is presented to improve detection rate. In addition, we show how context information can be used for further improvement of detection results.

3.1. Boosting

In general, boosting converts (boosts) a weak learning algorithm into a strong one. Boosting has been analyzed carefully and tested empirically by many researchers (e.g., Schapire et al., 1997. Various variants of Boosting have been developed, e.g. Real-Boost (Freund and Schapire, 1997), LP-Boost (Demiriz et al., 2002). We focus on the discrete AdaBoost (adaptive boosting) algorithm introduced in Freund and Schapire (1997). It adaptively reweights the training samples instead of re-sampling them. The basic algorithm works as follows: Given a training set $\mathcal{X} = \{\langle \mathbf{x}_1, y_1\rangle, \ldots, \langle \mathbf{x}_L, y_L\rangle | \mathbf{x_i} \in \mathrm{R}^m, y_i \in \{-1, +1\}\}$ with positive and negative labeled samples and an initial uniform distribution $p(\mathbf{x_i}) = \frac{1}{L}$ over the examples. Based on χ and $p(\mathbf{x})$, a weak classifier h^{weak} is trained. A weak classifier is a classifier that has to perform only slightly better than random guessing, i.e., for a binary decision task, the error rate must be less than 50%. The classifier is obtained by applying a learning algorithm, e.g. applying statistical learning for a decision stump. Based on the error e_n the weak classifier $h_n{}^{\mathrm{weak}}$ gets assigned a weight $\alpha_n = \frac{1}{2} \cdot \ln\left(\frac{1-e_n}{e_n}\right)$. The probability $p(\mathbf{x})$ is updated such that it increases for the samples that are misclassified. The corresponding weight is decreased if the sample is classified correctly. Therefore, the algorithm focuses on the difficult examples. At each boosting iteration a new weak classifier is added and the process is repeated until a certain stopping condition is met (e.g. a given number of weak classifiers are trained). Finally, a strong classifier $h^{\mathrm{strong}}(\mathbf{x})$ is computed as linear combination of a set of N weak classifiers $h_n{}^{\mathrm{weak}}(\mathbf{x})$:

$$h^{\mathrm{strong}}(\mathbf{x}) = \mathrm{sign}(\mathrm{conf}(\mathbf{x})), \tag{1}$$

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

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

Freund and Schapire (1997) proved strong bounds on the training and generalization error of AdaBoost. For the case of binary classification the training error drops exponentially fast with respect to the number of boosting rounds N, i.e., number of weak classifiers. Schapire et al. (1997) and Rudin et al. (2004) showed that boosting maximizes the margin and proved that larger margins for the training set are translated to superior upper bounds on the generalization error.

3.2. Boosting for feature selection (combination)

Boosting for feature selection was first introduced by Tieu and Viola (2000). In their work, feature selection from a large set of features is done by Adaboost. The main idea is that each feature corresponds to a single weak classifier and boosting selects an informative subset from these features.

Training proceeds similar to the described boosting algorithm. Given a set of possible features $\mathcal{F}=\{f_1,\ldots,f_k\}$ in each iteration step n the algorithm builds a weak hypothesis based on 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 hypotheses. Finally, a strong classifier h^{strong} is computed as a weighted linear combination of the weak classifiers, where the weights α_n are estimated according to the errors of h_n^{weak} as described above.

3.3. On-line boosting for feature selection

Boosting for feature selection as described above works off-line. Thus, to train a classifier, all training samples must be given in advance. In our work we use a novel on-line feature selection algorithm (Grabner and Bischof, 2006) based on an on-line version of AdaBoost (Oza and Russell, 2001b,a). This allows to adaptively train the detector and efficiently generate the training set. First, we briefly summarize on-line boosting. Second, we discuss how it can be used for feature selection.

The basic idea of on-line boosting is that the importance or difficulty of a sample can be estimated by propagating it through a set of weak classifiers. One can think of this as modeling the information gain with respect to the first n classifier and code it by the importance weight λ (initialized by 1) for doing the update of the $n+1$-st weak classifier. Oza and Russell (2001a,b) have proved that, if off-line and on-line boosting are given the same training set, then the weak classifiers returned by on-line boosting converges statistically to the one obtained by off-line boosting as the number of iterations $N\rightarrow\infty$. Therefore, for repeated presentation of the training set, on-line boosting and off-line boosting give the same result. In our framework, on-line boosting for feature selection is based on introducing "selectors" and performing on-line boosting on these selectors and not directly on the weak classifiers.

Each selector $h^{\text{sel}}(\mathbf{x})$ holds a set of M weak classifiers $\{h_1^{\text{weak}}(\mathbf{x}),\ldots, h_M^{\text{weak}}(\mathbf{x})\}$ and selects one of them

$$h^{\text{sel}}(\mathbf{x}) = h_m^{\text{weak}}(\mathbf{x}) \tag{2}$$

according to an optimization criterion (we use the estimated error e_i of each weak classifier h_i^{weak} such that $m=\arg\min_i e_i$). Note, that the selector can be interpreted as a classifier as it switches between the weak classifiers. Training a selector means that each weak classifier is updated and the best one with the lowest estimated error is selected. Similar to the off-line case, the weak classifiers correspond to features, i.e. the hypotheses generated by the weak classifier are based on the response of the features.

In particular, the on-line training version of AdaBoost for feature selection works as follows: First, a fixed set of N selectors, $h_1^{\text{sel}},\ldots, h_N^{\text{sel}}$, is initialized randomly with weak classifiers, i.e. features. When a new training sample $\langle\mathbf{x},y\rangle$ arrives, the selectors are updated. This update is done with respect to the importance weight λ of the current sample. For updating the weak classifiers, any on-line learning algorithm can be used (see Section 3.4 for more details). The weak classifier with the smallest estimated error is chosen by the selector. The corresponding voting weight α_n and the importance weight λ of the sample are updated and passed to the next selector h_{n+1}^{sel}. The weight increases if the example is misclassified by the current selector and decreases otherwise. For more details, see Grabner and Bischof (2006).

