Fast Boosting based Detection using Scale Invariant Multimodal Multiresolution Filtered Features

Arthur Daniel Costea, Robert Varga and Sergiu Nedevschi
Image Processing and Pattern Recognition Research Center
Technical University of Cluj-Napoca, Romania
{arthur.costea, robert.varga, sergiu.nedevschi}@cs.utcluj.ro

Abstract

In this paper we propose a novel boosting-based sliding window solution for object detection which can keep up with the precision of the state-of-the art deep learning approaches, while being 10 to 100 times faster. The solution takes advantage of multisensorial perception and exploits information from color, motion and depth. We introduce multimodal multiresolution filtering of signal intensity, gradient magnitude and orientation channels, in order to capture structure at multiple scales and orientations. To achieve scale invariant classification features, we analyze the effect of scale change on features for different filter types and propose a correction scheme. To improve recognition we incorporate 2D and 3D context by generating spatial, geometric and symmetrical channels. Finally, we evaluate the proposed solution on multiple benchmarks for the detection of pedestrians, cars and bicyclists. We achieve competitive results at over 25 frames per second.

1. Introduction

Due to the fast evolution of intelligent vehicles, there is a pressing need for robust but also real-time environment perception solutions in order to enable advanced driver assistance or autonomous driving. One of the main perception tasks is the detection of traffic participants, which is still an active research problem. The performance of state-of-the-art solutions is getting closer and closer to human level recognition [46]; however, results are far from saturation and there is still room for improvement.

The current benchmarks are dominated by deep learning solutions that are able to automatically learn image features at multiple abstraction levels from raw image data. Unfortunately, these powerful approaches come with a high computational cost. The fastest top-performing deep learning solutions struggle to achieve a processing speed of 2-3 FPS even with high-end GPUs.

Our main goal is to provide a fast solution that can keep up with the current best performing deep learning solutions. We also focus on efficient ways to exploit additional information such as motion and depth. There are some existing solutions that use motion [33], depth from stereo [26] or LIDAR data [21] to improve detection results from mono. However, current benchmarks are dominated by solutions that use as input only individual monocular images and seem to outperform the current multimodal solutions, or report only minor improvements [47]. As noted in [3], the efficient use of other modalities has not been explored yet.

In this work we consider as baseline the solution proposed in [10], which relies on multiresolution filtered channel features. It is a sliding window type approach and applies a fast filtering scheme over LUV+HOG channels to generate classification features. We focus on improving classification features and exploring efficient ways to incorporate multimodal features. The main contributions of this paper are:

- multimodal multiresolution channels;
- a feature correction scheme for achieving scale invariant classification features;
- exploitation of 2D and 3D context information;
- a common framework for detecting pedestrians, cars and bicyclists.

2. Related work

The progress of the pedestrian detection algorithms is due to the existence of numerous and extensive benchmarks. Pushing the boundaries for obtaining better results each year on these datasets has lead to the fast evolution of detection methods. Some of the most relevant benchmarks are: Inria[12], Caltech-USA [18], KITTI [22].

For a comprehensive review on the best performing detection methods the reader is invited to consult recent reviews and surveys. The review from [3] indicates that improvements due to newly proposed features will continue.

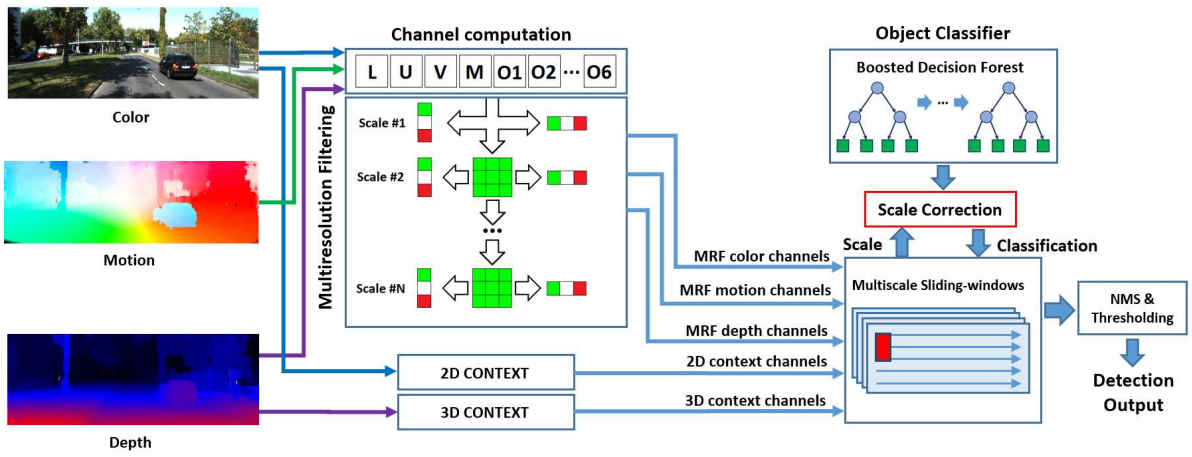

Figure 1. System overview.

It also recommends and evaluates introducing optical flow, context information and other complementary information sources to improve detection accuracy. In [46] the authors investigate how far the current approaches are from an ideal single frame detector. In 2015 the best methods made ten times more errors than human annotators, indicating that there is room for improvement.

