# Fast multiclass vehicle detection on aerial images

Kang Liu and Gellert Mattyus

***Abstract*—Detecting vehicles in aerial images provides important information for traffic management and urban planning. Detecting the cars in the images is challenging due to the relatively small size of the target objects and the complex background in man-made areas. It is particularly challenging if the goal is near real-time detection - within few seconds - on large images without any additional information, e.g. road database, accurate target size. We present a method which can detect the vehicles on a 21 MPixel original frame image without an accurate scale information within seconds on a laptop single threaded. Beside the bounding box of the vehicles we extract also an orientation and type (car/truck) information. First we apply a fast binary detector using Integral Channel Features in a Soft Cascade structure. In the next step we apply a multiclass classifier on the output of the binary detector which gives the orientation and type of the vehicles. We evaluate our method on a challenging dataset of original aerial images over Munich and a dataset captured from a UAV.**

***Index Terms*—vehicle detection, classification, near real-time**

## I. INTRODUCTION

The detection of vehicles in aerial images is important for various applications e.g. traffic management, parking lot utilization, urban planning, etc. Collecting traffic and parking data from an airborne platform gives fast coverage over a larger area. Getting the same coverage by terrestrial sensors would need the deployment of more sensors, more manual work, thus higher costs.

A good example for an airborne road traffic measuring system is the one in the project *Vabene* [1] of the German Aerospace Center (DLR). In this real-time system aerial images are captured over roads and the vehicles are detected and tracked across multiple consecutive frames. This gives a fast and comprehensive information of the traffic situation by providing the number of vehicles and their position and speed. Fig. 1 provides the overview of our work flow and illustration of the output. The detection is a challenging problem due to the small size of the vehicles (a car might be only 30 × 12 pixels) and the complex background of man-made objects which appear visually similar to the cars. Providing both the position and the orientation of the detected objects supports the tracking by giving constraints on the motion of the vehicles. This is particularly important in dense traffic scenes where the object assignment is more challenging. The utilization of roads and parking lots depends also on the type of the vehicle (e.g. a truck impacts the traffic flow different as a personal car). A system having access to this richer information can manage the infrastructure better. In a real-time system as in [1] the processing time (and computing power) is limited. Therefore the processing method should be as fast as possible.

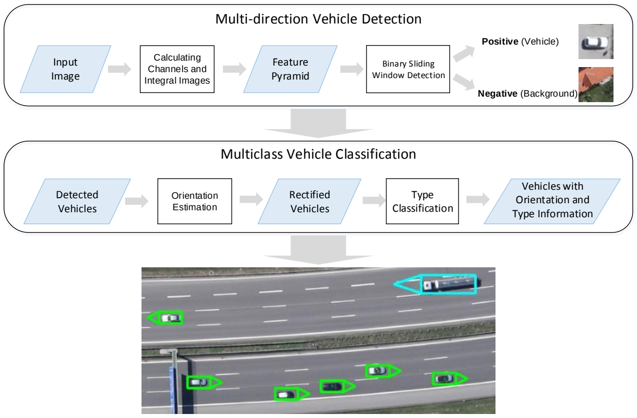

Multi-direction Vehicle Detection
Input Image
Calculating Channels and Integral Images
Feature Pyramid
Binary Sliding Window Detection
Positive (Vehicle)
Negative (Background)
Multiclass Vehicle Classification
Detected Vehicles
Orientation Estimation
Rectified Vehicles
Type Classification
Vehicles with Orientation and Type Information

Fig. 1. Proposed vehicle detection framework. The input image is first evaluated by the multi-direction vehicle detector. A sliding window goes along $x$- and $y$-axes. Features are extracted from the detection window and sent to trained binary classifier. The binary classifier classify whether current detection window contains a positive object or not. Detected vehicles are then processed for estimating their orientations and categories.

Our vehicle detection method provides both robust performance, fast speed and vehicle orientation and type information fully automatically based only on the input image.

