Vehicle Detection in Aerial Imagery : A small target detection benchmark

Sébastien Razakarivony, Frédéric Jurie

Vehicle detection in aerial Imagery : A small target detection benchmark

Sebastien Razakarivony[b,a,1], Frederic Jurie[b]

[a] *Safran, 1 rue Genevive Aube, 78114 Magny-les-Hameaux, France*
[b] *University of Caen – CNRS UMR 6602 – ENSICAEN, 14000 Caen, France*

Abstract

This paper introduces *VEDAI: Vehicle Detection in Aerial Imagery* a new database of aerial images provided as a tool to benchmark automatic target recognition algorithms in unconstrained environments. The vehicles contained in the database, in addition of being small, exhibit different variabilities such as multiple orientations, lighting/shadowing changes, specularities or occlusions. Furthermore, each image is available in several spectral bands and resolutions. A precise experimental protocol is also given, ensuring that the experimental results obtained by different people can be properly reproduce and compared. Finally, the paper also gives the performance of baseline algorithms on this dataset, for different settings of these algorithms, to illustrate the difficulties of the task and provide baseline comparisons.

Keywords: Detection, Low resolution images, Vehicles, Database, Aerial imagery, Infrared imagery, Computer vision.

1. Introduction

Automatic Target Recognition (ATR), which is the task of automatically detecting targets in images, has an history of more than 35 years of research and development in the computer vision community. The basic aim of such ATR systems is to assist or remove the role of man from the process of detecting and recognizing targets and hence to implement efficient and reliable systems of high performance. One typical application is surveillance and reconnaissance, two tasks which need to be more and more automatic, as

*Corresponding author, sebastien.razakarivony@safran.fr

recent high resolution surveillance sensors produce imagery with high data bandwidth. As explained by Wong [1], a surveillance mission over a 200 mile square area with a one foot resolution (an appropriate size for recognizing many targets), will generate approximately 1.5×10^{12} pixels of data. If the area is split in 10 million pixels images, photo interpreters would have to examine over 100,000 images, which is an impractical workload and results in a delayed or incomplete analysis. In addition, the delay would allow movable targets to relocate so that they cannot be found in subsequent missions. Vehicle detection is hence of crucial matter in defense applications.

Despite the aforementioned very long history of ATR in the computer vision literature, it is still a challenging problem even with the most recent developments of this area. This is demonstrated by the figures given in the experiments section of this article.

The traditional way to address ATR consists in the following pipeline [2]: (i) preprocessing, which consists in improving target contrast and reducing noise and clutter (ii) target detection, *i.e.* the process of localizing the area in an image where a target is likely to be present, often done by computing image regions with high contrasts (iii) segmentation, which consists in accurately extracting the potential targets from the background and (iv) recognition, consisting in extracting visual features from these potential target and finally classifying them.

Modern approaches for automatic object detection uses a rather different paradigm. Indeed, they try to avoid taking intermediate decisions by relating directly the input space with the final decision space and make extensive use of machine learning techniques. Two prototypical examples are the face detector of Viola and Jones [3] based on the use of Haar wavelets and a cascade of boosted classifiers and the Dalal and Triggs's pedestrian detector [4] using Histogram of Oriented Gradients (HOG) combined with Support Vector Machine (SVM) classifiers. The popular bag-of-words model [5] has also been used successfully for object detection [6]. The combination of such efficient machine learning algorithms with discriminative features is the foundation of modern object detection algorithms. More improvement has also been done recently by using more complex object models, such as the Deformable Parts Model [7].

One reason to explain the progress in this field is the release of publicly available datasets allowing the development, the evaluation and the comparison of new algorithms in realistic conditions. PASCAL VOC [8] benchmark provides one of the key datasets for object detection. From 2005 to 2013,

yearly evaluation campaigns have been organized. The detection competitions of PASCAL VOC consist in predicting the bounding box and the label of each object from twenty possible target classes in the test image. In 2012, a total of more than 10,000 annotated images were available for training and validation. Several other datasets, presented in the related work section, are available for the evaluation of different detection tasks (*e.g.* person detection, face detection), such as ImageNet [9] or LabelMe [10].

However, none of these datasets is actually adapted to ATR. Indeed, one specificity of ATR is to require the detection of small targets while these dataset includes objects whose size in images is usually bigger than 200 pixels and can be the main topic of the image. These recent datasets are more concerned by the diversity of object appearance, articulated objects, number of categories than by target size, image noise, multi-spectral images, sensor technology.

On the other hand, and as far as we know, none of the recent approaches for object detection (*e.g.* [4, 7, 11]) have been evaluated in the context of ATR.

Within this context, the motivation for this paper is twofold. First, the paper introduces VEDAI (Vehicle Detection in Aerial Imagery), a new database designed to address the task of small vehicle detection in aerial images within a realistic industrial framework. [1] This dataset was made to help the development of new algorithms for aerial multi-class vehicle detection in unconstrained environment, giving the possibility to evaluate the influence of image resolution or color band on detection results. Images includes various backgrounds such as woods, cities, roads, parking lot, construction sites or fields. In addition, the vehicles to be detected have different orientations, can be altered by specular spots, occluded or masked. No specific constraints were put on the types of vehicles. This diversity of backgrounds and vehicle appearances will allow to make progress in the field of automatic scene analysis, scene surveillance and target detection. Second, we benchmark some baseline algorithms and show their performance on the proposed dataset, to allow people to have some point of comparison.

The organization of the paper is as follows. After presenting the related works in Section 2, we introduce the dataset (*i.e.* the images, the vehicle classes as well as the background types, the annotations and the organiza-

[1] The database can be downloaded at https://downloads.greyc.fr/vedai/

tion of the dataset) in Section 3. To make comparisons between algorithms possible, we give in Section 4 the evaluation protocol associated with the dataset. We finally present in the last section (Section 5.2) experiments in which baseline algorithms are evaluated on the dataset, giving baseline results and some analysis of the influence of the parameters on the performance.

2. Related works

Object detection – often considered as being one of the most challenging computer vision task – has a long history in the computer vision literature. This section focuses on three aspects of the problem, namely (i) the datasets publicly available to develop, validate and compare object detectors, (ii) the different ways to measure detection performance, (iii) the current state-of-the-art approaches for object detection.

2.1. Databases for object detection

Modern approaches in computer vision rely on machine learning and require annotated training data. In addition, there is also an increasing need for comparing approaches with each other and establishing what are the most promising avenues. The consequence is that a lot of new datasets have been recently produced and made publicly available. If most of them are related to object/scene recognition (*e.g.* [12, 13, 14, 15, 16]) – which is related to our problem but covers different needs – only a few of them specifically address object detection.

More precisely, datasets for detection usually fall into the following categories: (i) pedestrian detection (ii) face detection (iii) detection of everyday objects (iv) vehicle detection. A summary of these datasets is given Table 1.

Person/pedestrian detection. This is one of the very popular detection task, probably because of the large number of applications (surveillance, indexing, traffic safety, etc.) that may result. The INRIA person dataset, first introduced in [4], contains several hundreds cropped images of humans with different resolution (64x128, 70x134, 96x160). Images are also provided with the whole background and the base is separated in train and test sets. The images come from various sets of personal photos and a few from the web. The people appear in any orientation and among a wide variety of backgrounds. Many people are bystanders taken from the backgrounds of the input photos, so ideally there is no particular bias in their pose. This dataset

was introduced because the previous dataset of reference – the MIT person dataset [17] – was not challenging enough.

The CalTech pedestrian dataset [18] has been introduced once the INRIA person dataset was considered to be too small and too easy and addresses more specifically the case of pedestrian detection. It is a collection of images taken from a vehicle driving through regular traffic in an urban environment. It contains 350,000 labeled pedestrian bounding boxes in 250,000 frames. Occlusions are annotated with a two bounding box system and annotations are linked between frames, forming tracks.

The increase of the number of images between these two datasets (from hundreds to thousands) reflects a current trend in the production of datasets.

Face detection. Face detection is another well known detection task. Contrarily to pedestrian detection and despite the fact that it is often considered as an important task related to interesting applications such as security or safety, only a few datasets exist. Most of the existing face-related databases are indeed oriented toward face recognition (*e.g.* [15]) and not face detection. The CMU-MIT dataset [19], which includes the MIT dataset [20], is one of the dataset of reference, extensively used in the past. It contains only 130 different images for a total of 507 different faces (front view only). Moreover, this dataset is small and the evaluation protocol and the metric are not clearly defined. The results presented by the numerous papers using it cannot be compared in a reliable way, as noticed by [21]. More recently, Kodak has compiled and released a new image database for benchmarking face detection and recognition algorithms [22]. This database has 300 images of different sizes, the size of faces in images varying from 13×13 pixels to 300×300 pixels. Finally, the most used and well-known dataset in face detection is Face Detection Dataset Benchmark (FDDB, [23]). It is made from images extracted from Faces in the Wild dataset ([24]). It is worth noting that FDDB removed all faces that are smaller than 20 pixels width or height.

Everyday life objects. Some other databases have a more general purpose and mix several object classes, often taken from the everyday life. For example the ETHZ dataset [25] contains 5 different classes (namely 'logo', 'bottle', 'giraffe', 'mug' and 'swan'), which are characterized by specific shapes. It contains distinct train and test subsets. The images have been downloaded from Flickr and Google Search Images. The dataset does not only include photos but also paintings, drawings and computer rendered images. Around

Figure 1: Some vehicles from PASCAL VOC 2007 dataset [8]

40 images are available for training and 40 for testing, per class. Another dataset containing everyday life objects is the dataset from Pascal VOC challenge [8], which is one of the most famous challenge in computer vision. This challenge includes a detection task. The latest editions of the Pascal VOC rely on a dataset of 20 classes, for a total of several thousands images, split into train, validation and test sets. The original images were taken from other publicly available datasets as well as from websites such as Flickr. A third multi-class challenge is proposed with the LabelMe dataset [10]. LabelMe is different from other database because it can be annotated by anyone. The precise number of classes is therefore changing over time. In December 2006, it was containing 111,490 polygons on 11,845 static images and 18,524 video sequences. Many images still need to be annotated. Finally, it worth to mention the recent ImageNet dataset [9], which is popular at the moment, and can be used for object detection. It contains more than 14 millions of images.

Images such as image from Flickr, commonly used in publicly available datasets, are photographies taken with consumer cameras. If they are obviously suited to the evaluation of image retrieval algorithms, the context is too different from aerial surveillance. In addition, objects of interest are often the main subject of the pictures e.g. vehicles are often in the center of the image and are very large. To illustrate this point, some vehicles from PASCAL VOC are shown Figure 1. Regarding the larger ImageNet dataset, which is not designed at all for surveillance and security application, it does not contain any annotations of vehicles in aerial image nor infrared images. We did not really find images that would fit the task covered by this paper, except some aerial images for which vehicles are on roads or highway but are to small to be detected in a reliable way, even for humans.