Features - Histogram of Oriented Gradients [12] was the original feature proposed for the specific task of pedestrian detection. The introduction of Haar-type features [41] with integral image calculation [40] enabled real-time detection. Generalization of Haar features which make use of different image channels was the next step forward as shown in [17]. Since then the majority of approaches rely on such types of features but innovative improvements have been proposed: locally decorrelated filters [31]; different checkerboard filtered feature patterns [47]; rotated filters [46].

Deep learning methods such as [24, 29] have reached state-of-the-art performance. The underlying image features are automatically learned by the convolutional layers of the network.

Multiscale - There are several ways to address the issue of detecting pedestrians at multiple scales. In the original work by Dalal and Triggs [12] the classifier model had a fixed dimension and the input image was resized multiple times to detect pedestrians at smaller scales. This had a clear computational burden of recomputing the features at several scales. Another alternative is to consider separate models for each scale such as the work from [2]. A hybrid approach by [16] proposed to recalculate features only at each octave and to perform approximations in between octaves for a faster feature extraction phase.

Multimodal - The review from [3] recommends employing information from complementary sources such as color, optical flow, depth and context to improve detection performance. Several works focused on exploiting these modalities and proposed features based on motion [43] [13] [20]; infrared imagery [27]; depth from LIDAR [38] [28] [19] [34] and depth from stereo [20] [45] [27].

The work in [23] introduces Multiview classifiers trained on multimodal information fused from RGB and depth maps. The dense depth maps are obtained via interpolation from the sparse 3D laser pointcloud.

The authors in [8] describe an approach that leverages both image and 3D information by utilizing CNNs applied on the LIDAR bird's eye view, LIDAR front view and the RGB image.

Multimodal information can be also used for object proposal generation. In [7] high quality 3D object proposals are obtained relying on stereo reconstruction.

3. Proposed solution

We propose a novel multimodal multiresolution approach for object detection introducing multiple key concepts for achieving robust detection at low computational costs. The solution takes advantage of multisensorial perception and exploits information from color, motion and depth. We introduce multimodal multiresolution filtering of signal intensity, gradient magnitude and orientation channels, in order to capture structure at multiple scales and orientations. Objects are detected using multiscale sliding windows with a boosting-based classifier. We propose a correction scheme to ensure scale invariance of classification features even after multiple iterations of low pass and high pass filters. To improve the robustness we introduce 2D and 3D context by generating spatial, geometric and symmetry channels. An overview of the solution is illustrated in Figure 1.

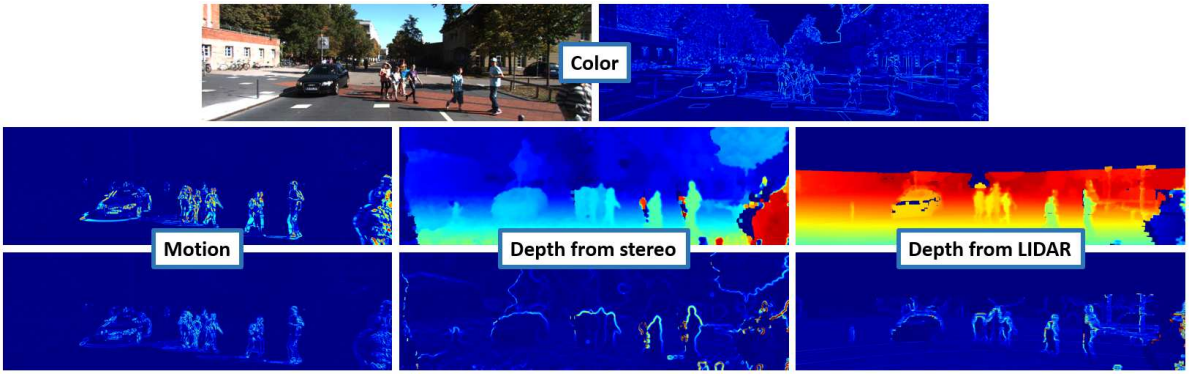

Figure 2. Multimodal channels - from top to bottom: color image and its gradient magnitude; temporal difference; 3D point cloud depth; stereo depth map; gradient magnitude for the channels above

3.1. Multimodal detection

Multimodal data can serve object detection as a source for context and also for local structure. We create a dense intensity image for each modality and use them to generate intensity, gradient magnitude and orientation channels. In Figure 2 we illustrate the gradient magnitudes for different modalities and it can be seen that each modality highlights different edge types. A boosting classifier can learn to select and combine relevant features from different modalities. We consider three types of input for multimodal detection: color, motion and depth.

Color - We rely on the LUV color transform that proved to be the most efficient for pedestrian detection approaches in [17].

Motion - A simple way for capturing motion is to compute the difference between two consecutive frames. Significant improvements for motion based detection were achieved by aligning the previous frame using coarse optical flow [33]. This way the temporal difference is able to capture relative motion and can be powerful especially for articulated object types.

Depth - Depth can be recovered in real-time from stereo image pairs using a fast stereo reconstruction solution such as rSGM [37] or directly from the available 3D LIDAR point cloud. In both cases, we use interpolation for regions without 3D measurement in order to achieve a dense representation. In the case of stereo images we have much denser reconstruction, but the accuracy decreases with the distance from camera. In the case of 3D point cloud the depth is interpolated using inverse distance weighing from a very low number of measurements (0.01 density) and the maximum height is limited to around 2 meters. The precision of the measurements is higher compared to stereo reconstruction and it does not decrease with distance. It is also independent of image quality and lighting conditions.