Finally, a strong classifier is obtained by linear combination of N selectors.

$$h^{\text{strong}}(\mathbf{x}) = \text{sign}\left(\sum_{n=1}^{N} \alpha_n \cdot h_n^{\text{sel}}(\mathbf{x})\right) \tag{3}$$

In contrast to the off-line version a classifier is available at any time and can be directly evaluated which allows to provide immediate user feedback at any stage of the training process.

3.4. Image representation and features

The main purpose of using features instead of raw pixel values as input to a learning algorithm is to reduce

the intra-class variability while increasing the extra-class variability and adding insensitivity to certain image variations (e.g illumination). We use three different types of features:

- Haar-like features (Viola and Jones, 2001): The feature value is calculated as the sum of pixel values within rectangular regions which are either positive or negative weighted. These features were introduced by Viola and Jones for face detection and are now widely used in computer vision. We use four different prototypes of features, see Fig. 1(a). A two-rectangle feature consists of two regions which have the same size and shape and are horizontally or vertically adjacent. For a three-rectangle feature, the sum for the two out- side rectangles is subtracted from the sum in the center rectangle. For the four-rectangle feature the difference between diagonal pairs of rectangles is computed. Finally, for a center-feature the center region is subtracted from the surrounding pixels. These features are calculated at different scales.
- Orientation histograms (Levi and Weiss, 2004; Dalal and Triggs, 2005): First, a gradient image is computed using the Sobel-filter. A magnitude weighted histogram over the gradient directions is built to represent the underlying rectangular patch. In particular, we use an 8 bin orientation histogram with constant bin size. The basic idea is to describe the appearance of an object part by the gradient information similar to the famous SIFT descriptor by Lowe (2004).
- A simplified version of local binary patterns (LBP) (Ojala et al., 2002): We use a four-neighborhood, i.e. $2^4=16$ patterns, as a 16 bin histogram feature similar to (Zhang et al., 2006). This is a texture descriptor which captures the statistic of normalized pixel values in a local neighborhood. The LBP-value of a 3×3 image patch $\mathbf{x}$ is calculated as follows (see also Fig. 1(b)):

$$\mathrm{LBP}(\mathbf{x}) = \sum_{i=0}^{3} s(x_i - x_{\mathrm{center}}) \cdot 2^i \tag{4}$$

$$\text{with} \quad s(z) = \begin{cases} 1 & z \geq 0 \\ 0 & z < 0 \end{cases}$$

The final representation is a histogram of the LBP values obtained by shifting the 3×3 patch in the whole image patch.

Note, that the computation of all these feature types can be done very efficiently using integral images (Viola and Jones, 2001) and integral histograms (Porikli, 2005) as data structures. This allows for exhaustive template matching for the whole image. An integral image, denoted as II, sums up all the pixel values from the upper left up to the current position. More formally, it is defined on an image I as

$$II(x,y) = \sum_{x'=1}^{x} \sum_{y'=1}^{y} I(x',y') \tag{5}$$

The pre-calculation of an integral image for all pixels can be efficiently implemented in one pass over the image. Afterwards, any sum of any rectangular region can be computed by only 4 memory accesses and 3 additions, see Fig. 2 for an example. This idea can be easily adapted to represent histograms: for each bin one integral image is built separately.

To obtain a weak classifier h_j^{weak} from a feature j, we model the probability distribution of this feature for positive and negative samples with $f_j(\mathbf{x})$ evaluating this feature on the image $\mathbf{x}$. Following Grabner and Bischof (2006) we estimate the probability $P(1|f_j(\mathbf{x}))$ assuming a Gaussian distribution $\mathcal{N}(\mu^+,\sigma^+)$, i.e. we incrementally

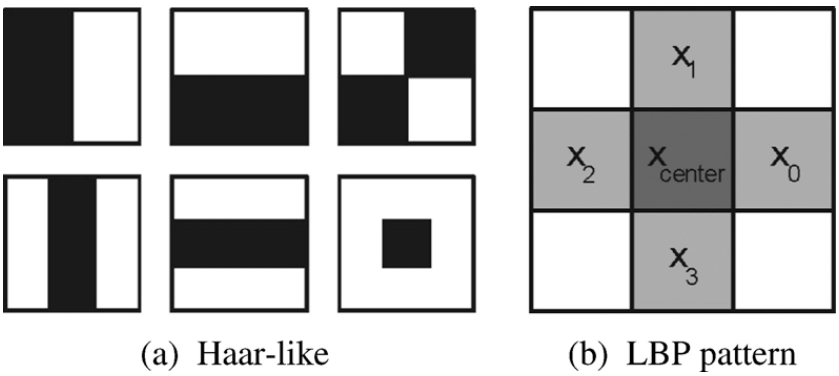

Fig. 1. Basic image features used. (a) The value of the Haar-like feature is the difference of the pixel values between the white and the black marked region. (b) Simple version to obtain a local binary pattern value (LBP).

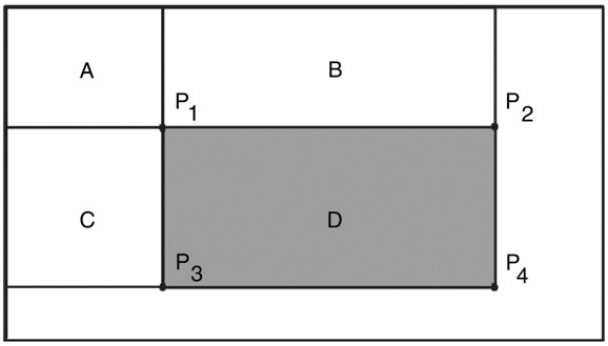

Fig. 2. Efficient calculation of the sum over a rectangular area. The value of the integral image at Position P_1 is the sum of the pixel values in region A. P_2 corresponds $A+B$, P_3 to $A+C$ and P_4 to $A+B+C+D$. Therefore, the sum over the area D can be calculated by $P_4+P_1-P_2-P_3$.

update μ^+ and σ^+ for positively labeled samples and $P(-1|f_j(\mathbf{x}))$ by $\mathcal{N}(\mu^-,\sigma^-)$ for negatively labeled samples.