Vehicle detection. As shown by the works of [27, 28], the databases we presented so far have some intrinsic bias, which can be explained by how the

Figure 2: Four illustrative OIRDS images [26].

images have been collected. Furthermore, they are very different from the task considered in this paper. Closer to our area of interest, vehicle detection has also received a lot of attention during the last decade. Despite the existence of available vehicle databases, most of these bases contain vehicles seen from the ground and the vehicle is the main topic of the image (e.g. INRIA Car dataset [29]). Some (even closer) works on target detection in aerial imagery use aerial databases ([30, 31]), but unfortunately they are not publicly available. Consequently, to our knowledge, none of the results addressing vehicle detection in aerial images are reproducible.

We can mention the work of [32], which regroups 9 sequences dedicated to the tracking of vehicles in aerial imagery. However there is only 9 sequences with max 50 images. Furthermore, all of them are in an urban environment. To our knowledge the only available dataset is OIRDS (Overhead Imagery Research DataSet) [26], which contains 180 vehicles in 900 annotated im-

Database	Classes	Folds	# Images	Eval.
Inria Pedestrian	1	train/test	2,000	DET+ OP
CalTech Pedestrian	1	train/test	250,000	MR/FPPI+ OP
CMU	1	no protocol	130	no protocol
FDDB	1	1O folds	2845	ROC curves
ETHZ	6	train/test	500	rec/FPPI
PASCAL	20	train/val/test	$\geq$10,000	AP+prec/rec
LabelMe	$\geq$400	no cut	$\geq$40,000	no protocol
ImageNet	21841	train/val/test	$\geq$14,000,000	AP+prec/rec
OIRDS	4	no cut	900	no protocol
Proposed Dataset	9	train/test	1200	AP+OP

Table 1: Summary of existing databases for object detection. DET means "Detection Error Trade-off", MR means "Miss Rate", OP is for "Operating Points", FPPI is the abbreviation of "False Positive Per Image", rec is used for "Recall", prec for "Precision", AP means "Average Precision", no protocol means that no protocol is given.

ages. It contains five classes of vehicles ('truck', 'pick up', 'car', 'van' and 'unknown'); annotations give information such as color, specularity, distance to the ground. Some images of the OIRDS dataset are given Figure 2. However, this database has two issues, which make it hard to use to benchmark target detection algorithms. First of all, no evaluation protocol is defined. The risk is hence that each evaluation can use different images for training or testing and use different scores. Second, the dataset is obtained by aggregating multiple sources of images (20 different sources) and does not have enough statistical regularity. Indeed, there are 20 different sources for 900 images, with only 45 images per source on average, which is too limited.

These issues make the results difficult to reproduce, preventing other researchers to make any comparisons with this work. For example, the authors of [33], which use this dataset, used their personal split of the database (easy, medium and hard) but the precise set of images in each split is not defined, preventing from reproducing the results. As well, [34] uses this dataset, but gives only qualitative results on it.

Performance evaluation

Fair empirical evaluations of detection algorithms can be done if and only if a clear evaluation protocol is defined. In addition to the datasets, a metric must be also specified. This section reviews the different performance metrics used in the object detection literature.

First, as pointed out by [18], we can oppose the *per windows* performance used in several papers on object detection (*e.g.* [4]) to the *per image* performance, which is the norm for object detection [35]. The *per windows* case assumes that the detection task is evaluated by classifying cropped windows centered on objects of interest against windows from images without the object. The typical assumption is that better per window scores will lead to better performance on entire images, but [18] shows that, in practice, per window performance can fail to predict per image performance.

As explained by [36], Receiver Operator Characteristic (ROC) curves [37] are commonly used (*e.g.* [3, 19]) to present results for binary decision problems. Please, note that even if object detectors often output probabilities of scores, they are usually thresholded (using a discrimination threshold), according to different operational points and hence considered as binary. The ROC curve is a graphical plot which illustrates the performance of a binary classifier system as the discrimination threshold is varied. It is created by plotting the fraction of true positives out of the positives (TPR = true positive rate) vs. the fraction of false positives out of the negatives (FPR = false positive rate), at various threshold settings. The ROC curve is sometimes substituted by the Detection Error Tradeoff (DET) curve (*e.g.* [4, 38]), plotting False Reject rate vs. False Acceptation rate. It contains the same information as ROC curves, but using 1-TP instead of TP. The $x-$ and $y-$axes are scaled non-linearly by their standard normal deviates, yielding tradeoff curves that are more linear than ROC curves, helping to see small differences when performance is high.

However, ROC curves can present an overly optimistic view of an algorithm's performance if there is a large skew in the class distribution. This is typically the case for object detection using sliding windows, as the vast majority of the windows to be processed are negative windows while only a very few windows per images are positive. It is even more true in surveillance application, as you may have hundreds of images without any vehicles to detect. Consequently, a large change in the number of false positives can lead to a small change in the false positive rate used in ROC analysis.

When dealing with highly skewed datasets, Precision–Recall (PR) curves [39] give a more informative picture of an algorithm's performance. Recall is defined as the fraction of correctly predicted true positive (TP) over the total of positive (P), *i.e.* $Recall = \frac{TP}{P}$, while Precision is the fraction of true positive (TP) among all the positive predictions (false positive FP plus true positive TP), *i.e.* $Precision = \frac{TP}{TP+FP}$. The Precision–Recall curves are used

to evaluate most of the object detection algorithms, such as [40, 41, 42, 43].

Instead of representing the performance by a curve, it is often more appropriated to use particular operational points adapted to a given application. A common measurement is the False Positive Per Image rate (FPPI) and the False Positive Per Window rate (FPPW) for a given threshold. For example, in [4], the operating point giving the recall at 10^{-4} FPPI is chosen while [18] chose 1 FPPI. Once a characteristic is fixed (*e.g.* the FPPI) the performance is given by the other one (*e.g.* the recall). This characterizes the behavior of the algorithm for the expected FPPI or FPPW. In industrial context, algorithms need to meet specifications; it explains why this metric is often used in such contexts by companies. Some other operating points are sometimes used, as for example the equal error rate, which is the point where FPR is equal to Recall. This choice is often arbitrary and chosen because easy to be computed.

Another way to represent detection algorithms performance by scalar numbers instead of curves is to use the area under the curve *e.g.* the Area Under the ROC curve or AUROC. The better an algorithm is, the closer to 1 its AUROC is. In the same way, the Average Precision is the area under the precision–recall curve. Usually, the Average Precision is computed on an interpolated curve of 11 points, as explained by [44]. Despite it is not adapted to the evaluation of surveillance tasks – for which using an operational point is more relevant – this measure is included in our protocol as it has been used extensively in the recent computer vision literature (*e.g.* in PASCAL VOC [8] which is a standard).

In addition the evaluation protocol has to specify when a prediction is a true positive, *i.e.* when the prediction and the object match. Detection algorithms usually output rectangular regions (bounding boxes) around detected objects. Predictions are then (generally) considered as being correct if both the position and the scale of the predicted bounding boxes are correct. This is often done by computing the Jaccard index (*e.g.* [8]), defined as the area of the intersection divided by the area of the union of the two rectangular bounding boxes (ground truth and prediction). Some other works simply measures the overlap (*e.g.* [33]) between both. When several objects are on the image, multiple detections can be assigned to the same ground truth annotation, *e.g.* when several objects are in line one beside the other. This can be handled in different ways, such as by building a bipartite graph [45, 46] or by relaxing of the overlap criterion [47, 48].

Finally, the use of rectangular bounding boxes raises some issues, as noted

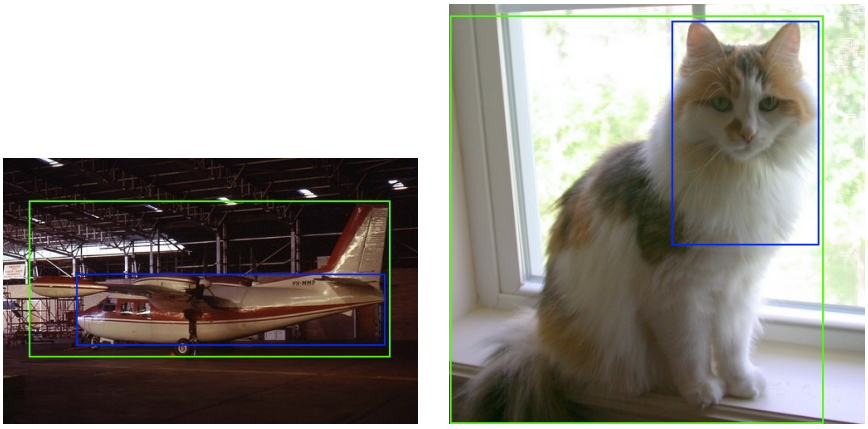

Figure 3: Left hand-side: the prediction (blue box) is counted as false positive according to the Jaccard index, as the ground truth (green box) covers non object. Right hand-side: while the prediction (blue box) cover the cat and only the cat, it is counted as a false positive as it does not cover it entirely. Viewed from another point of view, this area could be counted a true positive.

by [49], because the shape of the object is in general not rectangular as shown by Figure 3.

2.2. State-of-the-art object detection approaches

Sliding window classification is the dominant paradigm in object detection. It consists in independently classify all sub-windows as being object or non-object and selecting predominant bounding boxes after a non maximal suppression stage.

The approaches differ in the classifier they use, which is in general either a boosted classifiers (*e.g.* [3, 50, 51]), SVM classifiers (*e.g.* [4, 52, 7, 53] or neural networks (*e.g.* [54]) and more recently, Convolutional Deep Networks[11, 55, 56, 57, 58].

The classifiers use a feature based representation, embedding some invariance to basic transformations such as small shift or illumination variations. The most efficient features are Haar-like wavelets (*e.g.* [59]) computed from integral histograms (*e.g.* [60]), edgelets (*e.g.* [51]), shapelets (*e.g.* [61]), but also histogram of gradients (HOG) (*e.g.* [4, 62]), bag-of-words (*e.g.* [63]), multi-scale spatial pyramids [64], covariance descriptors (*e.g.* [38]), co-occurrence features (*e.g.* [65]), local binary patterns (*e.g.* [66]), color-self-similarity (*e.g.* [67]), or combinations of such features (*e.g.*

[66, 50, 67, 65, 68]). Regarding deep learning approches, the features are learnt at the same time as the classifier.