We detect the bounding box of the vehicles by a very fast binary sliding window detector using Integral Channel Features and an AdaBoost classifier in Soft Cascade structure. The bounding boxes are further classified to different orientations and vehicle type based on HOG features [2].

We test and evaluate our method on a challenging dataset over the city Munich, Germany and another dataset collected by a UAV. These datasets contain original, non-orthorectified frame images which makes the problem more challenging since the exact GSD[1] is unknown (we have only an approximate prior). To make our results better comparable to other methods, we release the Munich images with the ground truth[2]. To show the robustness of the method we also present qualitative results on images downloaded from Google Earth around the world in the supplementary material.

Our main contributions are: (i) The presented method uses features which can be calculated rapidly in a Soft Cascade structure. This makes the detection very fast, it takes only a few seconds on a 21 MPixel image on a laptop single threaded. (ii) Our method also works on a single original frame image without any georeferencing, exact GSD, street database or 3D image information. (iii) Beside the location we also estimate the orientation and type of the vehicles.

K. Liu and G. Mattyus are with the Remote Sensing Technology Institute of the German Aerospace Center.

[1] Ground Sampling Distance

[2] http://www.dlr.de/eoc/desktopdefault.aspx/tabid-5431/9230_read-42467/

## II. Related work

The vehicle detection in aerial images has a large literature, here we mention only a few important recent papers.

Moranduzzo and Melagni [3], [4] process very high resolution (2 cm GSD) UAV images for car detection. In [3] a feature point detector and SVM classification of SIFT descriptors is applied, while the method in [4] uses a catalog of HoG descriptors and later an orientation estimation.

In [5] the cars are detected by a deep neural network running on the GPU in a sliding window approach on a known constant scale. In [6] the vehicles are detected with online boosting on Haar-like features, local binary patterns and orientation histograms. They train the detector for cars in one direction and during testing they rotate the image in 15 degrees step. This detector is trained for a known object size $35 \times 70$ pixels and tested on images with the same scale.

Leitloff et al. [1] use a two stage approach for the detection of cars: first an AdaBoost classifier with Haar-like features and then an SVM on various geometric and radiometric features. They use the road database as a prior to detect only along the roads in a certain direction. The method achieves good results running fast on a CPU, however it is limited to orthorectified images and areas covered by the road database.

Tuermer et al. [7] utilize the road map and stereo matching to limit the search area to roads and exclude buildings. HOG features with an AdaBoost classifier are applied to detect the cars on the selected region. This method is limited to georeferenced image pairs and areas covered by the road database.

## III. Multi-direction Vehicle Detection

We handle the vehicle detection problem in two stages. The first stage is a very fast binary sliding window object detector which delivers axis aligned bounding boxes of the vehicles without type or orientation information. The second stage is a multiclass classifier applied on the bounding boxes estimating the orientation and the type of the vehicles. The processing steps are shown in Fig. 1.

### A. Binary sliding window detector

For fast detection both the feature calculation and the classification has to be efficient.

*1) Fast image features:* Viola and Jones [8] introduced the integral image concept with Haar-like features for fast and robust face detection. By using the integral image $I_\Sigma$ the pixel intensity $I$ sum of the Haar-like features is calculated by a few operations independent of the area of the feature. The value $I_\Sigma(x,y)$ at $(x,y)$ location in an integral image is the sum of the pixels above and to the left of $(x,y)$:

$$I_\Sigma(x,y) = \sum_{i=0}^{i\leq x} \sum_{j=0}^{j\leq y} I(i,j) \tag{1}$$

The integral $f_I$ within an axis aligned rectangle defined by its upper left corner $x_0, y_0$, width $w$ and height $h$ is calculated as $f_I = I_\Sigma(x_0+w, y_0+h) + I_\Sigma(x_0,y_0) - I_\Sigma(x_0+w,y_0) - I_\Sigma(x_0, y_0+h)$.

