Integral Channel Features

Piotr Dollár[1]
pdollar@caltech.edu
Zhuowen Tu[2]
zhuowen.tu@loni.ucla.edu
Pietro Perona[1]
perona@caltech.edu
Serge Belongie[3]
sjb@cs.ucsd.edu

[1] Dept. of Electrical Engineering
California Institute of Technology
Pasadena, CA, USA

[2] Lab of Neuro Imaging
University of CA, Los Angeles
Los Angeles, CA, USA

[3] Dept. of Computer Science and Eng.
University of California, San Diego
San Diego, CA, USA

Abstract

We study the performance of 'integral channel features' for image classification tasks, focusing in particular on pedestrian detection. The general idea behind integral channel features is that multiple registered image channels are computed using linear and non-linear transformations of the input image, and then features such as local sums, histograms, and Haar features and their various generalizations are efficiently computed using integral images. Such features have been used in recent literature for a variety of tasks – indeed, variations appear to have been invented independently multiple times. Although integral channel features have proven effective, little effort has been devoted to analyzing or optimizing the features themselves. In this work we present a unified view of the relevant work in this area and perform a detailed experimental evaluation. We demonstrate that when designed properly, integral channel features not only outperform other features including histogram of oriented gradient (HOG), they also (1) naturally integrate heterogeneous sources of information, (2) have few parameters and are insensitive to exact parameter settings, (3) allow for more accurate spatial localization during detection, and (4) result in fast detectors when coupled with cascade classifiers.

1 Introduction

The performance of object detection systems is determined by two key factors: the learning algorithm and the feature representation. Considerable recent progress has been made both on learning [8, 10, 24, 26] and features design [5, 23, 28]. In this work we use a standard boosting approach [11] and instead focus our attention on the choice of features.

Our study is based on the following architecture: multiple registered image channels are computed using linear and non-linear transformations of the input image [12, 17]; next, features are extracted from each channel using sums over local rectangular regions. These local sums, and features computed using multiple such sums including Haar-like wavelets [27], their various generalizations [7], and even local histograms [20] are computed efficiently

© 2009. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.

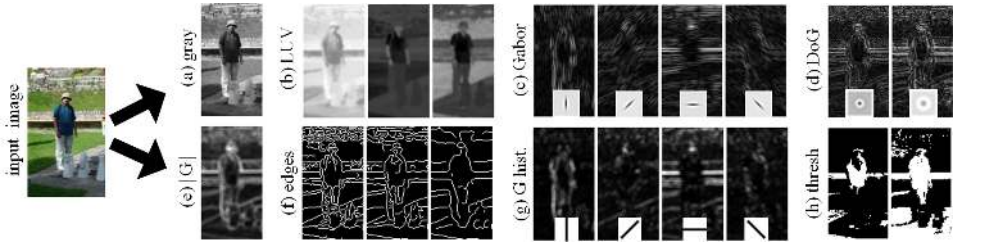

Figure 1: Multiple registered image channels are computed using various transformations of the input image; next, features such as local sums, histograms, and Haar wavelets are computed efficiently using integral images. Such features, which we refer to as *integral channel features*, naturally integrate heterogeneous sources of information, have few parameters, and result in fast, accurate detectors.

using integral images. We refer to such features as *integral channel features*, see Fig. 1. Integral channel features combine the richness and diversity of information from use of image channels with the computational efficiency of the Viola and Jones detection framework [27].

A number of papers have utilized variants of integral channel features, applications have included object recognition [16, 24], pedestrian detection [7, 31], edge detection [6], brain anatomical structure segmentation [25] and local region matching [2]. A unified overview of the feature representations in these works is given in Sec. 1.1.

Although integral channel features have proven effective, little effort has been devoted to analyzing or optimizing the features themselves. In many of the above mentioned works the focus was on the learning aspect [7, 8, 24] or novel applications [2, 6, 25] and it is difficult to decouple the performance gains due to the richer features from gains due to more powerful learning methods. In [16, 30, 31] the authors used integral channel features for computing histograms of oriented gradients; although these methods achieved good performance; they do not explore the full potential of the representation.