If most of the sliding window based approaches consider the object (and its associated templates) as being rigid, some recent approaches have successfully investigated models in which the objects are represented by the combination of different parts, which are spatially organized. We can mention, for example, the Deformable Part-based Models (DPM) [7, 53, 61] based on the pictorial structures of [69], the poselet model [70] or the mixture of parts of [71].

One way to make the sliding window faster – following the success of Viola and Jones [3] – is to use cascades. Cascades have been successfully applied to the star-structured models of [7] in [43] and [42] for this purpose. It is also possible to skip the windows that are likely to contain none of the targets by alternative approaches, such as the use of interest points in the form of a cascade of jumping windows as proposed by [72], the use of saliency operators like *objectness* [73, 74] or the use of segmentations [41, 40] as candidates to object-like window hypotheses.

A couple of works are specifically addressing vehicle detection. The work of [30] make the comparison between different methods, but uses classic features and classic classifier as well. They also use color information, which cannot be used in case of infra-red images. The work of [75] use a pixel classification approach, which is different from the sliding window approach. However, their work is also based on learning color feature to detect vehicles.

3. The VEDAI dataset

As explained in the introduction, our motivation is to propose a dataset which would allow the development and benchmarking of (small) target detection algorithms, and more specifically of vehicle detection in aerial images.

The specifications of the dataset must satisfy the following constraints:

- the images should be free of copyright or at least freely usable within the computer vision community, which is a strong criterion as producing aerial images is usually expensive,
- the number of different targets as well as their types should be diverse enough to represent the needs of the related applications,
- backgrounds should be as diverse as possible,

Figure 4: Four illustrative images of the database: two color images (first row) and two infrared images (second row).

- targets should be small (in pixels),
- the ground-truth, included in the annotations distributed with the dataset, should be complete enough to allow the development and evaluation of target detection algorithms.

Images. After reviewing all the possible sources of images that would fit with the previous requirements, we decided to retained the satellite images

of the Utah AGRC [76], which gives access to many freely distributable aerial orthonormal images. More precisely, we selected the *HRO 2012 6 in. photography* set, with has a resolution of 4.92 in. $\times$ 4.92 in. per pixel (12.5 cm $\times$ 12.5 cm per pixel). The images were taken during spring 2012. Raw images have 4 uncompressed color channels (three visible color channels and one near infrared channel).

The original satellite images are very large, probably too large for standard detection algorithms. We therefore decided to cut them in smaller images. Another advantage in doing this is to focus the dataset on *interesting* regions. Indeed, the vast majority of the images contains repeated similar textures (*e.g.* lakes, mountains, forest) that would bias the performance evaluation. By selecting by ourselves interesting regions, we tried, as far as possible, to maximize the diversity of the dataset, in terms of vehicles as well as in terms of backgrounds and distractors.

A total of 1,210 1024 $\times$ 1024 images have been manually selected and included into 4 different sub-sets (see Table 2): (i) large-size color images (LCIs), (ii) small-size color images (SCIs), (iii) large-size infrared images (LIIs), (iv) small-size infrared images (SIIs). The exact position of the 1024x1024 image in the big image provided by the UTA database is given, in case someone wish to extend the database. Please note that the small images are exact duplicates or the large ones, except they are downscaled to (512 $\times$ 512) to make the targets smaller. The color images have three 8-bits channels image (R,G, B) while the infrared images have only one 8-bits channel. All images have been taken from the same distance to the ground. This is a big difference with other datasets such as PASCAL [8] or LabelMe [10], were distance to objects can change a lot. This choice was made because for surveillance application, the airplane is often at a known distance from the ground, and the surveillance system often have some specification of height. Furthermore, there is no oblique view for the images, unfortunately we did not find some oblique views to add to our dataset. These two limitations make the problem easier, however it does not make the database trivial, as it can be seen in the results. Splitting the dataset according to target sizes and type of imaging reflects the two main needs of surveillance applications. Thanks to the near infrared set, we can have a determination of the performance of algorithm without any color cue. The number of images is not very high, however it is representative of what an industrial firm could collect with a new sensor. It is however possible to learn on other data and test on ours.

While some datasets contain a single type of background (*e.g.* [31] con-

name	image size	resolution(cmpp)	#chanels	chanels type
large-size color images (LCIs)	1024x1024	12.5x12.5	3	colors
small-size color images (SCIs)	512x512	25.0x25.0	3	colors
large-size infrared images (LIIs)	1024x1024	12.5x12.5	1	near infrared
small-size infrared images (SIIs)	512x512	25.0x25.0	1	near infrared

Table 2: the different subsets.

tains only vehicles on roads in urban environment), our dataset has many different type of backgrounds (*e.g.* fields, grass, mountains, urban area, *etc.*).

Images with too many vehicles (for example a large parking lot, such as presented in [33]) were excluded, as an algorithm which returns random positions would provide reasonable scores on such images. In addition, we are more interested by isolated vehicles and avoiding false alarm on rich backgrounds. Figure 4 shows some illustrative images.

Vehicles. The proposed dataset contains nine different classes of vehicles, namely the 'plane', 'boat', 'camping car', 'car', 'pick-up', 'tractor', 'truck', 'van', and the 'other' category. Typical images illustrating theses classes are given Figure 6. Two meta-classes are also defined and considered in the experiments.The 'small land vehicles' class containing the 'car', 'pick-up', 'tractor', and 'van' classes, and the 'large land vehicles' class containing the truck' and the 'camping car' classes. There is an average of 5.5 vehicles per image, and they occupy about 0.7% of the total pixels of the images.

Folds. Images, among each subset, are divided in 10 folds, in order to allow a 10-folds validation protocol. The folds are made in such a way that each fold contains approximately the same number of vehicles of each class, except for the 'airplanes' class for which there were too few targets for splitting them equally. Table 3 gives some statistics about the database.

Annotations. Each target in the dataset has been annotated by one human operator in the following way. First, the coordinates of the center of the vehicle in the image is given, as well as its orientation (the angle it makes with the horizontal line, modulo π or 2π), the coordinates (in pixels) of its 4 corners and its class label. In addition, there are also two binary flags stating if the vehicle is occluded or not and if it is fully contained in the image. The orientation is set to the range $[0 - \pi]$ when the difference between front and back is not obvious, which is the case for the 'camping car' and the 'other' classes. For the 'plane' class, we annotated the plane so that the wings'

Class name	Tag	Targets per fold	Total	Orientation
Boat	boa	17	170	$[-\pi\ \pi]$
Camping Car	cam	39	390	$[0\ \pi]$
Car	car	134	1340	$[-\pi\ \pi]$
Others	oth	20	200	$[0\ \pi]$
Pickup	pic	95	950	$[-\pi\ \pi]$
Plane	pla	–	47	$[-\pi\ \pi]$
Tractor	tra	19	190	$[-\pi\ \pi]$
Truck	tru	30	300	$[-\pi\ \pi]$
Vans	van	10	100	$[-\pi\ \pi]$
Small Land Vehicles	slv	295	2950	$[-\pi\ \pi]$
Large Land Vehicles	llv	69	690	$[0\ \pi]$

Table 3: Statistics of the dataset

```
00000000  580.69  1009.22   3.012318   554  [...]  1021  002  1  0
00000001  344.82   812.36  -0.013888   326  [...]   819  001  1  0
00000001  413.21   811.24  -0.011363   392  [...]   819  009  1  0
```

Table 4: Exerpt from the annotation file. This file contains, for each target and from left to right (one target per line), the image ID, the coordinates of the center in the image, the orientation of the vehicle, the 4 coordinates of the 4 corners, the class name, a flag stating if the target is entirely contained in the image, a flag stating if the vehicle is occluded.

plane are within the annotation rectangle and that the orientation follow the fuselage. For the 'other' class, the orientation is along the length of the vehicle. One single annotation file, given with the dataset, gathers all the annotations. Some illustrative lines are shown in Tab. 4.

For convenience, the annotations are also provided as a set of independent files, one per image, each file being named according to the image name (*e.g.* annotation file of image `00000010.png` is named `00000010.txt`). They contain exactly the same information than the main annotation file. One set of annotation is provided for 512×512 images (both IR and color) and a different one for the 1024×1024 images. A tool to explore the database is also provided. This tool allows to inspect the images of a specific set and see the polygons around the vehicles, as well as their orientation. An example is given Fig. 5.

Figure 5: One illustration of the annotation provided, as given when using the tool to explore the database.

4. Performance evaluation

As explained before, our motivation for proposing this new dataset is to provide the computer vision community with tools allowing to make progress in the field of automatic target detection. However, progress can be measured if and only if the way to evaluate performance is clearly defined. This is the purpose of this section, which defines the evaluation protocols accompanying the dataset. These evaluations are done independently for each sub-dataset and each category.

Figure 6: Images illustrating the different categories of the dataset. From left to right: car, truck, camping car, tractor, plane, boat, other, pickup and van.

000001	Cam	150	12	1.13
000001	Car	415	189	-0.50
000002	Car	319	489	1.12
000002	Pla	32	411	1.01
000002	Tru	144	28	0.12

Table 5: Exerpt of a detection file. Each line represent a single detection encoded by the image number of the image containing the target, the target class, the x and y coordinates of the target center in the image and finally the detection score.

4.1. Protocol

The protocol relies on a ten-folds cross validation procedure. It is a standard way to evaluate algorithms in machine learning [77]. Cross-validation is a model validation technique to assess how the results of a statistical analysis will generalize to an independent data set. One round of cross-validation involves partitioning a sample of data into complementary subsets, performing the analysis on one subset (called the training set) and validating the analysis on the other subset (called the testing set). To reduce variability, multiple rounds of cross-validation are performed using different partitions and the validation results are averaged over the rounds. Our dataset is provided with pre-defined folds (*i.e.* the list of images in each fold is fixed), the folds being defined in such a way that the number of vehicles per fold is approximately the same, for each category.

Among the 10 folds, a single fold is retained as testing data to evaluate the model, while the remaining 9 folds are used as training data. The cross-validation process is then repeated 10 times, with each of the subsamples used exactly once as test data. The 10 results from the different folds are then averaged to produce a single estimation and the standard deviation over the 10 folds is reported as well.

The advantage of this method over repeated random sub-sampling is that all observations are used for both training and validation and each observation is used for testing exactly once. Please note that despite train/validation splits are provided, any split can be used for the cross validation. The performance on the test fold is evaluated through the measures of performance defined in the next section.

4.2. Metrics

The detection algorithm to be evaluated has to process the set of images contained in one of the test fold and output a set of predictions, which are the predicted locations of the target in the test images. Detections are 5-tuples vectors containing respectively the id of the images in which the target was detected, the class of the target, its coordinates in the image reference and a confidence score. There are no restriction regarding the range of values of the detection scores, but a higher score is interpreted as more confidence in the detection.