This idea is generalized by the Integral Channel Features (ICF) in the work of Dollar et al. [9]. Instead of working on pixel intensity values as in [8], an *ICF* can be constructed on top of an arbitrary feature channel (i.e. the transformation of the original image). Features are defined as linear combinations of sums over local rectangular regions in the channels. By using the concept of integral images, an integral channel can be pre-computed for each feature channel so that the computation of the sum over the rectangle is very fast. The most commonly used channels are the color intensities, the gradient magnitude and the gradient histogram. The gradient histogram is a weighted histogram where the bin is determined by the gradient orientation. It is given by $Q_\Theta(x,y) = G(x,y)\mathbf{1}[\Theta(x,y) = \theta]$, where $G(x,y)$ is the gradient magnitude and $\Theta(x,y)$ is the quantized gradient orientation at $x, y$ image location. The gradient histogram can approximate the powerful and widely used HOG features [2]. If the rectangles are defined as squares, the sum can be aggregated to a single pixel in a downsampled image. In this case the integral is calculated even faster as a single pixel look up. This method is also called Aggregated Channel Features (ACF) [10]. For rapid speed we apply this method with fast feature pyramid calculation as described in [10].

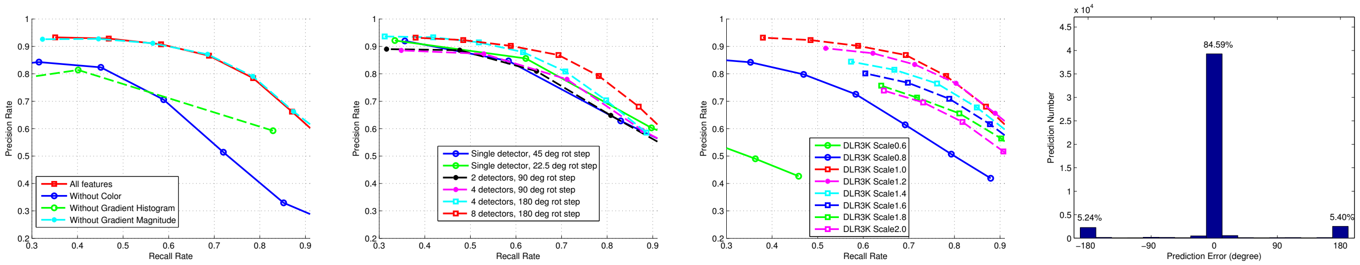
*2) AdaBoost classifier in Soft Cascade structure:* The number of ICFs is very large (larger as the number of pixels in the image window) since it is the linear combination of local rectangular regions in the image window. We select only relevant features by the Discrete AdaBoost algorithm [11] for $N$ weak classifiers $h_t(\mathbf{x})$. $h_t(\mathbf{x})$ is a simple classifer, e.g. a threshold or a shallow decision tree of a few features from the input feature vector $\mathbf{x}$. AdaBoost is an iterative algorithm, in each step it reweights the samples in the training set according to the classification result from the previous weak classifier. The final strong classifier $H$ is composed of the weighted $\alpha_t$ weak classifiers $h_t(\mathbf{x})$.

$$H = \text{sgn} \sum_{t=1}^{N} \alpha_t h_t(\mathbf{x}) \tag{2}$$

At numerous sliding window positions (e.g. homogeneous regions) not all the weak classifiers have to be evaluated to classify the image as non vehicle. To leverage this property for speed improvement we form a Soft Cascade [12] from the weak classifiers. During the training a threshold $r_t$ is set for all the weighted weak classifiers $c_t = \alpha_t h_t(\mathbf{x})$. If the cumulative sum $H_t(\mathbf{x}) = \sum_{i=1,\ldots,t} c_i(\mathbf{x})$ of the first $t$ output functions is $H_t(\mathbf{x}) \geq r_t$, then input sample is passed to the subsequent evaluation process; otherwise it is classified as negative and rejected immediately.