3.2. Multimodal multiresolution filtered channels

For each of the previously described intensity inputs we compute a normalized gradient magnitude and magnitudes at 6 orientations, resulting in a total of 10 channels for color and 8 for motion and depth. To capture multimodal edges at multiple scales and multiple orientations we apply a fast multiresolution filtering scheme [10]. A 3×3 box filter is used multiple times iteratively to generate smoothed images at multiple scales. Vertical and horizontal difference is applied at each scale for additional high-pass filtering. Due to the simplicity of the features, computation is possible in less than 3 ms per VGA resolution image on a GPU. We opt for the removal of the aggregation step from [10] in order to increase the resolution of the filtered channels.

3.3. Achieving scale invariance for multiscale detection

A fast solution for multiscale detection is to use a single image scale, resulting in very fast feature computation, and apply sliding windows at multiple scale. This was achieved in [10] using a single flexible classification model that could be used for the classification of sliding windows at multiple scales. The classification features for a detection window were sampled using a grid of 20×10 samples from the filtered channels and the grid was adapted to the window size. The scale invariance of the classification features is lost due to the use of a single image feature scale and single classifier model for all pedestrian scales. Providing scale invariance for the classification features should further increase detection robustness.

We define the ratio function (or correction factor) for a classification feature f as: the feature value extracted at scale s divided by the feature value at the original scale. It is important to note, that due to rescaling, the position of the same classification feature changes with the scale, resulting in:

$$r_f(s) = f(s, x, y)/f(1, x/s, y/s) \quad (1)$$

Our goal is to extract classification features at any scale using only the raw image features computed at the original scale. For this we need a model for the function $r_f(s)$, enabling us to write:

$$f(s, x, y) = r_f(s) \cdot f(1, x/s, y/s) \quad (2)$$

We will determine the form of the ratio function for different feature types. As baseline feature types we have color, gradient magnitude and gradient orientation bins. The baseline features can be filtered using smoothing low pass filters or first order difference filters (horizontal or vertical), resulting in additional feature types. In the first part we ignore discretization errors, resizing artifacts and consider the image as a continuous signal to determine the form of the ratio function theoretically. In the second part, we collect data from the Caltech dataset and perform a linear fit to find the form of the ratio function empirically.

Theoretical estimation - In the following we estimate the theoretical ratio between the classification features from s times larger/smaller bounding boxes and from the original bounding boxes. Note, that the bounding boxes represent the same object at different sizes. Color features should not change with the scale due to scale invariance:

$$I_s(x, y) = I(x/s, y/s) \quad (3)$$

where I_s denotes the image at scale s and I represents the image at the original scale. This shows that $r_I(s) = 1$ for color features.

For the gradient magnitude we have $r_M(s) = s^{-1}$ since:

$$M_s(x, y) = \frac{1}{s} M(x/s, y/s) \quad (4)$$

The factor is also transmitted to gradient orientation bin features since these are proportional to the gradient magnitude.

In the case of smoothing operations the correction factor is not changed. Applying the derivative over color shows that $r_{Idx}(s) = s^{-1}$ since:

$$\frac{\partial}{\partial x} I_s(x, y) = \frac{1}{s} \frac{\partial}{\partial x} I(x/s, y/s) \quad (5)$$

The derivative over gradient magnitude results in $r_{Mdx}(s) = s^{-2}$.

Empirical estimation - In order to estimate the correction factors empirically, we extract features from the Caltech dataset at multiple scales. We follow the protocol described in [16] for approximating each feature type using the ratio function:

$$f(s) = ae^{-\lambda s} f(0) = exp(log(a) - \lambda s) f(0) \quad (6)$$

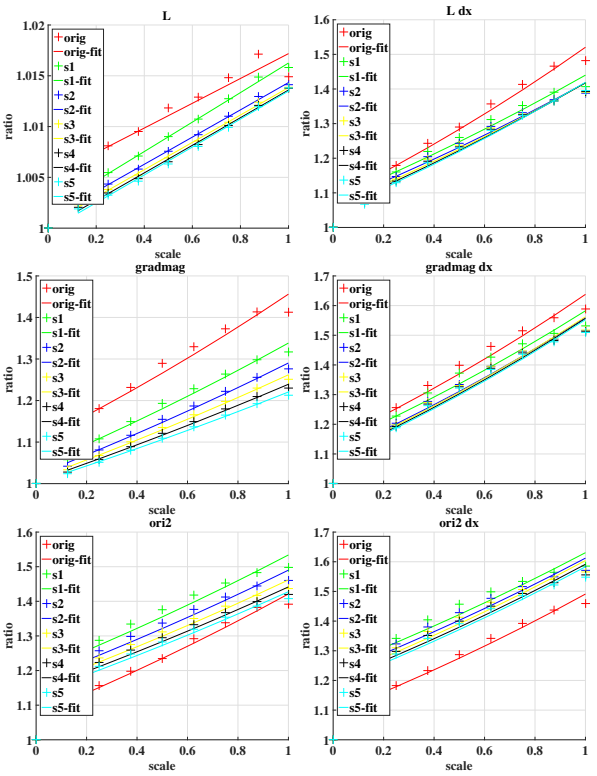

Figure 3. Scale correction factors for different feature types. The factors represent the ratio between the feature value at scale s and the feature value at the original scale. The figures show on the x-axis the scale $-log_2(s)$ (0 - original scale; 1 - downsampling by a factor of 2) and on the y-axis the ratio function $r(-log_2(s))$ for different feature types. Left: L-channel, gradient magnitude and second gradient orientation bin channel. Right: the same channels with horizontal derivative filter. The different lines from the graph plot the behavior of the original channel and the 5 iteratively smoothed channels, i.e. *sx* signifies *x* number of smoothing filters.