Ideally – and as we are interested by different operating points (*e.g.* different points of the ROC curve) – the operating point should be a parameter of the detection algorithm, which should be run for the different operating points. It is especially true for algorithms relying on cascades, as they cannot score their detections for the whole range of score (low scores are rejected by the first levels of the cascade). However, for simplicity and because it covers most of the cases, we assume that the detection algorithm can provide scores and the different operating points are obtained by thresholding the detection scores accordingly.

In practice, algorithms have to output a text file with one line per predicted detection, as illustrated Table 5. Each line contains the image id, the class, the position and the score of the detection.

As it is commonly done in such situation, we aim at evaluating the performance for different operating points, which is done, as said before, by varying a detection threshold. More precisely, the first metric is the Precision–Recall Curves, chosen because (i) contrarily to the ROC curve it is not dependent

on the number of windows processed by images and (ii) because of its wide use in the computer vision literature. The second one is the average recall for a given number of False Positive Per Image, which is closer to what people who develop applications need to know. Both are defined in the following paragraphs.

For a given image I and a particular value of the threshold t, only the detections in I having a score greater than that t are supposed to be detections, the other are discarded. The number of detections in I that match with the ground truth are denoted as True Positive or $TP(I,t)$ while the detections corresponding to non-target regions are denoted as False Positive or $FP(I,t)$, for the given threshold t. In the same way, targets that are not detected (missed) are denoted as False Negative or $FN(I,t)$.

It allows us to define the precision and the recall of a given testing fold, for a given operating point t, as:

$$precision(t) = \frac{\sum_{I \in \text{fold}} TP(I,t)}{\sum_{I \in \text{fold}} TP(I,t) + \sum_{I \in \text{fold}} FP(I,t)} \tag{1}$$

$$recall(t) = \frac{\sum_{I \in \text{fold}} TP(I,t)}{N_T} \tag{2}$$

where N_T denote the number of targets in the considered fold.

We also define the *False Positive Per Image rate* (FPPI) as:

$$FPPI(t) = \frac{\sum_{I \in \text{fold}} FP}{N_{test}} \tag{3}$$

where N_{test} is the number of images in the considered fold.

The whole precision-recall curve, obtained by considering all the possible values for t is very informative, but there is often a desire to summarize it into a single number. The traditional way of doing this is the *11-point interpolated average precision*. The interpolated precision is measured at the 11 recall levels of 0.0, 0.1, 0.2, ... , 1.0. Then the arithmetic mean of these interpolated precisions is computed. This method has been used in TREC 8 [44]. For real use purposes, it is required that a specific t value is chosen. It can be chosen according to the desired operating point of the Precision–Recall curve.

In addition to the average precision, we also compute the recall for the 4 different FPPI rates of 10^{-2}, 10^{-1}, 1 and 10 false positives per images.

This gives a total of 5 measurements: the mean average precision and the FPPI for 4 recall rates. These 5 measurements are computed for each ones of the 10 test folds, and the mean values and corresponding standard deviations are reported as well.

We now have to define what a True Positive is, *i.e.* when a predicted target fits with one of the target of the ground truth. In principle, it would be necessary to find the best set of matches between predictions and ground truth. In our case the targets do not overlap and, as objects have a single scale, it can be done much more simply by assigning each prediction to the nearest target in the ground truth.

By denoting as $p = (x, y)$ the coordinates (in pixels) of the predicted target and as $P = (X, Y)$ the coordinates of the closest target in the ground truth, the prediction is considered as being correct (*i.e.* as a TP) if the prediction is within an ellipse centered on the ground truth, according to the following criterion:

$$(p-P)^t \begin{pmatrix} \cos a & -\sin a \\ \sin a & \cos a \end{pmatrix}^t \begin{pmatrix} \frac{1}{W^2} & 0 \\ 0 & \frac{1}{H^2} \end{pmatrix} \begin{pmatrix} \cos a & -\sin a \\ \sin a & \cos a \end{pmatrix} (p-P) \leq 1 \quad (4)$$

where W and H are the half height and half width of the target in pixels and a is the (ground truth) orientation of the target. By construction, the ellipses are touching the edge of the targets (except for the plane category, but we remind that it is not used for the detection as it has too few images). Consequently, there is no overlap between the ellipse of two targets and therefore no need for considering more complex assignation procedures. If several detections are associated to the same ground truth target, the one having the best score is counted as a TP while other ones are discarded because they are not False Positive nor True Positive either. However, multiple False Alarms are all counted as False Alarms.

5. Experiments with baseline algorithms

We have tested several recent algorithms on our dataset, providing a baseline to be compared with. These experiments are both motivated by (i) characterizing the dataset by showing how state-of-the-art algorithms behave on the dataset, and by (ii) giving some baseline results allowing to position new approaches. We do know that they are not the best possible algorithms, however they can be considered as strong baselines to be compare with, and, in addition, they show how challenging this dataset is.

We concentrated our effort on sliding windows approaches, which are the most efficient approaches for this type of task, at the moment. More specifically, we have implemented a standard SVM-based sliding window pipeline, with 3 different features, namely HOG [4], LBP [78], and LTP [79]. We have also experimented the code of [7] (Deformable Part Model or DPM in short), which, in addition, decompose objects into different roots and parts. For completeness, we have also implemented a template matching approach [80], as template matching was considered in the past as the standard approach for small target detection, and we tested a non sliding window protocol based on the Hough forest [81].

5.1. Experimental settings of the different detectors

5.1.1. The sliding window pipeline

Visual features. Rectangular regions to be classified are represented by one of three features or by their combination (concatenation of the representations). The following paragraphs describe more precisely how these features are implemented (namely Histogram of Oriented Gradient, Local Binary or Ternary Pattern).

Our implementation of the Histograms of Gradients 31 (HOG31) is based on [4] and includes the improvements suggested by [7], supposed to give better results. [7] claims that augmenting the standard HOG feature with contrast sensitive and contrast insensitive features improves performance. It gives a 31-dimensions descriptor, the 27 first components are sensible to the orientation of the gradient, while the 4 last components are sensible to its intensity. We will refer to this descriptor as HOG31 in the different tables. The norm of the gradient used here is the maximum norm over the different channels in case of color image. With these features, the rectangular region to be represented is divided according to a grid of 16×16 blocs with an overlap of 8 pixels. Histograms of oriented gradient are computed for each block with 9-orientation bins when the orientation is unsigned and 18 otherwise.

Local Binary Patterns (LBP) [78] are made of local differences of the image. More precisely, each pixel p of the image is given a binary code in the following way : $code = \sum_{i \in V} 2^i (I(p) < I(p_i))$ where the p_i are the pixels in the neighborhood (V) of the pixel p to be encoded. We take a 3×3 neighborhood resulting in a set of 256 possible different codes. We use only *uniform* LBP, which are those containing at most two bitwise transitions from 0 to 1 or vice versa when the bit pattern is traversed circularly (consequently there

are 59 different uniform LBP). Histograms of codes are computed within 16x16 cells, before being L1-normalized.

Local Ternary Pattern (LTP) [79] is an extension of LBP, aiming at dealing with problems on near constant image areas. With this encoding the difference between the center pixel and a neighboring pixel is represented by three different values (1, 0 or -1) according to a threshold ϵ. The ternary pattern is the combination of two binary patterns taking into account its positive and negative components: $code = \sum_{i \in V} 2^i.(\epsilon < p_i - p) + \sum_{i \in V} 2^{i+size(V)}.(\epsilon < p - p_i)$. In our experiments, we set $\epsilon = 5$, which is a standard value. As for LBP, noisy codes are removed by taking only uniform LTP. Histograms are computed within 16×16 blocks, normalized by L1-normalization. For LBP and LTP, color images have just been used as grayscale images, averaging the three different channels.

Support Vector Machine (SVM) classifier. In our experiments we used the svmlight [82] library. For efficiency, we use a linear kernel in all of our experiments. An initial classifier is trained with the targets of the training set as positive training data and randomly chosen background regions (not overlapping with targets) as negative data. The performance of the classifier can be improved in a second time by adding *hard negative examples*, as in [4]. They are the false positive regions with highest scores. We repeat this process until we pass through the whole database.

Step size. One key parameter of the algorithm is the step-size, which is amount of shift (in pixel) between two positions of the sliding window. In practice, the step size has been set to 8 pixels – which appeared to be a good tradeoff between efficiency and performance for the 1024x1024 images – and of 4 pixels for the 512x512 images. As the size of the vehicles is supposed to be approximately known, only 4 scales are explored in addition to the original one, all above the original scale, as the models are build from the smallest size of positive example. The scaling factor is of $2^{\frac{1}{10}}$, which is a typical value in such situations (*e.g.* [7]). In practice, the image is downscaled while the size of the sliding window is the same.

Multi-roots classifier. As we use linear classifiers, representing object's appearance with a single model is usually not enough. In practice we used the multi-root SVM approach of [7] and model the targets with 12 different roots. We cluster the target orientations into 6 different groups and learn one detector for each group. Each view has an average height and width,

obtained by averaging training bounding box sizes. It fixes the aspect ratio of each view. It is important to note that the rectangular regions used for training the classifiers are not those given in the ground truth: we instead use the closest regions of the grid (using the step-size given above) so that the training examples capture the shifts of the targets within the bounding boxes due to the step-size. In addition, we flip all the images according to vehicle orientation and learn another set of 6 detectors. It gives a total of 12 detectors. Each of them is applied on the image independently.

Non-maximal suppression. It has been observed that sliding window detectors usually output multiple detections for one single target. Filtering out these extra detection is often called in the literature *Non Maximum Suppression* (NMS) [3]. In our experiment, we did this by keeping only the detections having the best score in case two detections overlap, as done by [3]. This merge the results from the different detectors.

5.1.2. Template matching

In addition, we also implemented a standard template matching approach. The templates are obtained by clustering training targets according to their orientation. We then represented each cluster by 2 templates obtained by the k-medoid algorithm ([83]). We then used the normalized sum of square differences to locate template occurrences:

$$s(f) = 1/min_{template}(\frac{\sum (f(i) - template(i))^2}{\sqrt{||f|| * ||template||}})$$

5.1.3. The Deformable Part Model

Our experiments with the Deformable Part Model (DPM) detector [7] use the latest implementation publicly available.[2] Only one parameter has to be set, the number of roots used to learn the model. However, to adapt our database, some modifications had to be done. To make the DPM working with so small objects, we have had to upscale the images, and remove the lines of code that discarded small positive examples. Following the work of [84], the learning of the parts was deactivated, as it does not seem relevant to learn parts on so small targets. When compared to our own detection pipeline, there are several differences. First, it uses a mirror approach in

[2]Version 5, available at http://cs.brown.edu/∼pff/

such a way that each root is divided in two roots with a left right distinction. Second, it has a latent learning, that match the examples that look alike together, instead of only relying on the orientations and height/width ratio.