### B. Multi-direction detection

The orientation of the vehicles in aerial images can be arbitrary. This increases the intra-class variation of the appearance in the axis aligned sliding windows. A straightforward but computationally expensive solution, used in [6], is to train the detector for one specific direction and rotate the input image and do detection for each rotation. This would need the computation of the integral images separately for



(a) feature contribution (b) classifier configurations (c) different scales (d) orientation estimation

Fig. 2. (a) Evaluation of the Integral Channel Features. Gradient histogram channels play the most important role while gradient magnitude channel has least affects on the final result. (b) Detection result of aggregated detectors. (c) Performance after rescaling the image with different factors. (d) Orientation estimation error histogram using artificial neural network with 16 output classes.

each direction and would result in slow processing speed. To overcome this we propose two methods: One is to train a single classifier which is able to detect differently oriented vehicles; The other is to aggregate several simple classifiers, where each is only sensitive to specific directions.

*1) Single classifier method:* A single binary classifier is trained with samples covering all the directions. The training process has to deal with the high intra-class variety and find the common part of all the positive samples. When the detector is applied on the input image, vehicles in any directions can be classified as positive samples.

*2) Aggregated classifier method:* Alternatively the intra-class variety is reduced by splitting the training to different orientations. Multiple binary classifiers are trained, each for specific vehicle orientations. These classifiers are employed in sequence during the detection phase, and the results from each classifier are aggregated using non-maximal suppression. The integral image does not need to be calculated multiple times, only the classification.

The performances of these two methods are examined in Section V.

## IV. Multiclass Vehicle Classification

The detector provides the axis aligned bounding boxes of the vehicles. In this next step we refine the extracted information by classifying the orientation and the type of the vehicle. We propose a two-step approach containing an orientation estimator and a type classifier. A sample is sent to the orientation estimator first, then rotated to horizontal direction according to the orientation estimation, and finally processed by the type classifier to identify which type category this vehicle belongs to.

### A. Orientation estimation

We consider the orientation estimation as a multi-class classification problem. The directions are clustered, each cluster is considered as a class. The ICF features can be calculated fast, but they have a very high number, thus they are not suitable for multiclass classifiers working on a fixed length feature vectors. Therefore we apply the powerful Histogram of Oriented Gradients (HOG) feature [2] which has a fixed feature vector length. We use a neural network with one hidden layer as a multi-class classifier [13].

### B. Type classification

The type classifier needs to classify the input image into corresponding categories. We have defined two type classes: car and truck but the presented method could be extended to more classes. The object bounding box is rotated to horizontal direction based on the orientation estimation. Unrelated context is cropped out and HOG features are again extracted and classified by the type classifier.