Furthermore, some of the integral channel features used in the literature have been computationally expensive, *e.g.* the channels in [6] took over 30s to compute for a 640×480 image. In this work we show how to compute effective channels that take about .05-.2s per 640×480 image depending on the options selected. For 320×240 images, the channels can be computed in real time at rates of 20-80 frames per second on a standard PC.

The INRIA pedestrian dataset [5] serves as our primary testbed. Pedestrian detection has generated significant interest in the past few years [9]; moreover, the Histogram of Oriented Gradient (HOG) descriptor [5] was designed specifically for the INRIA dataset. HOG has since been successfully adopted for numerous object detection tasks and is one of the most common features used in the PASCAL object challenges [19]. Therefore, not only is pedestrian detection interesting in and of itself, HOG's success in other domains serves as strong evidence that results obtained on pedestrians should generalize effectively.

Our detailed experimental exploration of integral channel features, along with a number of performance optimizations and use of complementary channel types, leads to large gains in performance. We show significantly improved results over previous applications of similar features to pedestrian detection [7, 8, 31]. In fact, full-image evaluation on the INRIA pedestrian dataset shows that learning using standard boosting coupled with our optimized integral channel features matches or outperforms all but one other method, including state of the art approaches obtained using HOG features with more sophisticated learning techniques. On the task of accurate localization in the INRIA dataset, the proposed method outperforms

state of the art by a large margin. Finally, we show results on the recently introduced Caltech Pedestrian Dataset [9], achieving a detection rate of almost 60% at 1 false positive per image compared to at most 50% detection rate for competing methods, including HOG.

The remainder of this paper is organized as follows. We begin with a review of related work below. In Sec. 2 we give a more detailed overview of integral channel features and we discuss implementation details in Sec. 3. We perform a detailed experimental evaluation in Sec. 4 and conclude in Sec. 5.

1.1 Related Work

The notion of channels can be traced back to the earliest days of computer vision. The Roberts Cross Edge Detector [22] employed two tiny (2x2) kernels representing orthogonal spatial derivative operators. The response of those filters, combined nonlinearly to obtain a rudimentary measure of edge strength and orientation, could be thought of as the ur-channels. Another early work was Fukushima's Neocognitron architecture which used layered channels of increasing discriminative power [12]. In the following decades, numerous extensions emerged. *E.g.*, the texture discrimination approach of Malik & Perona [17] employed dozens of channels computed via nonlinear combination of the responses of a bank of bandpass filters. Malik & Perona performed spatial integration via Gaussian smoothing; eventually statistics were pooled using histogram based representations [15, 21] still popular today.

Soon thereafter Viola and Jones proposed a boosted object detection approach with a front-end that eschewed computationally expensive bandpass kernels for efficient Haar-like wavelets implemented using integral images [27]. Nevertheless, with computing power increasing rapidly, computing the responses of a bank of bandpass filters became less of a bottleneck. The idea of using integral images on top of such channels to efficiently pool statistics with different regions of support naturally followed, giving rise to the representation we refer to as integral channel features.

The first application of integral images to multiple image channels that we are aware of appears in Tu's work on probabilistic boosting trees [24]. Tu computed the response of Gabor filters and Canny edges [4] at multiple scales and used Haar-like features on top of these channels in a boosted object detection framework. Later, Tu extended this approach to 3D MRI brain volume segmentation in [25], where each resulting channel and Haar feature was 3D. Likewise, Dollár *et al.* [6] used integral channel features to train a per-pixel edge detector; state of the art results were obtained that matched the performance of methods using more carefully tuned features. A large number of channels were used including gradients at various scales, Gabor filter responses, difference of offset Gaussian filters, etc. Similar channels were used for pedestrian detection [7] and learning patch descriptors [2].

In a different line of work, integral channel features have been used to compute histograms of oriented gradients efficiently. As described in [20], rectangular histograms can be computed efficiently by quantizing an image into multiple channels (details are given in Sec. 2). This key idea has been exploited at least three separate times in the literature [16, 30, 31]. Zhu *et al.* [31] used integral histograms to approximate HOG features for use in a cascade. Laptev [16] likewise computed gradient histograms using integral images, resulting in effective object detectors. In later work, [30] used a similar representation for cat detection, except features were individual histogram bins as opposed to entire histograms as in [16, 19]. Although details vary, none of these methods explored the richness of possible channel types to seamlessly integrate heterogeneous sources of information.