where $f(s)$ is the feature after a downsampling of 2^s and $f(0)$ is the feature at the original scale. According to the previous model, a linear fit for $log(f(s)/f(0))$ determines a and λ. Note, that we approximate the ratio function for pointwise features, whereas in [16] sums over rectangular regions were considered. To obtain graphs compatible with their work we would need to plot $\frac{f(s)}{2^{2s} f(0)}$ as a function of s because the sum over the rectangular region introduces the $(2^s)^2$ term.

Figure 3 shows the data points that indicate the mean ratio and the linear fit for 6 representative feature types. We plot the ratio function in terms of $-log_2(s)$ for values $s \in [0.5, 1]$ (one octave). This was used for linear fit and $r(s)$ is obtainable via a variable change.

The graphs show that: for color channels, the features retain their values after resize operations, just as the theoretical model predicted: $r_I(s) = 1$; partial derivative oper-

ations do not conform to the theoretical model: $r_{Idx}(s) = s^{-0.585}$ (at a shrinking factor of 2 it is around 1.5 and not 2); smoothing operations decrease the exponent further; the first smoothing operation for orientation features behaves differently (see the change between *orig* to *s1* and *s1* to *s2*).

Having determined the correction factors both theoretically and empirically, we can apply them for scale correction. Due to the decrease in accuracy of the approximation with the scale change, we recompute the image features for each octave and use the approximation only for the intermediate scales as in [16].

3.4. Context channels

Pedestrians and vehicles are bounded by spatial and geometric constraints. In the following we define such constraints and incorporate them as context channels next to the multiresolution filtered channels. This way we can enable the boosting classifiers to learn the context of pedestrians or other object types.

3.4.1 2D context

Traffic scenario images captured from vehicle mounted cameras tend to have a stable spatial layout. The position and layout of pedestrians in the 2D images is constrained by camera parameters and bounded by 3D size and 3D position. Objects can appear in any place, but we focus only on those which stand on the ground plane. Some approaches learned the spatial distribution of different object types in 2D images and incorporated them as spatial priors for semantic segmentation [14, 36]. Instead of using a constant prior we use spatial channels (used originally for segmentation [9]) that enable the boosting classifier to learn constraints on vertical and horizontal position as classification features. The filtered channels are extended with 3 additional channels consisting of a vertical, horizontal and symmetric-horizontal channel. These channels have values from 0 to 1 and represent the normalized vertical and horizontal position in each location of the 2D image (see Figure 4). The employed boosting classifier can learn 2D constraints on the top, center and bottom part of the sliding window by simply learning thresholds over channel features from these 2D spatial channels.

We also introduce symmetry channels that capture vertical edge symmetries at multiple ranges. For example, shorter ranges capture the legs or the head, while the longer ranges capture the whole torso. We define an individual channel for each range in the form of a symmetry cost:

$$S_r(x,y) = \sum_{i=r/2}^{r} \left(\frac{D_x(x-i,y) - D_x(x+i,y)}{D_x(x-i,y) + D_x(x+i,y)} \right)^2 \quad (7)$$

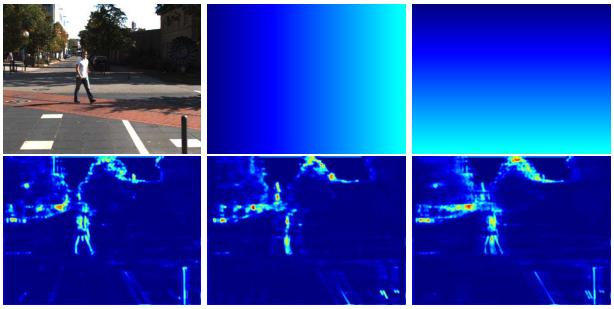

Figure 4. 2D context channels - from top left: input; horizontal; vertical; S_6 symmetry; S_{12} symmetry; $S_6 + S_{12}$ symmetry channels.

where D_x is the partial derivative along the x-axis and r is the symmetry range. In our experiments we used the pixel ranges $r \in \{6, 12, 18, 24\}$. We also generate a channel that is a sum of all these symmetry channels to obtain a range independent channel (Figure 4) at octave level. The computation of these channels takes less than 1 ms on a GPU.

3.4.2 3D context

Using 3D information from stereo or LIDAR it is possible to learn the 3D context of the traffic participants. We segment the image into 16000 superpixels having an average size of around 100 pixels and permiting the capturing of body parts for far pedestrians. We implement an approximation of SLIC [1] segmentation on GPU, achieving a runtime of less than 2 ms for a 0.5 MP image. We compute the 3D position of each superpixel using hybrid median filtering and generate normalized 3D spatial context channels for the X, Y, and Z coordinates. For example the Y channel represent the height above ground at each pixel location. For better robustness, we estimate the road plane using a fast RANSAC based approach and correct the 3D points via a rotation that aligns the plane normal to the the y axis. We additionally generate a binary channel for the ground by marking all image points that are at a height of at most 20 cm above the ground.