5.1.4. Random Hough Forest

We also tested a Random Hough Forest algorithm of [81], which is not based on a sliding window protocol. [3] We used 15 trees for all sets. For the VeDAI-Small sets we used 4x4 patchs (which were the best in preliminary experiment), and for the VeDAI-Large sets, we used 8x8 patches.

5.2. Results

Four different types of results are presented in this section. First, the performance of the baseline detectors introduced in section 5 are compared on the small-size infrared images (SIIs sub-set). In a second time, the best of the baseline detectors is evaluated on the 3 other sub-sets, and the influence of the type of images or resolution is commented. Third, a brief parametric study is proposed, in which we experiment on the key parameters of the different detection algorithms. Finally, we show how the parameters of the evaluation process influence the evaluation of the performance.

5.2.1. Results on the SIIs sub-set

This sub-set is the most difficult as only one bandwidth is used and as the images are the smallest. The quantitative results are shown in Table 6.

We run 7 different detectors on this sub-set. The 4 first are based on the standard sliding window pipeline previously described with different features: SVM+HOG31, SVM+LBP, SVM+LTP and SVM+HOG+LBP (concatenation of HOG and LBP descriptors). The three last ones are the DPM (using only the root and no parts), the Hough Forest (HF) and the template matching (TM) approaches.

Best results are obtained with the combination of HOG and LBP features and a SVM classifier, but it must be noticed that all the SVM-based methods give approximatively the same performance. The template matching algorithm gives very bad results. We experimented also an alternative template matching approach using HOG instead of raw pixel intensities as input. It did improve the results but not sufficiently to surpass the SVM+feature

[3] We used the code provided at `http://www.iai.uni-bonn.de/~gall/projects/houghforest/houghforest.html`

pipeline. The performance of the DPM is comparable to the performance of the HOG based detector. Finally, the Hough Forest method give some interesting results, but is worse than all the SVM + feature methods.

But the most interesting lesson we can get from this table is the bad overall performance obtained by all of these methods. Indeed, even if the mAP is quite high, the recall at 0.01 FPPI – which is a realistic operating point for the targeted applications – is between 10% and 20%, which means that the vast majority of the targets are not detected. One can also note that for low FPPI, the standard deviation becomes larger as the number of detected target becomes too small to have a reliable estimation of the variance. These bad results are consistent with previous works on pedestrian detection, in low resolution images which report a miss rate of more than 60% [85, 86] at 0.1 FFPI for images of 640x480 pixels.

Classes with fewer examples ('tractors', 'boats' and 'other') are hard to learn, as we could expect. The 'van' category however have few training examples but is not so badly detected, probably because its appearance is quite distinct from other classes.

5.2.2. Performance on large-size images and color images

We took the detector performing best on the small-size images sub-set (*i.e.* the SVM+HOG+LBP detector), and tested it on the three other sub-sets, namely LIIs, SCIs, LCIs. These results can be seen in Table 7. Results obtained with color images are usually better, even if the HOG feature does not strongly exploit color information. Surprisingly, the performance of the 'pick up' and 'tractor' classes are not as good as on infra-red images. This could be due to the fact that pick-ups and tractors often carry objects, which could give much more irrelevant color gradient in the visible bandwidth. In conclusion, the choice of the imaging technology (infrared or color images) depends on the targeted applications.

Resolution is also a factor of improvement, even if we expected a more significant gap. For some classes the gain in performance is significant (*e.g.* 'oth', 'cam', 'boa'), but surprisingly the performance of the pick up' and 'tractor' classes decreases. This can be due to the loads at the rear of these vehicles, which is blurred at low resolution, making the appearance more stable.

mAP										
detector	slv	llv	Boa	Cam	Car	Oth	Pic	Tra	Tru	Van
DPM	**72.4±2.6**	**40.7±4.6**	**26.1±19.2**	**41.9±11.7**	**60.5±4.2**	**6.0±4.5**	**52.3±5.3**	**33.8±15.6**	**34.3±5.9**	**36.3±10.2**
SVM +HOG31	**71.5±2.5**	**36.0±4.7**	**32.2±14.9**	**33.4±8.9**	55.4±2.6	**6.9±4.2**	**48.6±4.9**	07.4±03.6	**32.5±8.0**	**40.6±14.8**
SVM +LBP	64.3±3.2	22.9±5.4	07.6±05.4	24.1±9.5	51.7±5.2	1.0±0.9	48.2±4.5	06.3±02.8	**25.7±6.3**	**38.1±14.0**
SVM + LTP	**73.1±3.1**	**40.9±5.5**	17.9±8.0	**45.6±7.9**	**60.4±4.0**	**6.5±3.1**	**56.9±5.5**	**21.2±08.6**	**35.7±9.5**	**51.6±17.1**
SVM +HOG31+LBP	**75.0±2.2**	**42.2±5.4**	**34.1±16.1**	**47.5±11.2**	**61.3±3.9**	2.0±1.3	**57.5±4.5**	**17.5±08.2**	**37.4±8.4**	**44.7±14.4**
TM	2.2±1.7	1.0±1.2	0.03±0.03	3.2±2.4	5.3±1.5	0.1±0.02	0.6±1.0	0.2±0.2	0.4±0.6	1.7±3.5
HF	39.7±4.0	28.2±6.9	10.1±3.1	32.5±8.7	39.8±4.0	5.2±0.02	30.7±5.5	15.3±4.7	17.7±4.5	40.5±12.6
recall at 1 FPPI										
detector	slv	llv	Boa	Cam	Car	Oth	Pic	Tra	Tru	Van
DPM	**71.8±2.6**	**62.9±5.5**	**48.0±32**	**72.9±8.6**	**74.5±4.5**	**22.5±12**	**70.6±5.0**	**61.4±19**	**60.1±4.7**	**85.9±7.5**
SVM +HOG31	**71.4±3.0**	**56.8±5.8**	**64.8±15**	**64.9±8.8**	**70.5±3.9**	**32.9±3.5**	**69.5±4.7**	34.9±14	**51.1±9.2**	**80.5±11**
SVM +LBP	62.9±2.9	38.0±4.8	29.4±14	42.0±8.5	65.9±4.7	8.6±5.1	**65.2±6.0**	29.5±8.3	**48.8±9.4**	**76.0±16**
SVM +LTP	**72.0±3.5**	**59.2±4.8**	**51.6±14**	**67.6±6.6**	**77.2±4.8**	**26.5±11**	**72.8±5.0**	**60.2±14**	**59.1±11**	**85.2±10**
SVM +HOG31+LBP	**74.7±2.9**	**59.7±5.8**	**56.9±16**	**72.4±7.6**	**77.6±3.8**	**16.2±8.9**	**75.1±4.4**	**53.1±14**	**60.6±10**	**86.8±9.6**
TM	3.8±2.4	3.5±3.6	0.6±1.9	12.3±3.7	10.5±2	0.0±0.0	1.2±1.7	1.9±4.2	1.8±3.9	9.5±12
HF	37.4±5.0	41.8±9.8	38.7±15	**59.3±6.7**	47.7±6.8	**33.9±8.8**	46.8±6.4	**51.6±10.8**	40.9±7.4	**75.2±12**
recall at 0.1 FPPI										
detector	slv	llv	Boa	Cam	Car	Oth	Pic	Tra	Tru	Van
DPM	**46.1±4.4**	**28.5±4.3**	**23.4±19**	**37.4±11**	**31.4±5.8**	**10.7±8.2**	**30.9±6**	**39.3±18**	**33.7±3.7**	**54.2±15**
SVM + HOG31	**41.9±5.7**	**24.0**	**39.0±14**	**30.1±11**	**24.4±5.7**	**9.3±5.9**	**27.7±5.4**	13.3±3.5	**33.5±9.8**	**48.9±17**
SVM + LBP	37.7±5.8	**17.9±6**	8.9±7.1	**24.6±9.5**	**26.3±7.9**	**1.9±3.1**	**29.3±5.5**	10.8±5.7	**26.9±6.7**	**47.1±18**
SVM + LTP	**43.7±6.5**	**29.0±5.4**	**22.4±7.2**	**42.6±7.4**	**29.1±6.2**	**8.3±6.0**	**37.1±5.1**	**25.4±12**	**35.3±9.4**	**64.3±16**
SVM + HOG31+LBP	**49.1±5.3**	**30.1±6.9**	**40.2±16**	**45.1±12**	**30.7±7.0**	**3.0±4.3**	**36.0±5.9**	**20.1±6.5**	**36.4±7.7**	**53.3±14**
HF	9.8±1.7	**21.1±5.7**	13.3±5.8	**28.2±11**	18.4±2.5	**3.2±5.2**	12.2±5.1	**16.7±7.1**	15.8±3.9	**48.4±16**
recall at 0.01 FPPI										
detector	slv	llv	Boa	Cam	Car	Oth	Pic	Tra	Tru	Van
DPM	**23.4±4.9**	**6.8±3.0**	**15.3±13**	**10.4±12**	**13.4±6.8**	**1.9±3.2**	**12.5±4.2**	**18.9±9.9**	**18.5±7.8**	**17.0±16**
SVM+HOG31	**17.5±6.5**	**9.5±4.3**	**19.7±15**	**7.1±5.7**	**7.8±5.5**	**2.8±3.9**	**6.9±4.4**	1.9±3.1	**17.4±7.0**	**29.0±16**
SVM +LBP	**16.2±5.6**	**6.1±13.5**	3.6±5	**11.8±6.6**	**5.5±2.2**	0.0±0.0	**10.9±2.2**	1.2±2.4	**12.5±8.0**	**29.5±12**
SVM +LTP	**16.5±6.9**	**13.9±4.9**	**8.9±7.3**	**20.0±7.6**	**9.3±3.7**	**2.6±3.6**	**11.8±7.3**	**7.8±7.3**	**17.6±10**	**35.5±18**
SVM + HOG31+LBP	**19.1±9.6**	**13.5±4.2**	**22.8±11**	**20.6±13**	**8.3±5.2**	0.0±0.0	**15.1±8.4**	**9.6±7.0**	**20.2±9.2**	**32.6±15**
HF	4.2±1.3	**8.7±4.1**	2.5±3.3	**10.0±4.7**	**9.1±3.9**	**0.5±1.6**	2.7±1.6	4.0±4.4	5.7±3.6	**29.0±13**