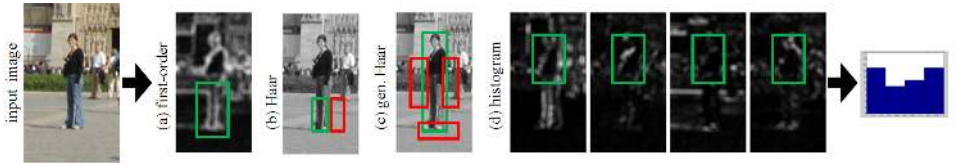

Figure 2: Examples of integral channel features: (a) A first-order feature is the sum of pixels in a rectangular region. (b) A Haar-like feature is a second-order feature approximating a local derivative [27]. (c) Generalized Haar features include more complex combinations of weighted rectangles [7]. (d) Histograms can be computed by evaluating local sums on quantized images [20] (see text for details).

2 Channel Types

Given an input image I, a corresponding channel is a registered map of the original image, where the output pixels are computed from corresponding patches of input pixels (thus preserving overall image layout). A trivial channel is simply $C = I$ for a grayscale image, likewise for a color image each color channel can serve as a channel. Other channels can be computed using linear or non-linear transformation of I, various choices are discussed below (see also Fig. 1). We use Ω to denote a channel generation function and write $C = \Omega(I)$. To allow for fast detection using sliding window detectors channels must be translationally invariant, that is given I and I' related by a translation, $C = \Omega(I)$ and $C' = \Omega(I')$ must be related by the same translation. This allows Ω to be evaluated once on the entire image rather than separately for each overlapping detection window.

We define a first-order channel feature f as a sum of pixels in a fixed rectangular region in a single channel, denoted by $f(C)$. Using an integral image [27] for each channel, $f(C)$ can be computed with three floating point operations, and as such first-order features are extremely fast to compute. We define higher-order channel features as any feature that can be computed using multiple first-order features (*e.g.*, the difference of two simple features in the same channel). See Fig. 2 for example channel features.

The above is a generalization of the features used in the detection framework of Viola and Jones (VJ) [27]. Specifically, VJ used $C = I$ with features resembling Haar Basis functions [18]. Using the terminology above, a Haar feature is a higher-order feature involving a sum of 2-4 rectangles arranged in patterns that compute first and second order image derivatives at multiple scales. This description is a bit oversimplified, as VJ actually used a two channel representation where $C_1(x,y) = I(x,y)$ and $C_2(x,y) = I(x,y)^2$. Together these 2 channels were used for variance normalization of detection windows leading to partial invariance to changing lighting conditions (see [27] for details).

A number of channel types for an example image are shown in Fig. 1, panels **a**-**h**. Note that all displayed channels are slightly smaller than the input image; it is of key importance to discard channel boundaries to avoid introducing undesirable artifacts. Below we present an overview of common channels used in the literature, all of which are translationally invariant.

Gray & Color: The simplest channel is simply a grayscale version of the image (panel **a**). Color channels can also be used, in panel **b** we display the three CIE-LUV color channels.

Linear Filters: A straightforward approach for generating channels that capture different aspects of an image is through use of linear filters. Panel **c** shows I convolved with 4 oriented Gabor filters [17]; each channel contains edge information in I at a different orientation. Convolving I with Difference of Gaussian (DoG) filters captures the 'texturedness' of the image at different scales (panel **d**). Convolution with a large bank of filters can be slow;

nevertheless, linear filters are a simple and effective method for generating diverse channels.

Nonlinear Transformations: There are countless translationally invariant non-linear image transformations, here we present a few illustrative cases. Gradient magnitude (panel **e**) captures unoriented edge strength while Canny edges (panel **f**) more explicitly compute edge information. Gradients for both can be computed at different scales by smoothing the input image; moreover, for color images a common trick is to compute the gradient on the 3 color channels separately and use the maximum response [5]. Panel **h** shows channels obtained by thresholding the input image with two different thresholds. Although this particular form of foreground segmentation is not robust, more sophisticated algorithms can be used.