For the geometric context channels we propose the use of a simple but very fast 3D clustering method. For grouping we use superpixel-level region growing and as grouping criteria we use an absolute threshold of 0.5 meters in the case of depth from LIDAR and a relative threshold of 2.5% from distance to camera in the case of depth from stereo. We ignore superpixels that belong to the ground. Finally, we determine the height, width and area of each group and save as normalized values for each pixel of each group, resulting in the geometric context channels. These channels enable classifiers to learn geometric constraints for objects by learning numerical thresholds over channel values. These channels are illustrated in Figure 5.

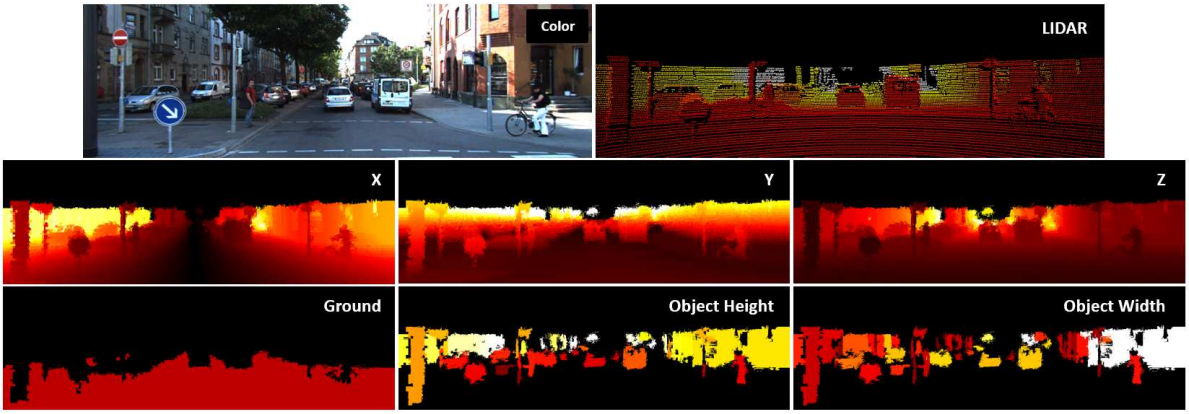

Figure 5. 3D context channels - from top left: color; projected 3D point cloud; X; Y; Z; ground plane; object height and width

4. Detecting multiple object types

In the case of pedestrians we opt for a single detector with a fixed aspect ratio. In order to handle object types with highly variable aspect ratios, we use different windows for different aspect ratio ranges. A simple solution would be to divide the positive sample set into equal parts and use a detection window with a fixed aspect ratio that maximizes the intersection over union overlapping criteria for the subset. In the case of cars, a minimum overlap of 85% (70% required at evaluation) can be achieved for all aspect ratios using only 5 fixed aspect ratios. Dividing the dataset based on appearance and orientation [25, 32] can provide better results, however it would increase computational costs at detection time for classification and a minimal number of detectors is preferable.

Training protocol - For sliding window classification we train Adaboost classifiers using 5-level decision trees as weak learners. We train an initial classifier using NrP (number of positive samples) and NrP random negative samples. Then we use multiple bootstrapping rounds in order to generate NrP additional hard negatives iteratively, until the hard negative count gets below NrP. For the first 4 classifiers we use 256, 512, 1024 and 2048 weak learners respectively, and for the rest of the rounds 4096. For better generalization, the learning rate of the boosting algorithm is adapted by a shrink factor as recommanded in [25]. To accelerate training and further reduce overfitting, we consider a random subset of only 1% of the classification features for each feature selection. To accelerate prediction, we use soft cascading with a variable rejection threshold decreasing from 1 with a step of 0.01 after each weak learner [15] at training time and 0.02 at testing time. This is in contrast to traditional approaches which fix the rejection threshold at -1 during training. We apply it in order to generate more hard negative samples.

5. Experimental Results

In order to assess the performance of the proposed solution and to compare it with the current state of art, we evaluated it on multiple detection benchmarks in the context of traffic environments. We consider the Caltech-USA dataset [18] for pedestrian detection, KITTI-object [22] for pedestrians and cars and Tsinghua-Daimler dataset [30] for cyclists. In the following we abbreviate the current approach as **MM-MRFC** standing for multimodal multiresolution filtered channels.

5.1. Caltech - Pedestrian

We provide the results on the Caltech dataset both using the standard training set and with the 10 times larger extended training set. We use the standard dataset for analyzing the improvements provided by each component of the solution and use the extended set for comparison with other solutions. In both cases we evaluate the log-average miss rate (MR) in the $[10^{-2}, 10^{0}]$ false positives per image (FPPI) range for the reasonable test setup.

In Table 1 we show the incremental improvements of the proposed solution. Both the theoretically and the empirically approximated scale correction schemes provide a significant performance gain. In all further experiments we use the theoretical scale correction factors due to their simplicity and their similar performance compared to the empirical scale correction factors. After adding scale correction and 2D context channels, a MR of 18.26% is achieved which is currently the lowest miss rate reported for training with the standard training set and using only color information. Further improvements are obtained with SDt [33] motion channel features and by applying multiresolution filtering over SDt motion channels.