Table 6: Performance of the different detectors on the SIIs sub-set. The mean performance as well as the standard deviation are given for (i) mAP (ii) recall at 1, 0.1 and 0.001 FPPI. DPM stands for Deformable Parts Model, TM for Template Matching and HF for Hough Forest. the underline score is the best one, the scores in bold are within the margin of error.

mAP										
sub-set	slv	llv	Boa	Cam	Car	Oth	Pic	Tra	Tru	Van
SIIs	75.0±2.2	42.2±4.4	34.0±16.1	47.5±11.2	61.3±3.9	2.0±1.3	57.5±4.5	17.6±8.2	37.4±8.4	44.7 ±14.4
SCIs	74.9±2.5	44.1±6.7	38.1±15.9	58.8±9.7	63.6±5.3	2.9±2.3	55.2±5.3	15.6±9.2	36.1±7.2	50.3±17.5
LIIs	77.0±1.6	45.7±3.9	36.9 ±16.2	52.9±9.3	62.6±4.5	4.4±1.8	54.4±5.2	10.9±6.2	34.3±6.8	48.1±19.9
LCIs	76.8±1.5	45.6±4.2	40.4±15.7	55.4±8.1	63.8±4.1	5.6±3.7	51.5±4.8	13.9±9.5	38.1±9.6	49.4±17.9
recall at 1 FPPI										
sub-set	slv	llv	Boa	Cam	Car	Oth	Pic	Tra	Tru	Van
SIIs	74.7±2.9	59.7±5.8	56.9±16	67.6±6.6	77.6±3.8	16.2±8.9	75.1±4.4	53.1±14	60.6±10	86.8 ±9.6
SCIs	74.6±2.8	62.8±6.4	64.4±15	80.2±7.1	79.8±4.3	15.9±7.9	73.8±4.5	48.2±15	63.8±8.8	84.6±15
LIIs	76.8±2.3	65.1±4.6	61.0±12	76.1±6.0	78.3±4.1	24.3±7.8	75.7±5.5	43.8±15	58.7±9.4	83.4±10
LCIs	76.5±2.3	66.2±4.5	68.5±13	80.6±6.5	78.9±3.5	29.6±8.3	73.1±3.5	42.4±14	61.0±12	83.3±14
recall at 0.1 FPPI										
sub-set	slv	llv	Boa	Cam	Car	Oth	Pic	Tra	Tru	Van
SIIs	49.1±5.3	30.1±6.9	40.2±16	45.1±12	30.7±7.0	3.0±4.3	36.0±5.9	20.1±6.2	36.3±7.7	53.3±14
SCIs	47.7±7.2	32.8±7.0	42.9±13	55.4±9.8	32.7±10	3.9±4.4	33.7±6.4	16.6±13	36.1±8.0	64.2±18
LIIs	51.6±5.9	33.8±5.3	39.9±17	47.4±12	31.5±6.6	7.1±4.2	31.0±5.3	14.5±8.2	35.9±5.8	56.3±20
LCIs	51.9±3.3	32.6±5.6	42.7±16	48.7±7.7	35.5±7.9	5.5±5.2	29.3±6.2	19.9±9.9	37.5±10	64.1±20
recall at 0.01 FPPI										
sub-set	slv	llv	Boa	Cam	Car	Oth	Pic	Tra	Tru	Van
SIIs	19.1±9.6	13.5±4.2	22.8±11	20.7±13	8.3±5.2	0.0±0.0	15.1±8.4	7.8±7.3	20.2±9.2	32.7 ±15
SCIs	19.7±8.5	13.7±8.9	26.1±13	29.2±14	8.9±7.0	1.0±2.2	11.8±8.3	6.6±5.6	16.5±7.6	36.6±18.4
LIIs	19.3±8.1	15.0±7.7	30.2±16	20.6±13	11.1±7.0	1.1±2.3	7.7±4.5	2.8±4.7	17.0±5.7	39.7±21
LCIs	24.1±8.6	12.3±5.7	28.5±15	24.6±9.5	10.0±6.4	2.1 ±3.6	9.2±4.1	6.1±7.5	20.0±9.1	34.3±16.6

Table 7: Performance of the SVM + HOGLBP detector given as mean average precision and recall for 10, 1, 0.1 and 0.01 FPPI. The table gives the mean over the 10 folds as well as the standard deviation.

5.2.3. Parametric study

Number of root models. The number of roots of the model is critical for obtaining good performance. It is in practice the most sensible parameter. Some preliminary experiments have shown that at least four roots are necessary to get good performances (see Figure 7). Targets aspect ratios are very different due to the diversity of possible orientations and vehicle appearances, explaining why different roots (with different aspect ratios) are needed. Regarding the DPM, using parts in addition to the root can improve the performance a little bit, but the code then become very unstable (we believe this is because when roots are too small, part sizes become lower than a critical threshold) and do not give any results. However, we have been able to see that it improves the performance by 3% of mAP on the folds when the code succeeded. This result is consistent with the previous work of Divvala [87], which states that the number of roots is more important than the latent learning or the deformable parts.

Parameters of the sliding windows. The step-size in the sliding windows algorithm can impact the performance. Figure 8 shows that if the step-size is too high (higher than twice the step size of the learning stage), the perfor-

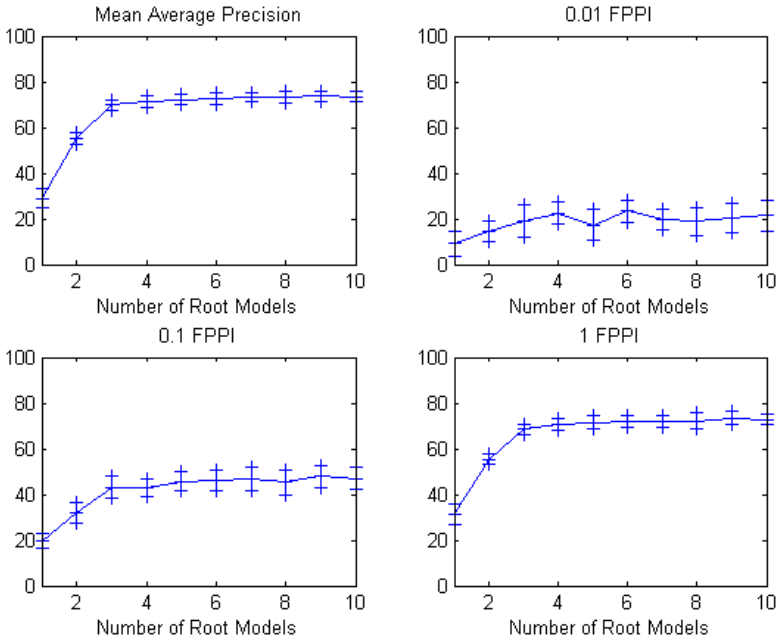

Figure 7: Performance (mAP) as a function of the number of roots in the models using the DPM and for the 'slv' category. Error bars represent the standard deviation.

mance significantly drops (except for the recall at 0.01 FPPI which is already so bad that it is not affected). The presented results are those given by the SVM+ HOG detector on the 'slv' category, but similar behavior has been observed for other classes and other detectors.

The number of scales at which the image is processed also affect the performances but not significantly, as shown by Figure 9. This was expected as the distance between the camera and the target is constant and is supposed to be known.

We also have tested the influence of the threshold used for the non-maximum suppression. Remind than when two detections are too close (in terms of overlap measured by the Jaccard index between them), only the best one (*i.e.* the one with the highest score) is kept. Fig. 10 shows that choosing a very small Jaccard index (such as 0.1) lowers the results a little bit. However, as the targets are in general far from each-other, detections are rarely affected by non maximal suppression. If the threshold is too high, the non-maximal suppression stage does not eliminate enough multiple de-

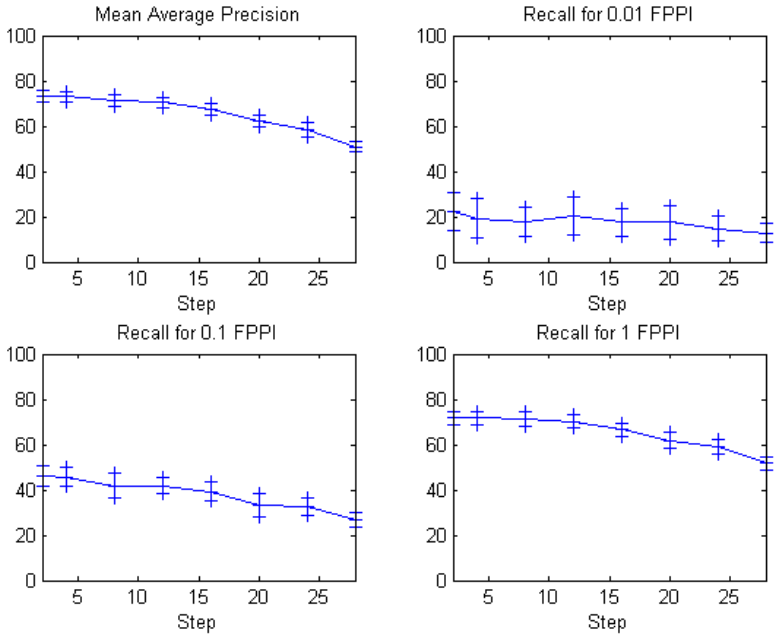

Figure 8: Influence of the sliding window step-size with the SVM+HOG detector on the 'slv' class. Error bars represent standard deviation.

tections and the performance drops by 10%.

5.2.4. Further analysis

Single class analysis. We investigate here a slightly different scenario in which we artificially remove all the targets which does not belong to the class of the detector to be evaluated. This is done, in practice, by only counting as false positives the predictions made on background regions (detection caused by vehicles from other classes are simply discarded and not counted as true positive neither).

Table 8 shows that, in this case, the performance only improves a little bit. The only significant gain is for cars and pick up, that are confused one from another. It demonstrates that most of the false positives do come from the background and not from the vehicles of other classes. This is due to the fact that rich background areas occupy the greatest proportion of the images.