For the classic Haar-like features, we use a Bayesian decision criterion based on the estimated Gaussian probability density function $g(x|\mu, \sigma)$.

$$h_j^{\text{weak}}(\mathbf{x}) = \text{sign}\left(P(1|f_j(\mathbf{x})) - P(-1|f_j(\mathbf{x}))\right) \approx \text{sign}\left(g(f_j(\mathbf{x})|\mu^+,\sigma^+) - g(f_j(\mathbf{x})|\mu^-,\sigma^-)\right) \quad (6)$$

For the histogram based feature types, orientation histograms and LBP, we employ a nearest neighbor learning algorithm. The positive and negative samples are modeled by one cluster each. The cluster centers $\mathbf{p_j}$ and $\mathbf{n_j}$ are incrementally updated. The weak classifier is given by

$$h_j^{\text{weak}}(\mathbf{x}) = \text{sign}\left(D\left(f_j(\mathbf{x}),\mathbf{p_j}\right) - D\left(f_j(\mathbf{x}),\mathbf{n_j}\right)\right) \quad (7)$$

where D is a distance metric, in our case the Euclidian norm is used.

Since we know the resolution of the image, search for cars at different scales is not necessary. Yet cars can appear at any orientation. Instead of training the classifier with different orientations we train it at one "norm" orientation. The detector can be made rotation invariant by computing the features at different angles. Lienhart in Lienhart and Maydt (2002) introduced an additional set of rotated Haar-like features, which comprise an enriched set of basic features and can be computed efficiently. Barczack et al. (2005) proposed to use different types of Haar-like features. A previously trained classifier is converted to work at any angle, so rotated objects can be detected. A real-time version for the rotation invariant Viola–Jones detector has been reported in Wu et al. (2004). A similar technique is employed in our system: the detector is rotated by increments in 10°. For the orientation histogram features, the rotation can be done easily by shifting the histogram.

3.5. Training and detection

The training process is performed by iteratively labeling samples from the images and updating parameters for the model. The labeled samples can be positive or negative. In order to minimize the hand labeling effort, we apply an active learning strategy. The key idea is that the user has to label only examples which are not correctly classified by the current classifier. In fact, it has been shown by the active learning community (Park and Choi, 1996), that it is more effective to sample at the current estimate of the decision boundary than at the unknown true boundary. This is exactly what we aim at with our approach.

We evaluate the current classifier on an image. The human supervisor labels additionally "informative" samples, e.g. marks a wrongly labeled example which can be either a false or missed detection. The classifier is evaluated and updated after each labeling of a sample. The new updated classifier is applied again on the same image or on a new image, and the process continues. This is a fully supervised interactive learning process.

Since labeling of samples in the training phase is an interactive process with human supervision, we can intuitively choose to label the most informative and discriminative sample at each update. This allows the parameters of the model to be updated in a greedy manner with respect to minimizing the detection error, meaning that the parameters of the model can be learned very fast. It also avoids labeling redundant samples that do not contribute to the current decision boundary. Therefore this saves a lot of labeling effort. Moreover, by storing parameters of the current training classifier, we can retrain it and make use of pre-trained classifier any time, if necessary.

After training, the detection is performed by applying the trained classifier exhaustively on the images. A car is considered to be detected if the output confidence value of the classifier is above a threshold, i.e. zero. 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 threshold the false positives decrease at the expense of the detections. The process results in many overlapping detections. Therefore, a post processing stage is needed to refine and combine these outputs. It significantly improves the detection rate.

3.6. Post processing

Following Grabner et al. (2005) we use non-parametric clustering-based object detection derived

from the probability distribution of classifier output. The strong classifier generates a probabilistic output. For each image location U we obtain multiple outputs P_k representing object appearance's probability (in our case the confidence $conf(\cdot)$ of the strong classifier) at each angle k of the image. To obtain a distribution of object probabilities for each rotation angle, we apply kernel density estimation. Let $\{U_i\}_{i=1,\ldots,n}$ denote the image locations where classification is performed. For each angle k we obtain a probability density estimate

$$\hat{p}_k(\mathbf{u}) = \sum_{i=1}^{n} P_k(U_i) \cdot K_{\mathrm{k}}\left(\frac{\mathbf{u} - U_i}{W}\right), \tag{8}$$

where K_{k} is a two-dimensional Gaussian kernel with a size equivalent to the object size W scaled by the confidence of the classifier output. The cumulative density estimate containing the sum of probabilities oven all angles is denoted as $\hat{p}_c(\mathbf{u}) = \Sigma_k \hat{p}_k(\mathbf{u})$. Mean shift clustering is applied to this density estimate to delineate the appearance of objects. In our case, a simple version is used where K is a two-dimensional flat kernel.

The mean shift algorithm is a non-parametric technique to locate density extrema on modes of a given distribution by an iterative procedure (Comaniciu and Meer, 1999). Starting from a location $\mathbf{u}$ the local mean shift vector represents an offset to $\mathbf{u}'$, the nearest mode along the direction of maximum increase in the underlying density function. The density is estimated within the local neighborhood by kernel density estimation where at a data point a kernel weights $K(\mathbf{a})$ are combined with weights associated with the data, i.e. with sample weights. In our case sample weights are defined by the values of the density estimate $\hat{p}_c(\mathbf{a})$ at pixel locations $\mathbf{a}$. The new location vector $\mathbf{u}'$ is obtained by

$$\mathbf{u}' = \frac{\sum_{\mathbf{a}} K(\mathbf{a} - \mathbf{u}) \cdot \hat{p}_c(\mathbf{a}) \cdot \mathbf{a}}{\sum_{a} K(\mathbf{a} - \mathbf{u}) \cdot \hat{p}_c(\mathbf{a})}. \tag{9}$$

For a uniform kernel K that we use here, it was shown that fast evaluation of Eq. (9) is feasible using integral images (Beleznai et al., 2004).

3.7. Land use classification and street layer

In some applications, context information is available and can be used for further improvement of the performance. In aerial images there may exist details of roofs, windows, etc. of buildings that look similar to cars and may therefore lead to false detections. These false detections can be eliminated by using results from other processing stages of the interpretation of aerial images (Zebedin et al., 2006; Leberl et al., 2003). In this work, a street layer obtaining by land use classification (Zebedin et al., 2006) is employed as context. The street layer contains road information, which is used for improving the detection rate.