## V. Experimental results

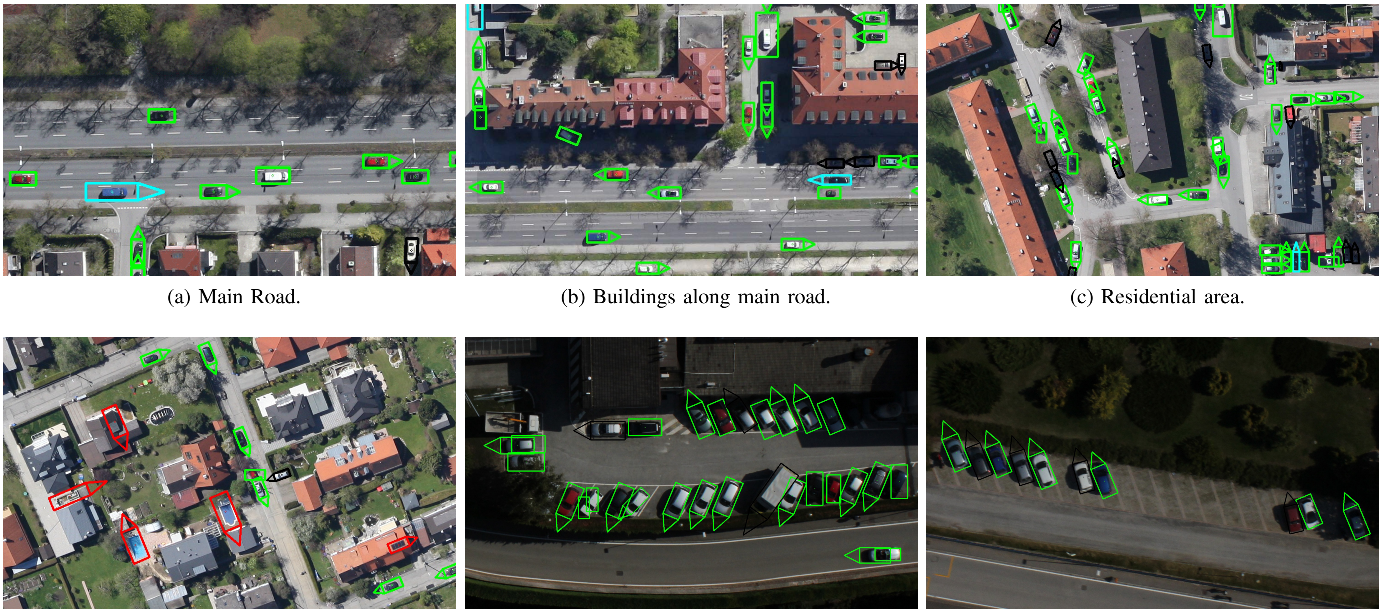
We test the multi-direction detection and multiclass classification parts in our detection method, respectively, and give quantitative results for the different processing stages. The binary detector is trained with 2048 weak classifiers in each test. We use depth-two decision trees as weak classifiers.

### A. Results on Munich images

The quantitative evaluation is performed on 20 aerial images captured by the *DLR 3K* camera system [1] over the area of Munich, Germany. We use the original nadir images with the resolution of 5616 × 3744 pixels. They are taken at a height of 1000 meters above the ground, the approximate ground sampling distance is 13 cm. The first 10 images are used for training and the other 10 for testing. Positive training samples come from 3418 cars and 54 trucks annotated in the training images, while the negatives are randomly picked from the background, i.e. areas without vehicles. To overcome the low number of truck samples we randomly transformed them additionally 30 times. Fig. 3 shows detection results on the test images. We set the detection window to 48 × 48 pixels. For the ground truth the vehicles in the images are annotated manually as oriented bounding boxes.

*1) Multi-direction vehicle detection:* Integral Channel Features contain rich information and can be computed rapidly. They are selected as the features for training and detection. Experiments are performed to evaluate the importance of each feature channel type and the performance of different classifier configurations.

*a) Feature channel:* We use three types of feature channels: Luv color, gradient magnitude and gradient histogram. We have evaluated the contribution of each feature channel, the Precision-Recall (PR) curves are plotted on Fig. 2 (a). These curves indicate that gradient histogram channels play the most



(a) Main Road.

(b) Buildings along main road.

(c) Residential area.

(d) Failure cases.

(e) Detection on dataset in [3], [4]

(f) Detection on dataset in [3], [4]

Fig. 3. Detection results from the DLR test images. Green and cyan bounding boxes are the correct detected samples, representing cars and trucks, respectively. Black bounding boxes are the missed ones and red are the false positives. The results show that our method works well in most scenarios (a)(b)(c), however the complicated rooftops or outdoor swimming pools may lead to false positive detections (d). We also evaluated our method on the dataset presented in [3], [4], the detection results are shown in (e)(f).

important role in representing the vehicles while the gradient magnitude channel affects the final result the least. For the later tests we use all the feature channels.

*b) Multi-direction detection methods:* We proposed two methods, single and aggregative classifiers, to detect vehicles in different directions (Section III-B). The performances are depicted in Fig. 2(b). The PR curve shows that the optimal solution is the 'Classifier aggregation method' with each classifier trained using samples in opposite directions (8 detectors with sample rotation step of 180° ). This means 8 detectors and thus longer computation time. 2.7 s is needed for a single detector while the detection with 8 classifiers takes 4.1 s. This is sublinear since the integral images doesn't have to be calculated again. We use the 8-classifier configuration for the later tests.