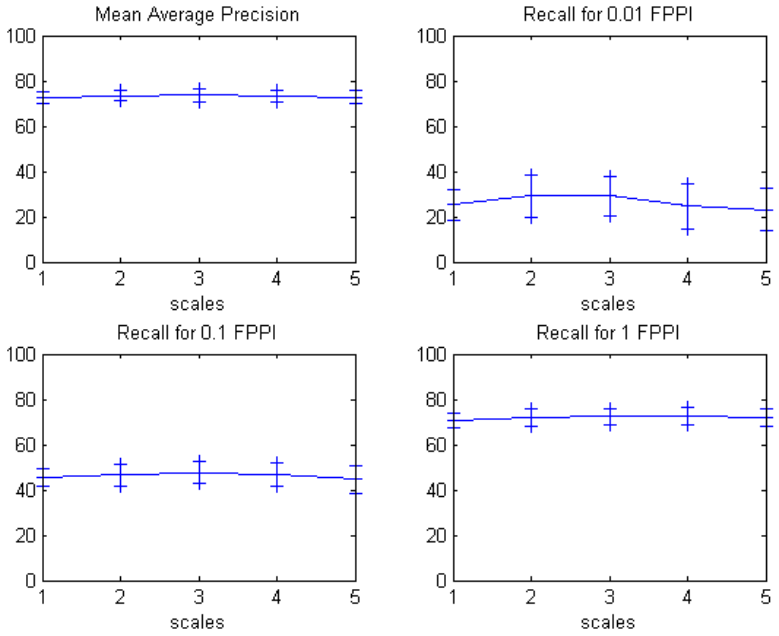

Figure 9: Influence of the number of scales used during the detection phase with the SVM+HOG detector on the slv class. Error bars represent standard deviations.

	slv	llv	boa	cam	car
with other vehicles	49.1±5.3	30.1±6.9	40.2±16	45.1±12	30.7±7
without other vehicles	54.2±4.8	33.1±6.5	45.5±15.7	48.6±11.8	65.7±6.5
gain	+5.1	+3.0	+5.3	+3.5	+30.5
	Oth	Pic	Tra	Tru	Van
with other vehicles	3.0±4.3	36.0±5.9	20.1±6.5	36.4±7.7	53.3±14
without other vehicles	4.0±4.8	61.0±5.0	24.5±9.1	42.4±9.1	62.4±13.7
gain	+1.0	+25.0	+4.4	+6.0	+9.1

Table 8: Amount of false positive due to background false alarms: in these experiments, false positives due to vehicles of other classes are not counted as false positives. This experiment was performed on the SIIs sub-set using a SVM+HOGLBP detector. We report here the mean recall for 0.1 FPPI as well as the standard deviation.

Detection accuracy. The following experiments were designed with the motivation of evaluating the accuracy of the detection. Remind, that a detection is counted as a true positive if the center of the detection falls in an ellipse specified by the size (in pixels) of the vehicle. A threshold of 1.0 corresponds

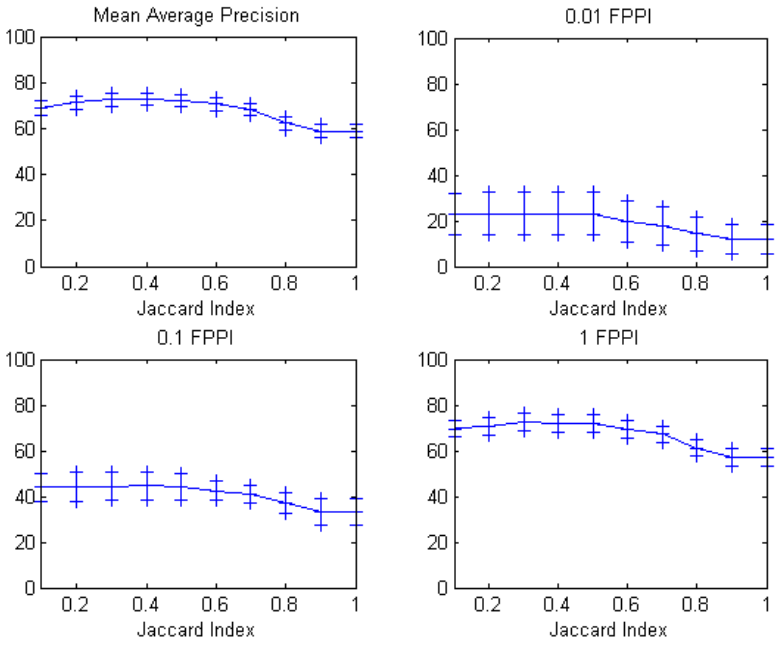

Figure 10: Influence of the Non Maximum Suppression threshold in the sliding windows approach, using the SVM+HOG31 detector on the 'slv' subset . Error bars represent standard deviations.

to the ellipse that fits with the edge of the vehicle. We can see Figure 11 how the performance is affected by setting differently this threshold. The performance is relatively stable meaning that (i) detections are relatively accurate (ii) the value chosen for the threshold in the definition of the metric is adequate and its choice is not critical.

Invariance to small shifts. In order to measure the invariance of the benchmark to small target shifts, we produced six variants of the dataset images obtained by flipping the original images horizontally, vertically and both. We apply the same transformations on transposed images. The motivation for doing this is that the target will be shifted with respect to the positions of the sliding windows (please remind that there is step-size is of 8 pixels in the sliding window) and its orientation will change as well. We ran the SVM+HOG31 on the 'slv' category and measure the standard deviation of the performance over these 7 sub-sets (the original one plus the 6 artificially generated). As shown in Table 9, the standard deviation is smaller than the

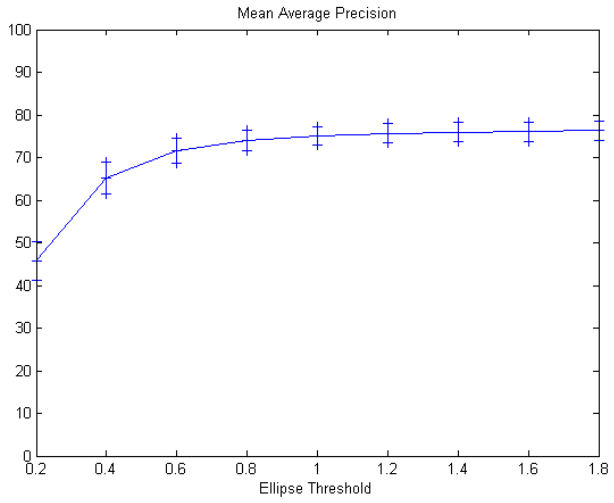

Figure 11: Mean average precision on the 'slv' category with a SVM+HOG detector, as a function of the threshold defining spatially the limit between true and false positives. Error bars represent standard deviations.

difference between folds, which shows the relative invariance of the detectors to small target shifts.

slv	mAP	0.01 FPPI	0.1 FPPI	1 FPPI	10 FPPI
Mean	72.9	20.8	45.0	72.5	90.5
Standard Deviation	0.25	1.6	0.80	0.44	0.35

Table 9: Mean and standard deviation of SVM+HOG31 detector over the 10 folds, when using different flips and transpositions of the original images.

6. Conclusions

This paper presents VEDAI (Vehicle Detection in Aerial Imagery), a new database for evaluating the detection of small vehicles in aerial images. The dataset includes different vehicle categories for a total of more than 3700 annotated targets in more than 1200 images. The images are split in four different categories corresponding to two different sizes of the images (1024×1024 and 512×512) color or infrared. The backgrounds as well as the targets are diverse. The dataset is released with precisely defined image splits and

metrics, allowing reliable evaluation. The dataset as well as the annotations are publicly available.

In addition to the dataset, we have experimented and evaluated different state-of-the-art detectors. The main conclusions of these experiments in that none of the detectors gives usable results on this dataset. For example, the best result on the 'car' class, is a recall of 13.4± 6.8 at 0.01FPPI, which is clearly below the need of targeted applications. We hope that this dataset will allow the development of new – and more efficient – target detection algorithms.

Acknowledgements. This work was supported by ANRT through the CIFRE sponsorship No 2011/0850 and by SAGEM-SAFRAN group.

References

[1] T. Wong, Atr applications in military missions, in: IEEE Symposium on Computational Intelligence in Security and Defense Applications, 2007, pp. 30–32.

[2] B. Bhanu, Automatic target recognition: State of the art survey, IEEE Transactions on Aerospace and Electronic Systems 22 (1986) 364–379.

[3] P. Viola, M. J. Jones, Robust real time face detection (2004) 137–154.

[4] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2005, pp. 886–893.

[5] G. Csurka, C. Dance, L. Fan, J. Willamowski, C. Bray, Visual categorization with bags of keypoints, in: European Conference on Computer Vision, 2004, p. 22.

[6] H. Harzallah, F. Jurie, C. Schmid, Combining efficient object localization and image classification, in: International Conference on Computer Vision, 2009, pp. 237–244.

[7] P. Felzenszwalb, R. Girshick, D. Mcallester, D. Ramanan, Object detection with discriminatively trained part based models, in: IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 32, 2009, pp. 1627–1645.

[8] M. Everingham, L. V. Gool, Williams, C. K. I., J. Winn, A. Zisserman, The pascal voc challenge, International Journal of Computer Vision 88 (2010) 303–338.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255.

[10] B. C. Russell, A. Torralba, K. P. Murphy, W. T. Freeman, Labelme: a database and web-based tool for image annotation, International Journal of Computer Vision 77 (2008) 157–173.

[11] Q. V. Le, Building high-level features using large scale unsupervised learning, in: ICASSP, IEEE, 2013, pp. 8595–8598.

[12] S. Munder, An experiment study on pedestrian classification, Vol. 28, 2006.

[13] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (1998) 2278–2324.

[14] A. Torralba, R. Fergus, W. T. Freeman, 80 million tiny images: A large data set for nonparametric object and scene recognition, in: IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 30, 2008, pp. 1958–1970.

[15] P. J. Phillips, H. Wechsler, J. Huang, P. J. Rauss, The feret database and evaluation procedure for face recognition algorithms, Image and Vision Computing 16 (1998) 295–306.

[16] F. S. Samaria, A. C. Harter, Parameterisation of a stochastic model for human face identification, in: IEEE workshop on Applications of Computer Vision, 1994, pp. 138–142.

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

[18] P. Dollár, C. Wojek, B. Schiele, P. Perona, Pedestrian detection: A benchmark, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 304–311.

[19] H. A. Rowley, S. Baluja, T. Kanade, Neural network-based face detection, in: IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 20, 1998, pp. 23–38.

[20] K.-K. Sung, T. Poggio, Example-based learning for view-based human face detection, in: IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 20, 1998, pp. 39–51.

[21] E. Hjelmås, B. K. Low, Face detection: A survey, Computer vision and image understanding 83 (2001) 236–274.

[22] A. Loui, C. Judice, S. S. Liu, An image database for benchmarking of automatic face detection and recognition algorithms, ICIP (1998) 146–150.

[23] V. Jain, E. Learned-Miller, Fddb: A benchmark for face detection in unconstrained settings, Tech. Rep. UM-CS-2010-009, University of Massachusetts, Amherst (2010).

[24] T. L. Berg, A. C. Berg, J. Edwards, D. A. Forsyth, Who's in the picture, Advances in neural information processing systems 17 (2005) 137–144.