Land use classification is a two-step process performed on multi spectral digital aerial images. For initial classification, RGB and NIR images are used. A support vector machine (Vapnik, 1995) is trained for classification on these images. For refined classification, additional height data generated by aerial triangulation and dense

Fig. 3. Examples of positively (a) and negatively (b) labeled training samples during the on-line training process.

matching are used. The classification results are data layers for streets, buildings, trees, low vegetation and water. In the context of car detection we are only interested in the street layer. The street layer is used to mask the possible regions such as road or parking lots, where cars may be located. This helps to reduce the number of false positives considerably.

4. Experiment and result

The aim of our experiments is to demonstrate the efficiency of on-line training and the robustness of our framework for car detection from aerial images.

4.1. Data sets

In this work we use two different datasets. The first dataset was acquired in the summer of 2005 from the city center of Graz, Austria. It consists of 155 images flown in 5 strips. The along-track overlap is 80% and the across-track overlap approximately 60%. The ground sampling distance is about 8 cm. Therefore, a car is supposed to consist of 24×50 image pixels. The second dataset was acquired in the winter of 2005 capturing the city center of Philadelphia. It consists of 158 images with an overlap of 90% and a sidelap of approximately 60%. The ground sampling distance is about 10 cm.

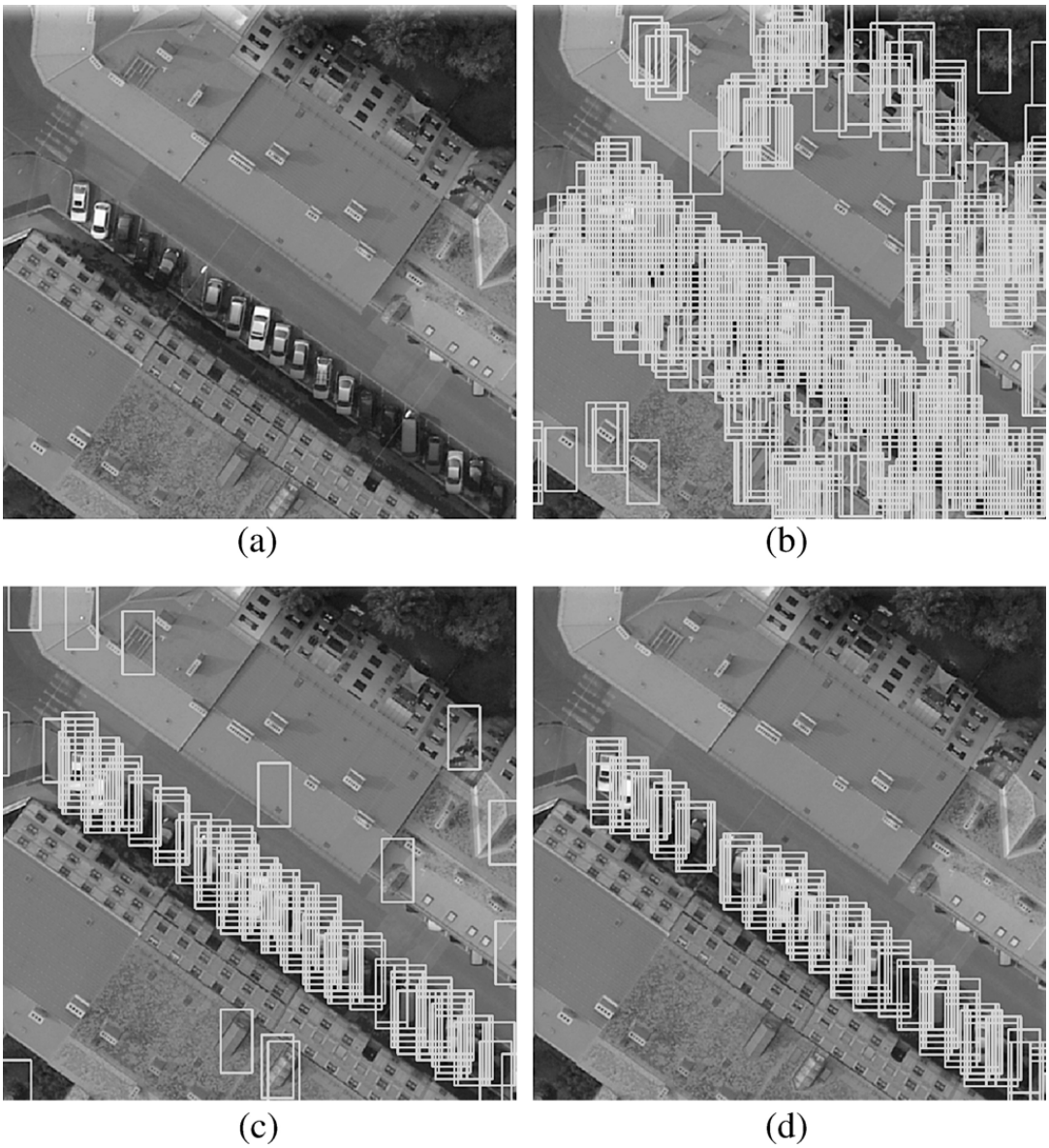

Fig. 4. Learning process: improvement of classifier performance — (a) original subimage from Graz data set, (b) result after training with only one positive sample, (c) after training with 10 samples and (d) final result without post processing after training with 50 samples.