*c) Detection on images with different scales:* To show the ability of our method to detect the cars on images with different scales we resized the image for the test but not the training. These results are shown on the Fig. 2(c). The detector performs best on the same scale as it was trained, if the resolution is increased the performance remains comparable. But if we decrease the resolution we lose information which leads to a lower performance.

*2) Multi-class vehicle classification:* After the axis-aligned bounding box detection we classify the orientation and type of the vehicles. We convert all the bounding boxes to $48\times48$ pixel gray images and calculate HOG features for this image. We get the best performance with $4\times4$ cell size, $1\times1$ block size, $1\times1$ block stride HOG feature configuration and use this for the later tests. The comparison of different HOG configurations can be found in the supplementary material.

TABLE I
PERFORMANCE COMPARISON BETWEEN DIFFERENT METHODS.

| Method | Ground Truth | True Positive | False Positve | Recall Rate | Precision Rate |
|---|---|---|---|---|---|
| Munich dataset | | | | | |
| Viola-Jones | 5892 | 3237 | 1467 | 54.9% | 68.8% |
| Ours | 5892 | **4085** | **619** | **69.3%** | **86.8%** |
| UAV dataset from [3] [4] | | | | | |
| [3] | 119 | 88 | 143 | 73.95% | 38.1% |
| [4] | 119 | 87 | 111 | 73.1% | 43.4% |
| Ours | 119 | **94** | **6** | **79.0%** | **94.0%** |

TABLE II
COMPARISON OF COMPUTATION TIMES.

| Method | Image Resolution | Detection Time Per Image [s] | Detection Time Per MPixel [s] |
|---|---|---|---|
| Proposed | 5616 × 3744 | **4.4** | **0.2** |
| Viola-Jones | 5616 × 3744 | 1160 | 55.2 |
| [4] | 5184 × 3456 | 14400 | 803.8 |
| [5] [a] | 1368 × 972 | 8 | 6.0 |

[a] Running on the GPU.

*a) Orientation estimation:* Orientation classification is performed according to Section IV-A with 16 classes (22.5° rotation difference between adjacent sample groups, respectively). The orientation estimation error histogram is depicted in Fig. 2(d). In the supplementary material we provide results with different number of classes and an additional random forest classifier [14]. The most common error is when the samples are classified in the opposite direction. This is because

TABLE III
CONFUSION MATRICES OF TYPE CLASSIFICATION USING DIFFERENT CROPPING CONFIGURATIONS.

| **Cropped Size** | $48 \times 48$ | | | $\mathbf{48 \times 28}$ | | |
|---|---|---|---|---|---|---|
| **Confusion Matrix** | A/P [a] | Car | Truck | A/P | Car | Truck |
| | Car | 2843 | 60 | Car | **2838** | **65** |
| | Truck[b] | 123 | 685 | Truck | **0** | **808** |
| **Accuracy** | 95.1% | | | **98.2%** | | |

[a] Actual class / Predicted class
[b] The number of truck type is increased by random transformation of the existing samples.

the front part of a vehicle might be similar to the rear part from the top view in aerial images.

*b) Type classification:* The detected bounding box is rotated to the horizontal direction according to the orientation estimation. We trim the input image by cropping the upper and lower parts, from $48 \times 48$ to $48 \times 28$. In our dataset the number of trucks is much less than the number of cars. We generate new ones from the existing samples using random transformation. The performances with different cropping configurations are compared in table III and the supplementary material. The optimal type classification can reach 98.2% in accuracy with a one-hidden-layer neural network.

*3) Baseline comparison:* As baseline we use the OpenCV[3] implementation of the Viola-Jones detector [8]. We have trained it on one vehicle direction while at detection we rotate the image similar as in [6] and apply the detector for each rotated image. Table I contains the numerical comparison of this method on the Munich dataset.