In Figure 6 we provide a comparison with the sate of art on the Caltech benchmark using the *reasonable* setup. The proposed solution achieves a MR of 12.31% provid-

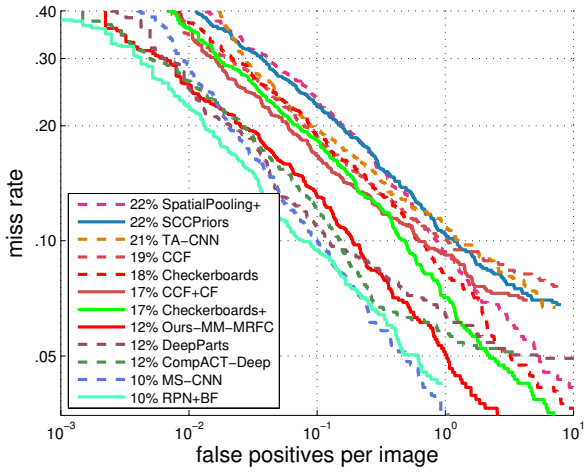

Figure 6. Comparison to the state-of-the-art on Caltech-USA pedestrian benchmark (reasonable test setup).

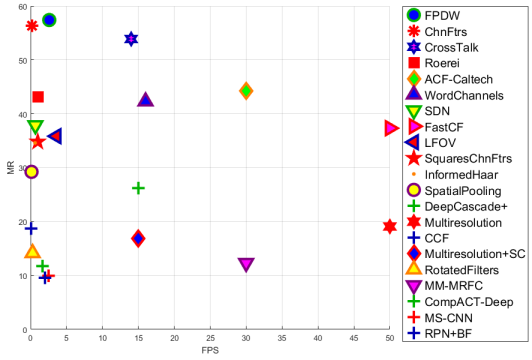

Figure 7. Caltech test set results. Average miss rate (MR) plotted against execution time (FPS) for multiple approaches. FPS is capped at 50 for better visualization (FastCF [11] is at 105 FPS and Multiresolution [10] at 60 FPS).

ing an improvement of 5% over the previous best performing boosting based solution (Checkerboards+ [47]). It also compares well with the best performing deep learning based solutions (all approaches with MR below 17%). It is important to highlight that our detector is capable of running at 30 FPS on a GPU, having an average execution time of 32 ms for a single image on system with an Intel i7 3.0 GHz CPU and an Nvidia GeForce GTX 980 Ti GPU. In Figure 7 we provide a comparison based on MR and frame rate (FPS) with all solutions that reported their execution time.

5.2. KITTI - Object

In order to evaluate the detection of pedestrians and cars on the KITTI benchmark [22] we follow the standard evaluation protocols. Object detection performance is measured by computing the average precision (AP) for the recall

Channel Type		**Caltech MR** - reasonable -
Color	MRFC no SC	24.46
	MRFC E-SC	22.69
	MRFC T-SC	22.84
	+ 2D spatial	20.80
	+ 2D symmetry	18.26
Motion	+ SDt	17.29
	+ MM-MRFC	16.11

Table 1. Results on Caltech test set using different scale correction schemes and multimodal feature channel types. Scale correction: no SC - without scale correction; E-SC - with empirical correction factors; T-SC with theoretical correction factors.

Context Type		**KITTI AP** Easy	Moderate	Hard
Color	MRFC no SC	62.84	59.98	51.10
	MRFC	67.14	61.45	52.76
	+ 2D spatial	69.58	63.83	54.83
	+ 2D symmetry	70.28	64.75	55.66
3D stereo	+ 3D spatial	77.88	70.30	60.63
	+ 3D geometric	77.97	70.61	61.47
	+ MRFC	82.53	74.82	65.95
3D LIDAR	+ 3D spatial	77.88	70.93	61.91
	+ 3D geometric	79.92	72.48	63.13
	+ MRFC	84.26	76.34	67.18
Motion	+MRFC	85.25	77.72	68.28

Table 2. Results on KITTI validation set using different feature channel types. Performance is measured in AP (%) for easy, moderate and hard test setups. Feature scale correction is employed in all feature setups, except *MRFC no SC*. The best result is achieved using color, depth from LIDAR, and motion.

range of [0, 1]. For the validation of the proposed features we evaluate the performance on the validation set using the validation/training split from [7]. In Table 2 we show the incremental improvements of the proposed features. Each proposal increases the AP values demonstrating the usefulness of the new feature channels.

The results for the test set compared to other approaches can be found in Table 3. We present the results for pedestrians and cars. Because the number of training samples for the bicyclist class is small, we evaluate this object class on another dataset (see next subsection). It can be seen that a competitive performance is achieved for both object classes at significantly lower computational costs. Pedestrian detection runs at 25 FPS and car detection at 20 FPS. The proposed solution achieves the highest AP among multimodal or boosting-based solutions and is comparable with the best performing deep learning based solutions.