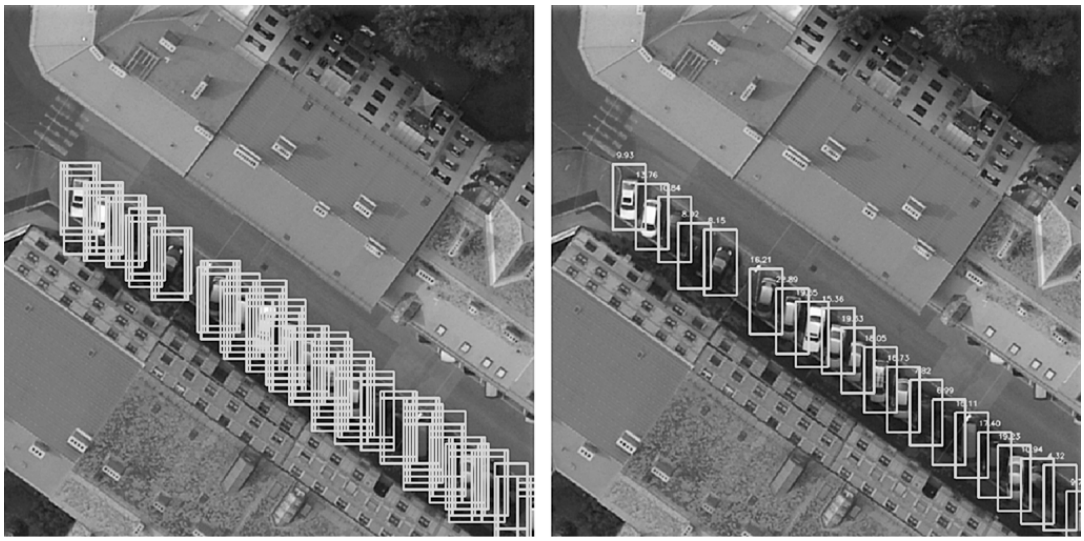

Fig. 5. Postprocessing: (a) raw output of the classifier applied on a subimage, (b) after combing multiple detections by mean shift based clustering (subimage from Graz data).

Both datasets were acquired by the UltraCamD camera. The high resolution panchromatic images used for car detection, aerial triangulation and dense matching have a size of 11,500 pixel across-track and 7500 pixel along-track. The multi spectral low resolution images employed in the initial step of land use classification have a size of 3680×2400 pixel. The high overlap was chosen to make aerial triangulation and dense matching more reliable.

Because we know the resolution of the aerial images, we can specify rectangle patch size of cars. It has to be carefully chosen to cover the area, which contains a car in the middle and a narrow margin (see Fig. 3(a)). This is done in order to include contextual information of cars. Usually the car's length is double its width. We have chosen the patch size to be 35×70 pixel.

For this paper, only four large typical subimages were employed as training and test sets. Two images are from Graz city, the other two from Philadelphia. Each subimage has a size of 4000× 4000. The test sets are separated from the training sets. The test sets Graz and Philadelphia contain 324 and 1495 cars, respectively. We use gray-valued images obtained from the original multi spectral images for training and testing.

4.2. Training

We start with a random classifier which comprises of 500 weak classifiers and 250 selectors. The classifier is improved on-line after labeling training samples by the user. Thus, we make use of the advantages of active learning. During training we have labeled 1420 samples. There are 410 positive samples, each sample containing a car, and 1010 negative samples, each showing diverse background patches (for examples see Fig. 3)[1]. The more informative the samples are, the faster the system learns. Moreover, the training samples can be diversified and adjusted during training to capture the variability of the real data. That the number of positive samples is much smaller than the number of negative samples stems from the fact that the variability of the background is much larger than of the cars. In comparison with other object (car) detection systems, our system needs quite a small number of training samples. As can be seen in Fig. 4, after several training iterations almost all cars which have a distinct appearance and fit to the (angle of the) detector are detected. Fig. 5 depicts the detected objects without the refinement step (a) and the detected points after applying mean shift clustering (b). Finally, Fig. 6 shows the continuous improvement of the classifier over time, i.e., for an increasing number of labeled training samples, on Graz data set.

[1] Since we train on-line a classifier is available all the time. An acceptable result can be obtained after labeling about 800 samples. The longer the training, i.e., the more samples are labeled, the better is the performance.

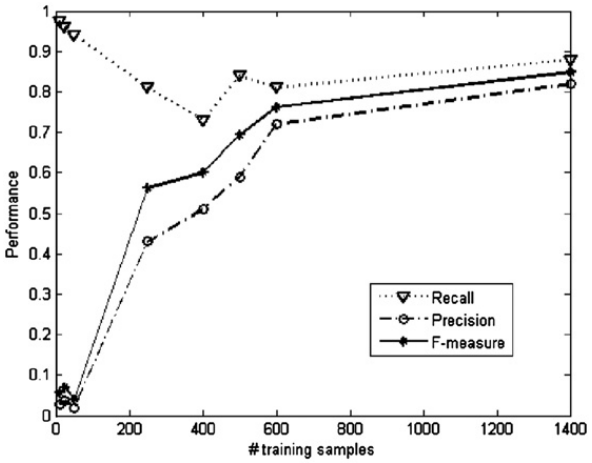

Fig. 6. Learning process: performance versus number of training examples, on Graz data.

Since we want to have good training samples for fast training the classifier, we first train the detector on Graz data. This data set is less noisy due to good imaging quality (in term of sharpness and contrast). We then evaluated the system on both test sets. The performance is quite good on Graz data set. However, it drops significantly on Philadelphia data. So further training is needed to cope with variability of real data. As we expected, after training with few samples from the Philadelphia data, the classifier adapts quite well and reaches the performance as reported in the following section. Note that we keep the same parameter settings as well as the same car patch size for both data sets in training and test phases, except at post processing step. The only parameter that needs to be adjusted is a threshold of the confidence (cf. Section 3.5). For Graz

Fig. 7. Results of car detection in large aerial images (left: Graz images, right: Philadelphia images): Cars appear with different orientations and are partly occluded all on highly complicated background. The dark squares represent detections at different angles and bright points are detections after post processing, each point corresponds to one detected car.

data, distinctive car features are easily obtained due to good imaging quality, this value is set higher to avoid some false positives that may occur. For Philadelphia data, the value is smaller.

4.3. Performance evaluation

We apply the trained detector on the whole image to detect cars. For quantitative evaluation, we count as a correct detection if the distance between a detected patch's center and a car's center in ground truth is less than half width of car size (i.e. 10 pixels).