*4) Computation time:* Since the processing time is also important for the detector we compare our method with other methods where the computation time is provided in the paper. Table II contains the computation times. Our experiments are performed on a laptop with Intel® Core™ i5 processor and 8 GB memory and our program is running single threaded written in Matlab and C++. The comparisons show that the speed our method is considerably faster. This makes our method more suitable for real-time systems where the computation time is a serious issue. The method of [5] achieves comparable detection performance but on a different dataset, therefore we show only the processing time of the method.

### *B. Baseline comparison on UAV images*

We also evaluated our method on the dataset presented in [3], [4] and compared to the results provided without screening. The results can be found in the Table I. The precision rate of our method outperforms the other methods significantly. Due to the higher resolution we set the detection window to $96 \times 96$ pixels for this dataset and have only car vehicle type (no truck).

### *C. Qualitative results from around the world*

To show the robustness of our detector we also run our detector on images downloaded from Google Earth. These can be found in the supplementary material.

## VI. CONCLUSION

We have presented a method which can detect vehicles with orientation and type information on aerial images in a few seconds on large images. The application of Integral Channel Features in a Soft Cascade structure results in both good detection performance and fast speed. The detector works on original images where no georeference and resolution information is available. As future work the performance could be further improved by using a deep neural network after the binary detector like R-CNN in [15]. Since this has to be applied only to a fraction of the image, the speed of the detector would be still fast.

## VII. ACKNOWLEDGEMENT

This work was funded by the DLR project *Vabene++*[4]. We thank for the authors of [3], [4] who generously provided their image dataset with the ground truth.

## REFERENCES

[1] J. Leitloff, D. Rosenbaum, F. Kurz, O. Meynberg, and P. Reinartz, "An operational system for estimating road traffic information from aerial images," *Remote Sensing*, vol. 6, no. 11, pp. 11 315–11 341, 2014.

[2] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

[3] T. Moranduzzo and F. Melgani, "Automatic car counting method for unmanned aerial vehicle images," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 52, no. 3, pp. 1635–1647, March 2014.

[4] ——, "Detecting cars in uav images with a catalog-based approach," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 52, no. 10, pp. 6356–6367, Oct 2014.

[5] X. Chen, S. Xiang, C. Liu, and C. Pan, "Vehicle detection in satellite images by hybrid deep convolutional neural networks," *Geoscience and Remote Sensing Letters, IEEE*, vol. 11, no. 10, pp. 1797–1801, Oct 2014.

[6] S. Kluckner, G. Pacher, H. Grabner, H. Bischof, and J. Bauer, "A 3d teacher for car detection in aerial images," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, 2007, pp. 1–8.

[7] S. Tuermer, F. Kurz, P. Reinartz, and U. Stilla, "Airborne vehicle detection in dense urban areas using hog features and disparity maps," *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, vol. 6, no. 6, pp. 2327–2337, Dec 2013.

[8] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1, 2001, pp. I–511–I–518 vol.1.

[9] P. Dollár, Z. Tu, P. Perona, and S. Belongie, "Integral channel features," in *BMVC*, vol. 2, no. 3, 2009, p. 5.

[10] P. Dollar, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 8, pp. 1532–1545, Aug 2014.

[11] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting," *Annals of Statistics*, vol. 28, p. 2000, 1998.

[12] L. Bourdev and J. Brandt, "Robust object detection via soft cascade," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 2. IEEE, 2005, pp. 236–243.

[13] Y. Lecun, L. Bottou, G. B. Orr, and K. R. Müller, "Efficient BackProp," in *Neural Networks—Tricks of the Trade*, ser. Lecture Notes in Computer Science, G. Orr and K. Müller, Eds. Springer Verlag, 1998, vol. 1524, pp. 5–50.

[14] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[15] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[3] http://opencv.org/

[4] http://www.dlr.de/vabene/