Method	Input	Time		Cars			Pedestrians		
				Easy	Moderate	Hard	Easy	Moderate	Hard
FusionDPM [34]	C DL	30s	CPU	-	-	-	59.51	46.67	42.05
ACF [15]	C	1s	CPU	-	-	-	60.11	47.29	42.90
VOTE-3Deep [19]	C DL	1.5s	CPU	76.79	68.24	63.23	68.39	55.37	52.59
MV-RGBD-RF [23]	C DL	4s	GPU	-	-	-	73.30	56.59	49.63
FilteredICF [47]	C	2s	CPU	-	-	-	67.65	56.75	51.12
DeepParts [39]	C	1s	GPU	-	-	-	70.49	58.67	52.78
CompACT-Deep [5]	C	1s	GPU	-	-	-	70.69	58.74	52.71
Regionlets [42]	C	1s	CPU	84.75	76.45	59.70	73.14	61.15	55.21
Faster-RCNN [35]	C	2s	GPU	86.71	81.84	71.12	78.86	65.90	61.18
Mono 3D [6]	C	4.2s	GPU	92.33	88.66	78.96	80.35	66.68	63.44
3DOP [7]	C DS	3s	GPU	93.04	88.64	79.10	81.78	67.47	64.70
SDP+RPN [44]	C	0.4s	GPU	90.14	88.85	78.38	80.09	70.16	64.82
MS-CNN [4]	C	0.4s	GPU	90.03	89.02	76.11	83.92	73.70	68.31
MM-MRFC	C DL F	0.05s	GPU	90.63	88.45	78.32	82.18	70.02	64.74

Table 3. Comparison with the state-of-the-art on the KITTI object benchmark (test set). For each approach we report the input modalities (C - color; DS - depth from stereo; DL - depth from LIDAR, F - flow), execution time (CPU or GPU) and average precision (%) for cars and pedestrians under easy, moderate and hard test setups.

5.3. Tsinghua-Daimler - Cyclist

The Tsinghua-Daimler benchmark [30] is an ideal benchmark for evaluating the detection of bicyclists, considering that it contains 22161 annotated cyclist instances in over 30000 images. These were recorded in the urban traffic of Beijing. The dataset also provides 3D stereo data for each image frame. The evaluation protocol is the same as for the KITTI detection benchmark.

Currently, only cyclists with a height of at least 60 pixels are annotated in the training dataset, even though the test set is fully labeled with cyclists having heights grater than 20 pixels. For this reason, we choose to evaluate the performance of our solution only for cyclists having a height of at least 60 pixels in a 2048×1024 pixel image (*Easy* test setup).

Multiple approaches were evaluated in [30] such as: traditional boosting-based solutions (ACF, LDCF); deep learning approaches with different object proposals (Selective Search, Edge Boxes, Stereo Proposal) and architectures (VGG, ZF); deformable part models (DPM). We train three detectors for narrow, intermediate and wide bicyclists similarly to other sliding window approaches from [30]. Table 4 provides a comparison in terms of AP for the *Easy Ignore* and *Easy Discard* test setups. In the first case, the false detections for other similar classes, such as pedestrians or other riders, are ignored. The previous best performance was achieved by DPM-bboxpred [30] relying on deformable part models and object proposal from stereo. Our proposed solution achieves a slight improvement in AP for the *Ignore* case and a significant improvement of 5% for the *Discard* case, achieving highest reported APs on the benchmark at 25 FPS.

Method	Easy Ignore	Easy Discard
SS-FRCN-VGG	76.7	63.8
EB-FRCN-VGG	83.8	72.6
SP-FRCN-VGG	87.2	78.6
DPM	89.4	81.6
LDCF	89.8	76.2
ACF	89.8	77.8
DPM-bboxpred	90.5	82.3
MM-MRFC	90.7	87.1

Table 4. Comparison with the state-of-the-art based on average precision (%) on the Tsinghua-Daimler cyclist benchmark.

6. Conclusions

In this paper we have introduced an object detection system that relies on several innovative proposals. First, it makes use of information coming from multiple complementary modalities: color, depth and motion. Second, it relies on multiresolution filtered channels for constructing discriminative features for detection. Third, it employs a scale correction scheme based on both theoretical and empirical considerations. Fourth, it proposes several contextual feature channels such as: 2D context, symmetry channels, 3D context, 3D geometrical channels.

Experimental results on multiple benchmarks show that the method achieves top performance while being ten to a hundred times faster than its competitors. It also shows, that although deep learning approaches may dominate the field, traditional sliding window approaches can offer a low cost alternative to these while being competitive.

Acknowledgment. This work was supported by the EU H2020 project UP-Drive under grant nr. 688652.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. In *PAMI*, 2012. 5

[2] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool. Pedestrian detection at 100 frames per second. In *CVPR*, 2012. 2

[3] R. Benenson, M. Omran, J. Hosang, and B. Schiele. Ten years of pedestrian detection, what have we learned? In *ECCV*, 2014. 1, 2

[4] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. In *ECCV*, 2016. 8

[5] Z. Cai, M. Saberian, and N. Vasconcelos. Learning complexity-aware cascades for deep pedestrian detection. In *ICCV*, 2015. 8

[6] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun. Monocular 3d object detection for autonomous driving. In *CVPR*, 2016. 8

[7] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun. 3d object proposals for accurate object class detection. In *NIPS*, 2015. 2, 7, 8

[8] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-view 3d object detection network for autonomous driving. In *CVPR*, 2017. 2