[25] V. Ferrari, T. Tuytelaars, L. Van Gool, Object detection by contour segment networks, in: European Conference on Computer Vision, 2006, pp. 14–28.

[26] F. Tanner, B. Colder, C. Pullen, D. Heagy, M. Eppolito, V. Carlan, C. Oertel, P. Sallee, Overhead imagery research data set: an annotated data library and tools to aid in the development of computer vision algorithms, in: Proceedings of IEEE Applied Imagery Pattern Recognition Workshop, 2009, pp. 1–8.

[27] A. Torralba, A. A. Efros, Unbiased look at dataset bias, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2011, pp. 1521–1528.

[28] J. Ponce, T. L. Berg, M. Everingham, D. A. Forsyth, M. Hebert, S. Lazebnik, M. Marszalek, C. Schmid, B. C. Russell, A. Torralba, et al., Dataset issues in object recognition, in: Toward category-level object recognition, Springer, 2006, pp. 29–48.

[29] P. Carbonetto, G. Dorkó, C. Schmid, H. Kück, N. De Freitas, Learning to recognize objects with little supervision, International Journal of Computer Vision 77 (2008) 219–237.

[30] J. Gleason, A. V. Nefian, X. Bouyssounousse, T. Fong, G. Bebis, Vehicle detection from aerial imagery, in: IEEE International Conference on Robotics and Automation, 2011, pp. 2065–2070.

[31] U. Stilla, E. Michaelsen, U. Soergel, S. Hinz, H. Ender, Airborne monitoring of vehicle activity in urban areas, International Archives of Photogrammetry and Remote Sensing 35 (2004) 973–979.

[32] website, `http://www.ipf.kit.edu/english/downloads\_707.php`.

[33] A. Kembhavi, D. Harwood, L. S. Davis, Vehicle detection using partial least squares, in: IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 33, 2011, pp. 1250–1265.

[34] C. Diegert, A combinatorial method for tracing objects using semantics of their shape, in: Applied Imagery Pattern Recognition Workshop (AIPR), 2010 IEEE 39th, IEEE, 2010, pp. 1–4.

[35] J. Ponce, M. Hebert, C. Schmid, A. Zisserman, Toward Category-Level Object Recognition, Springer, 2006.

[36] J. Davis, M. Goadrich, The relationship between precision-recall and roc curves, in: International Conference on Machine Learning, 2006, pp. 233–240.

[37] L. Lusted, Signal detectability and medical decision-making, Science (1971) 1217–1219.

[38] O. Tuzel, F. Porikli, P. Meer, Region covariance: A fast descriptor for detection and classification, in: European Conference on Computer Vision, 2006, pp. 589–600.

[39] B. P. Raghavan, V., G. S. Jung, A critical investigation of recall and precision as measures of retrieval system performance, in: ACM Transactions on Information Systems, 1989, pp. 205–229.

[40] C. Gu, P. A. Arbelez, Y. Lin, K. Yu, J. Malik, Multi-component models for object detection., in: European Conference on Computer Vision, 2012, pp. 445–458.

[41] K. E. A. van de Sande, J. R. R. Uijlings, T. Gevers, A. W. M. Smeulders, Segmentation as selective search for object recognition., in: International Conference on Computer Vision, 2011, pp. 1879–1886.

[42] M. Pedersoli, A. Vedaldi, J. Gonzlez, A coarse-to-fine approach for fast deformable object detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2011, pp. 1353–1360.

[43] P. F. Felzenszwalb, R. B. Girshick, D. A. McAllester, Cascade object detection with deformable part models., in: IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 2241–2248.

[44] C. D. Manning, P. Raghavan, H. Schütze, Introduction to information retrieval, Cambridge University Press Cambridge, 2008.

[45] G. Liu, R. M. Haralick, Optimal matching problem in detection and recognition performance evaluation, Pattern Recognition 35 (10) (2002) 2125–2139.

[46] X. Jiang, C. Marti, C. Irniger, H. Bunke, Distance measures for image segmentation evaluation, EURASIP Journal on Applied Signal Processing 2006 (2006) 209–209.

[47] A. Hoover, G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bunke, D. B. Goldgof, K. Bowyer, D. W. Eggert, A. Fitzgibbon, R. B. Fisher, An experimental comparison of range image segmentation algorithms, Vol. 18, 1996, pp. 673–689.

[48] V. Y. Mariano, J. Min, J.-H. Park, R. Kasturi, D. Mihalcik, H. Li, D. Doermann, T. Drayer, Performance evaluation of object detection algorithms, in: IAPR International Conference on Pattern Recognition, Vol. 3, 2002, pp. 965–969.

[49] D. Hoiem, Y. Chodpathumwan, Q. Dai, Diagnosing error in object detectors, in: European Conference on Computer Vision, 2012, pp. 340–353.

[50] P. Dollár, Z. Tu, P. Perona, S. Belongie, Integral channel features, in: British Machine Vision Conference, 2009, p. 5.

[51] B. Wu, R. Nevatia, Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors, in: International Conference on Computer Vision, 2005, pp. 90–97.

[52] S. Maji, A. Berg, J. Malik, Classification using intersection kernel support vector machines is efficient, in: IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.

[53] L. Zhu, Y. Chen, A. L. Yuille, W. T. Freeman, Latent hierarchical structural learning for object detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 1062–1069.

[54] X. Wang, A discriminative deep model for pedestrian detection with occlusion handling, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 3258–3265.

[55] W. Ouyang, X. Wang, Joint deep learning for pedestrian detection, in: International Conference on Computer Vision, IEEE, 2013, pp. 2056–2063.

[56] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, 2012, pp. 1097–1105.

[57] A. Sharif Razavian, H. Azizpour, J. Sullivan, S. Carlsson, Cnn features off-the-shelf: an astounding baseline for recognition, arXiv preprint arXiv:1403.6382.

[58] M. Oquab, L. Bottou, I. Laptev, J. Sivic, Learning and transferring mid-level image representations using convolutional neural networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2014.

[59] P. Viola, M. J. Jones, D. Snow, Detecting pedestrians using patterns of motion and appearance, International Journal of Computer Vision 63 (2005) 153–161.

[60] F. M. Porikli, Integral histogram: A fast way to extract histograms in cartesian spaces, in: IEEE Conference on Computer Vision and Pattern Recognition, 2005, pp. 829–836.

[61] P. Sabzmeydani, G. Mori, Detecting pedestrians by learning shapelet features, in: IEEE Conference on Computer Vision and Pattern Recognition, 2007, pp. 1–8.

[62] T. Malisiewicz, A. Gupta, A. A. Efros, Ensemble of exemplar-svms for object detection and beyond, in: International Conference on Computer Vision, 2011, pp. 89–96.

[63] C. H. Lampert, M. B. Blaschko, T. Hofmann, Beyond sliding windows: Object localization by efficient subwindow search, in: IEEE Conference on Computer Vision and Pattern Recognition, 2008.

[64] A. Bosch, A. Zisserman, X. Muoz, Representing shape with a spatial pyramid kernel, in: Conference on Image and Video Retrieval, 2007, pp. 401–408.

[65] W. R. Schwartz, A. Kembhavi, D. Harwood, L. S. Davis, Human detection using partial least squares analysis., in: International Conference on Computer Vision, 2009, pp. 24–31.

[66] X. Wang, T. Han, S. Yan., An hog-lbp human detector with partial occlusion handling, in: International Conference on Computer Vision, 2009, pp. 32–39.

[67] S. Walk, N. Majer, K. Schindler, B. Schiele, New features and insights for pedestrian detection., in: IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 1030–1037.

[68] M. Enzweiler, D. Gavrila, Monocular pedestrian detection: Survey and experiments, in: IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 31, 2009, pp. 2179–2195.

[69] P. F. Felzenszwalb, D. P. Huttenlocher, Pictorial structures for object recognition., International Journal of Computer Vision 61 (2005) 55–79.

[70] L. D. Bourdev, J. Malik, Poselets: Body part detectors trained using 3d human pose annotations, in: International Conference on Computer Vision, 2009, pp. 1365–1372.

[71] Y. Yang, D. Ramanan, Articulated pose estimation with flexible mixtures-of-parts., in: IEEE Conference on Computer Vision and Pattern Recognition, 2011, pp. 1385–1392.

[72] A. Vedaldi, V. Gulshan, M. Varma, A. Zisserman, Multiple kernels for object detection., in: International Conference on Computer Vision, 2009, pp. 606–613.

[73] B. Alexe, T. Deselaers, V. Ferrari, What is an object?, in: IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 73–80.

[74] E. Rahtu, J. Kannala, M. B. Blaschko, Learning a category independent object detection cascade., in: International Conference on Computer Vision, 2011, pp. 1052–1059.

[75] H.-Y. Cheng, C.-C. Weng, Y.-Y. Chen, Vehicle detection in aerial surveillance using dynamic bayesian networks, Image Processing, IEEE Transactions on 21 (4) (2012) 2152–2159.

[76] website, Utah agrc website (2012).
URL `\url{http://gis.utah.gov/}`

[77] C. M. Bishop, et al., Pattern recognition and machine learning, Vol. 1, springer New York, 2006.

[78] T. Ojala, M. Pietikainen, T. Maenpaa, Multiresolution gray-scale and rotation invariant texture classification with local binary patterns, in: IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, 2002, pp. 971–987.

[79] X. Tan, B. Triggs, Enhanced local texture feature sets for face recognition under difficult lighting conditions, in: Analysis and Modeling of Faces and Gestures, 2007, pp. 168–182.

[80] R. Brunelli, Template Matching Techniques in Computer Vision: Theory and Practice, 2009.

[81] J. Gall, V. Lempitsky, Class-specific hough forests for object detection, in: 2013 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 1022–1029.

[82] T. Joachims, Making large-scale support vector machine learning practical, `http://svmlight.joachims.org/` (1999).

[83] L. Rousseeuw, L. Kaufman, Clustering by means of medoids, Statistical data analysis based on the L1-norm and related methods 405–416.

[84] D. Park, D. Ramanan, C. Fowlkes, Multiresolution models for object detection, in: European Conference on Computer Vision, 2010, pp. 241–254.

[85] J. Yan, X. Zhang, Z. Lei, S. Liao, S. Z. Li, Robust multi-resolution pedestrian detection in traffic scenes, in: IEEE Conference on Computer Vision and Pattern Recognition, Vol. 1, 2013.

[86] P. Dollar, C. Wojek, B. Schiele, P. Perona, Pedestrian detection: An evaluation of the state of the art, Vol. 34, 2012, pp. 743–761.

[87] S. K. Divvala, A. A. Efros, M. Hebert, How important are deformable parts in the deformable parts model?, in: European Conference on Computer Vision, 2012, pp. 31–40.