We report the results for the two data sets Graz and Philadelphia, which contain 324 and 1495 cars, respectively. Figs. 7 and 9 present the detection results in several subimages. They show complicated backgrounds of urban scenes with many car-like objects. The cars also appear with slightly different view angles, different contrast, lighting condition, etc. Many cars are severely occluded by buildings or trees, dominated by their shadow, or have very low contrast. As one can see, all the cars with distinctive features have been detected. Also almost all difficult cars were found. Some partly occluded cars are detected, some are missed. Some objects that look like cars are reported as cars, but with low confidence value and have been removed at the post processing stage. The system can also deal with slightly different sizes of cars. We have trained it on samples of size of 35×70 pixels. We then applied it for detection of cars on both Graz and Philadelphia datasets with ground sampling distances of approximately 8 cm and 10 cm, respectively. The results show no significant differences. For performance evaluation, we use a common measure for object detection namely recall-precision curves (RPC) as in (Agarwal et al., 2004):

$$\text{Precision rate} = \frac{\#\,\text{true positives}}{\#\,\text{true positives} + \#\,\text{false positives}} \quad (10)$$

$$\text{Recall rate} = \frac{\#\,\text{true positives}}{\#\,\text{true positives} + \#\,\text{false negatives}} \quad (11)$$

$$F - \text{measure} = \frac{2 \cdot \text{Recall rate} \cdot \text{Precision rate}}{\text{Recall rate} + \text{Precision rate}} \quad (12)$$

The precision rate shows how accurate we are at predicting the positive class. The recall rate tells us how many of the total positives we are able to identify. For detection there is always a compromise between precision and recall. This is evaluated by the F-measure as the harmonic mean. The RPCs characterizing the performance of our framework for the two datasets with the same parameters setting are given in Fig. 8 (lower curves).

In applications such as the estimation of traffic flow, context information can be given. We use the street layer from land use classification for road verification (see Section 3.7). This information improves the detection by eliminating false positives (cf. Fig. 9).

For a comparison, besides the regular RPC curves, the RPC curves taking into account context information are also given in Fig. 8, upper ones. As expected, the performance of the system is improved.

Experimental results show that in general the performance of our framework is good and even superior in

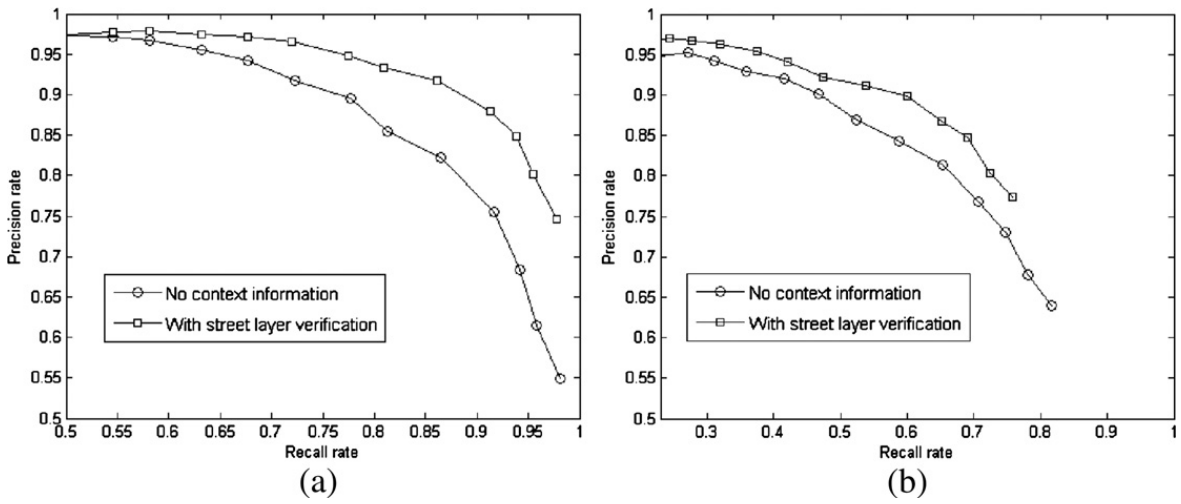

Fig. 8. RPC of the system on Graz data set (a) and on Philadelphia data set (b); upper curves: increasing detection performance on the Graz and Philadelphia datasets when including context information (street layer classification).

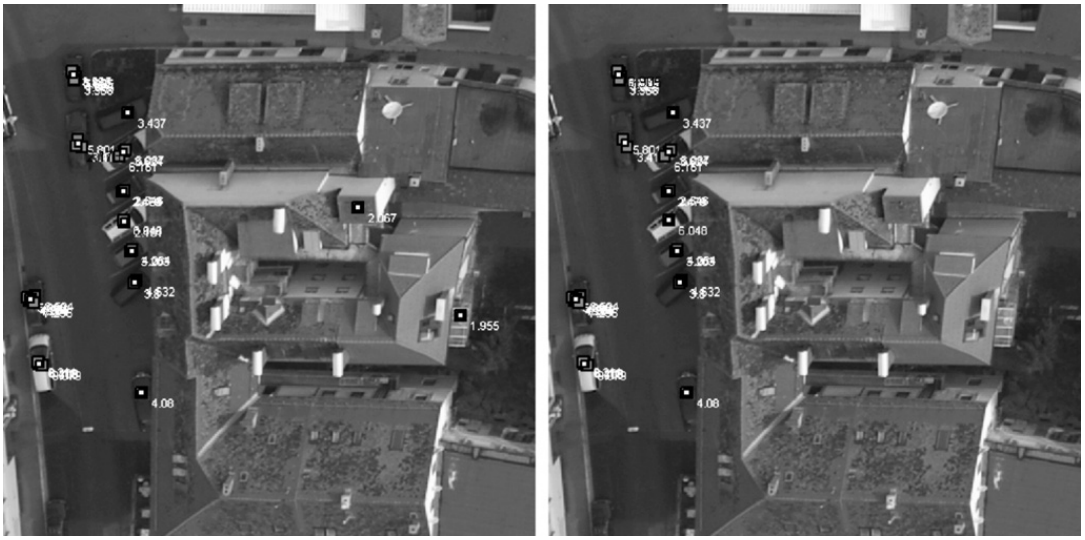

Fig. 9. Objects on the roof which have been reported as cars are removed using the road mask. The dark squares represent detections at different angles and bright points are detections after postprocessing, each point corresponds to one detected car.

comparison related work (Zhao and Nevatia, 2003; Hinz, 2003; Hinz and Stilla, 2006; Ruskone et al., 1996; Yao and Zhang, 2005).