[9] A. D. Costea and S. Nedevschi. Fast traffic scene segmentation using multi-range features from multi-resolution filtered and spatial context channels. In *IV*, 2016. 5

[10] A. D. Costea and S. Nedevschi. Semantic channels for fast pedestrian detection. In *CVPR*, 2016. 1, 3, 7

[11] A. D. Costea, A. V. Vesa, and S. Nedevschi. Fast pedestrian detection for mobile devices. In *ITSC*, 2015. 7

[12] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 1, 2

[13] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *ECCV*, 2006. 2

[14] S. Di, H. Zhang, X. Mei, D. Prokhorov, and H. Ling. Spatial prior for nonparametric road scene parsing. In *ITSC*, 2015. 5

[15] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. In *PAMI*, 2014. 6, 8

[16] P. Dollár, S. Belongie, and P. Perona. The fastest pedestrian detector in the west. In *BMVC*, 2010. 2, 4, 5

[17] P. Dollár, Z. Tu, P. Perona, and S. Belongie. Integral channel features. In *BMVC*, 2009. 2, 3

[18] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. In *PAMI*, 2012. 1, 6

[19] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In *arXiv:1609.06666*, 2016. 2, 8

[20] M. Enzweiler and D. M. Gavrila. A multilevel mixture-of-experts framework for pedestrian classification. In *TIP*, 2011. 2

[21] A. Ess, B. Leibe, K. Schindler, and L. Van Gool. Robust multiperson tracking from a mobile platform. In *PAMI*, 2009. 1

[22] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 1, 6, 7

[23] A. González, D. Vázquez, A. M. López, and J. Amores. On-board object detection: Multicue, multimodal, and multiview random forest of local experts. In *IEEE transactions on Cybernetics*, 2016. 2, 8

[24] J. Hosang, M. Omran, R. Benenson, and B. Schiele. Taking a deeper look at pedestrians. In *CVPR*, 2015. 2

[25] Q. Hu, S. Paisitkriangkrai, C. Shen, A. van den Hengel, and F. Porikli. Fast detection of multiple objects in traffic scenes with a common detection framework. In *TITS*, 2016. 6

[26] C. G. Keller, M. Enzweiler, M. Rohrbach, D. F. Llorca, C. Schnorr, and D. M. Gavrila. The benefits of dense stereo for pedestrian detection. In *TITS*, 2011. 1

[27] S. J. Krotosky and M. M. Trivedi. On color-, infrared-, and multimodal-stereo approaches to pedestrian detection. In *TITS*, 2007. 2

[28] B. Li, T. Zhang, and T. Xia. Vehicle detection from 3d lidar using fully convolutional network. In *RSS*, 2016. 2

[29] J. Li, X. Liang, S. Shen, T. Xu, and S. Yan. Scale-aware fast r-cnn for pedestrian detection. In *arXiv:1510.08160*, 2015. 2

[30] X. Li, F. Flohr, Y. Yang, H. Xiong, M. Braun, S. Pan, K. Li, and D. M. Gavrila. A new benchmark for vision-based cyclist detection. In *IV*, 2016. 6, 8

[31] W. Nam, P. Dollár, and J. H. Han. Local decorrelation for improved pedestrian detection. In *NIPS*, 2014. 2

[32] E. Ohn-Bar and M. M. Trivedi. Learning to detect vehicles by clustering appearance patterns. In *TITS*, 2015. 6

[33] D. Park, C. L. Zitnick, D. Ramanan, and P. Dollár. Exploring weak stabilization for motion feature extraction. In *CVPR*, 2013. 1, 3, 6

[34] C. Premebida, J. Carreira, J. Batista, and U. Nunes. Pedestrian detection combining rgb and dense lidar data. In *IROS*, 2014. 2, 8

[35] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 8

[36] T. Scharwächter and U. Franke. Low-level fusion of color, texture and depth for robust road scene understanding. In *IV*, 2015. 5

[37] R. Spangenberg, T. Langner, S. Adfeldt, and R. Rojas. Large scale semi-global matching on the cpu. In *IV*, 2014. 3

[38] L. Spinello, R. Triebel, and R. Siegwart. Multiclass multimodal detection and tracking in urban environments. In *IJRR*, 2010. 2

[39] Y. Tian, P. Luo, X. Wang, and X. Tang. Deep learning strong parts for pedestrian detection. In *ICCV*, 2015. 8

[40] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, 2001. 2

[41] P. Viola and M. J. Jones. Robust real-time face detection. In *IJCV*, 2004. 2

[42] X. Wang, M. Yang, S. Zhu, and Y. Lin. Regionlets for generic object detection. In *ICCV*, 2013. 8

[43] C. Wojek, S. Walk, and B. Schiele. Multi-cue onboard pedestrian detection. In *CVPR*, 2009. 2

[44] F. Yang, W. Choi, and Y. Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *CVPR*, 2016. 8

[45] J. J. Yebes, L. M. Bergasa, and M. García-Garrido. Visual object recognition with 3d-aware features in kitti urban scenes. In *Sensors*, 2015. 2

[46] S. Zhang, R. Benenson, M. Omran, J. Hosang, and B. Schiele. How far are we from solving pedestrian detection? In *CVPR*, 2016. 1, 2

[47] S. Zhang, R. Benenson, and B. Schiele. Filtered channel features for pedestrian detection. In *CVPR*, 2015. 1, 2, 7, 8