Moreover, it is a robust and automatic system on large-scale aerial images. In terms of detection rate and especially efficiency: Due to the lack of public available datasets for evaluation of the system, a fair comparison is not possible. Additionally, different methods have been employed for evaluation. Some related works even did not provide clearly their performance evaluation, only some intuitive results were shown (Schlosser et al., 2003; Hinz, 2003).

4.4. Exploiting the redundancy in multiple images

The high overlap of the UltraCamD images results in a high redundancy which can be exploited to improve the car detection at no additional costs. Leberl and Szabo (2005): "One can produce as many images within a flight line as one wishes, with no added costs, and thus increase the traditional forward overlap from 60% to 80% or 90%". The high overlaps produce multiple images for each ground point. This can be exploited to reduce occlusions due to buildings and vegetation, providing superior results. As one can see in Fig. 10, cars which are occluded by buildings or trees in one image can become visible and detected in other image of the same flight. For applications such as estimation of transportation flow or terrestrial texture restoration, the use of redundancy is certainly very helpful. Since the establishment of ground truth is tedious for overlapping images, we have not yet systematically evaluated the improvement by using redundancy.

5. Conclusion and future work

We have developed an efficient framework for automatic car detection from aerial images. This is the first proposal to use a state-of-the-art machine learning technique, namely Adaboost, for the detection of cars from large-scale aerial images. We have used integral images for efficient representation and computation of car features. Three types of features, Haar-like, orientation histogram and local binary pattern, are employed for generating hypothesis for training the detector. Moreover, a novel on-line version of boosting is used for efficient training of the developed system. On-line learning avoids building huge pre-labeled training set and makes use of interactive training. This results in a robust and efficient system for car detection from large aerial images. The system also deals well with the variability of car appearances in complicated backgrounds of urban aerial images. Experimental results show the applicability and even the superiority of our framework for applications including estimation of transportation flow, road verification for supporting land use classification and for restoring texture to complete 3D map generation from digital aerial images.

The system can be improved and extended in the following ways:

- Inclusion of more data samples for training. This results in improvement of the generalization of the detector and better performance.
- Diversification of the features or parameters for weak classifiers. This increases the complexity of the system making it possible to deal with hard samples.

Fig. 10. The utilization of multiple overlapping images with different viewing angles: objects (cars) that are occluded in one image (left images) can be visible and therefore can be detected in another image (right images).

- Use of information from aerial triangulation and possibility also dense matching to detect cars in multiple, overlapping images that differ in their viewing angle including automatic combination of the results. This yields higher performance for the system.

Acknowledgments

This work has been sponsored in part by the Austrian Joint Research Project Cognitive Vision under projects S9103-N04 and S9104-N04, by the EU FP6-507752 NoE MUSCLE IST project, and by the Austrian Exchange Service (OeAD) under project EZA-894. Parts of this work have been done in the VRVis research center, Graz, Austria (http://www.vrvis.at), which is partly funded by the Austrian government research program Kplus.

References

Abramsol, Y., Freuld, Y., 2005. SEmi-automatic VIsual LEarning (SEVILLE): Tutorial on active learning for visual object recognition Tech. rep., UCSD.

Agarwal, S., Awal, A., Roth, D., 2004. Learning to detect objects in images via a sparse, part-based representation. Transactions on Pattern Analysis and Machine Intelligence 26 (11), 1475–1490.

Alba-Flores, R., 2005. Evaluation of the Use of High-resolution Satellite Imagery in Transportation Applications. Tech. rep., Intelligent Transportation System Institute. University of Minnesota.

Barczack, A.L.C., Johlsol, M.J., Messom, C.H., 2005. Real-time computation of Haar-like features at generic angles for detection algorithms. Research Letters in the Information and Mathematical Sciences 9, 98–111.

Beleznai, C., Fruhstuck, B., Bischof, H., Kropatsch, W., 2004. Detecting humans in groups using a fast mean shift procedure. Proceedings Workshop of the Austrian Association for Pattern Recognition. Austrian Computer Society (OCG), Hagenberg, Austria, pp. 71–78.

Bernstein, E., Amit, Y., 2005. Part-based statistical models for object classification and detection. Proceedings of Computer Vision and Pattern Recognition, Vol. 2. IEEE Computer Society, San Diego, CA, USA, pp. 734–740.

Bileschi, S.M., Leung, B., Rifkin, R.M., 2004. Towards component-based car detection. ECCV Workshop on Statistical Learning and Computer Vision. Springer, Prague, Czech Republic, pp. 75–98.

Comaniciu, D., Meer, P., 1999. Mean shift analysis and applications. Proceedings, IEEE International Conference on Computer Vision. IEEE Computer Society, pp. 1197–1203. Kerkyra, Greece.

Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection. Proceedings Conference on Computer Vision and Pattern Recognition. Vol. 1. IEEE Computer Society, San Diego, CA, pp. 886–893.

Demiriz, A., Bennett, K., Shawe-Taylor, J., 2002. Linear programming boosting via column generation. Machine Learning 46 (1-3), 225–254.

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

Grabner, H., Beleznai, C., Bischof, H., 2005. Improving adaboost detection rate by wobble and mean shift. Proceedings Computer Vision Winter Workshop. Austrian Computer Society (OCG), Zell an der Pram, Austria, pp. 23–32.

Grabner, H., Bischof, H., 2006. On-line boosting and vision. Proceedings Conference on Computer Vision and Pattern Recognition. Vol. 1. IEEE Computer Society, New York, NY, pp. 260–267.

Heisele, B., Riskov, I., Morgenstern, C., 2006. Components for Object Detection and Identification. Vol. 4170, Springer Berlin, Heidelberg Germany, Ch. III, pp 225–237.

Hinz, S., 2003. Detection and counting of cars in aerial images. International Conference on Image Processing. Vol. 3. IEEE, Barcelona, pp. 997–1000.

Hinz, S., Stilla, U., 2006. Car detection in aerial thermal images by local and global evidence accumulation. Pattern Recognition Letters 27 (4), 308–315.

Javed, O., Ali, S., Shah, M., 2005. Online detection and classification of moving objects using progressively improving detectors. Proceedings Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, San Diego, CA, pp. 695–700.

Leberl, F., Gruber, M., Ponticelli, M., Berloegger, S., Perko, R., 2003. The UltraCam large format aerial digital camera system. Proceedings of the ASPRS Annual Conveltion, Anchorage USA, pp. 5–9. May. On CDROM.

Leberl, F., Szabo, J., August 2005. Novel totally digital photogrammetric workflow. Tech. rep., Semana Geomatica, IGAC-Bogota, Colombia.

Leibe, B., Leolardis, A., Schiele, B., 2004. Combined object categorization and segmentation with an implicit shape model. ECCV'04 Workshop on Statistical Learning in Computer Vision. Springer-Verlag, Prague, pp. 17–32.

Levi, K., Weiss, Y., 2004. Learning object detection from a small number of examples: The importance of good features. Proceedings of Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, Washington, DC, pp. 53–60.

Levin, A., Viola, P., Freund, Y., 2003. Unsupervised improvement of visual detectors using co-training. Proceedings of International Conference on Computer Vision. Vol. 2. IEEE Computer Society, Nice, pp. 626–633.

Lienhart, R., Maydt, J., 2002. An extended set of Haar-like features for object detection. Proceedings of the IEEE International Conference on Image Processing. IEEE, New York, pp. 900–903.

Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60 (2), 91–110.

Moon, H., Chellappa, R., Roselfeld, A., 2002. Performance analysis of a simple vehicle detection algorithm. Image and Vision Computing 20 (1), 1–13.

Nair, V., Clark, J., 2004. An unsupervised, online learning framework for moving object detection. Proceedings Conference on Computer Vision and Pattern Recognition. Vol. 2. IEEE, Washington, DC, pp. 317–324.

Ojala, T., Pietikäinen, M., Mäenpää, T., 2002. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. Pattern Analysis and Machine Intelligence 24 (7), 971–987.

Oza, N., Russell, S., 2001a. Experimental comparisons of online and batch versions of bagging and boosting. Proceedings ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM Press, San Francisco, CA, pp. 359–364.

Oza, N., Russell, S., 2001b. Online bagging and boosting. Proceedings Artificial Intelligence and Statistics. Morgan Kaufmann, Florida, pp. 105–112.

Papageorgiou, C., Poggio, T., 2000. A trainable system for object detection. International Journal of Computer Vision 38 (1), 15–33. June.

Park, J.-H., Choi, Y.-K., 1996. On-line learning for active pattern recognition. IEEE Signal Processilg Letters 3 (11), 301–303.

Porikli, F., 2005. Integral histogram: a fast way to extract histograms in Cartesian spaces. Proceedings Conference on Computer Vision and Pattern Recognition. Vol. 1. IEEE Computer Society, San Diego, CA, pp. 829–836.

Rajagopalan, A.N., Burlina, P., Chellappa, R., 1999. Higher order statistical learning for vehicle detection in images. International Conference on Computer Vision. Vol. 2. IEEE Computer Society, Corfu, pp. 1204–1209.

Roth, P., Grabner, H., Skočaj, D., Bischof, H., Leonardis, A., 2005. On-line conservative learning for person detection. Proceedings Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance. IEEE, Beijing, China, pp. 223–230.

Rudin, C., Daubechies, I., Schapire, R., 2004. The dynamics of adaboost: cyclic behavior and convergence of margins. Journal of Machine Learning Research 5, 1557–1595.

Ruskone, R., Guigues, L., Airault, S., Jamet, O., 1996. Vehicle detection on aerial images: a structural approach. International Conference on Pattern Recognition. Vol. 3. IEEE Computer Society, Vienna, pp. 900–904.

Schapire, R., 2003. The Boosting Approach to Machine Learning: An Overview. Nonlinear Estimation and Classification. Springer.

Schapire, R., Freund, Y., Bartlett, P., Lee, W., 1997. Boosting the margin: a new explanation for the effectiveness of voting methods. Proceedings International Conference on Machine Learning. Morgan Kaufmann, Nashville, TN, pp. 322–330.

Schlosser, C., Reitberger, J., Hinz, S., 2003. Automatic car detection in high resolution urban scenes based on an adaptive 3D- model. GRSS/ISPRS Joint Workshop on Data Fusion and Remote Sensing over Urban Areas, 2nd. IEEE, pp. 167–171 (No. 0-7803-7719-2).

Schneiderman, H., Kanade, T., 2000. A statistical method for 3d object detection applied to faces and cars. Proceedings Conference on Computer Vision and Pattern Recognition. Vol. 1. IEEE Computer Society, Hilton Head, SC, pp. 746–751.

Stojmenovic, M., 2006. Real time machine learning based car detection in images with fast training. Machine Vision and Applications 17 (3), 163–172.

Tieu, K., Viola, P., 2000. Boosting image retrieval. Proceedings Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, Hilton Head, SC, pp. 228–235.

Vapnik, V., 1995. The Nature of Statistical Learning Theory. Springer, New York.

Viola, P., Jones, M., 2001. Rapid object detection using a boosted cascade of simple features. Proceedings Conference on Computer Vision and Pattern Recognition. Vol. 1. IEEE Computer Society, Kauai, HI, pp. 511–518.

Wu, B., Ai, H., Huang, C., Lao, S., 2004. Fast rotation invariant multi-view face detection based on real adaboost. Proceedings International Conference on Automatic Face and Gesture Recognition. IEEE Computer Society, Seoul, Korea, pp. 79–84.

Yao, J., Zhang, Z., 2005. Semi-supervised learning based object detection in aerial imagery. Proceedings of the Computer Vision and Pattern Recognition. Vol. 1. IEEE Computer Society, San Diego, CA, pp. 1011–1016.

Zebedin, L., Klaus, A., Gruber-Geymayer, B., Karnera, K., 2006. Towards 3d map generation from digital aerial images. ISPRS Journal of Photogrammetry and Remote Sensing 60 (6), 413–427.

Zhang, H., Gao, W., Chen, Y., Zhao, D., 2006. Object detection using spatial histogram features. Image and Vision Computing 24 (4), 327–341.

Zhao, T., Nevatia, R., 2003. Car detection in low resolution images. Image and Vision Computing 21 (8), 693–703.