Pointwise Transformations: Each pixel in a channel can be transformed by an arbitrary function as a post processing step. This can be used to overcome the limitation that a feature f must be a local *sum*. *E.g.*, taking the log, we can obtain local *products* since $\exp(\sum_i \log(x_i)) = \prod_i x_i$. Likewise, raising each element to the p-th power can be used to compute the generalized mean: $(\frac{1}{n}\sum_{i=1}^{n} x_i^p)^{1/p}$. Setting p to a large value approximates the *maximum* of x_i, which can be useful for accumulating values as argued in [12].

Integral Histogram: Porikli [20] presented an efficient method for computing histograms using integral images. Let Q be a quantized version of I with values in $\{1 \dots q\}$, and let $Q_i(x,y) = \mathbf{1}[Q(x,y) = i]$ for each $i \leq q$ ($\mathbf{1}$ is the indicator function). Counting the elements in a region of Q equal to i can be computed by summing over Q_i in the same region. Therefore, a histogram over a rectangular region can be efficiently computed given an integral image for each 'histogram' channel Q_i. Although histogram channels can be computed over any input [16], the most common use is a variant known as gradient histograms [16, 30, 31] described below. Note that a related representation known as 'value encoding' has been studied extensively in the neuroscience literature, *e.g.* see [3].

Gradient Histogram: A gradient histogram is a weighted histogram where bin index is determined by gradient angle and weight by gradient magnitude. In other words the channels are given by $Q_\theta(x,y) = G(x,y) \cdot \mathbf{1}[\Theta(x,y) = \theta]$, where $G(x,y)$ and $\Theta(x,y)$ are the gradient magnitude and quantized gradient angle, respectively, at $I(x,y)$. An example quantized to 4 orientations is shown in panel **g**. Gradients at different scales can again be obtained by smoothing I and an additional channel storing gradient magnitude (panel **e**) can be used to L_1 normalize the resulting histogram, thus allowing gradient histograms to approximate HOG features[31]. Note also the similarity between the channels in panels **c** and **g**. Although seemingly quite different, convolving I with oriented Gabor filters and quantizing gradient magnitude by gradient angle yields channels capturing qualitatively similar information.

We briefly discuss channel scale. We differentiate between **pre-smoothing** (smoothing the input image) and **post-smoothing** (smoothing the created channels). In the terminology of Gårding *et al.* [13], pre-smoothing determines the 'local scale' of the representation and serves to suppress fine scale structures as well as image noise. Pre-smoothing can have a significant impact on the information captured, *e.g.* pre-smoothing determines the scale of subsequently computed gradients, affecting computed Canny edges (panel **f**) or gradient histograms (panel **g**). Post-smoothing, on the other hand, helps determine the 'integration scale' over which information is pooled. However, the integration scale is also determined by the support of local sums computed over the channels, making post-smoothing less useful.

The channel representation has a number of appealing properties. Most channels defined above can be computed with a few lines of code given standard image processing tools; furthermore, many of the channels can be computed very efficiently (we discuss optimization techniques in Sec. 3). In Sec. 4 we show that most parameters used to compute the channels are not crucial (assuming they're consistent between training and testing).

3 Implementation Details

We evaluated combinations of three types of channels: gradient histograms, color (including grayscale, RGB, HSV and LUV), and gradient magnitude. These channels can be computed efficiently and capture diverse information. Below we discuss parameter settings, of which there are very few, followed by a short discussion of optimization techniques. We also discuss the boosting framework and full image detection.

Scale: Common parameters for the channels include local scale (pre-smoothing) and integration scale (post-smoothing). For computational efficiency we smooth using an approximation of (discretized) Gaussian filters based on binomial coefficients [14]. The width of the binomial filter is determined by a radius parameter r, which corresponds to a Gaussian filter with $\sigma \approx \sqrt{(2r+1)/4}$. At least r pixels of padding must be used to avoid introducing boundary effects. We use the same scale settings for each channel.

Parameters: In addition to scale, for gradient histograms the number of orientation bins must be specified. Gradient magnitude and the color channels have no additional parameters.

Optimization: The above channels can be computed with a few lines of code given standard image processing tools. Nevertheless, a few optimization techniques can dramatically decrease computation time. For LUV color channels, the bottleneck is cube root computation; we use a coarse approximation instead. A common trick for computing gradient histograms, used *e.g.* in [10], is to approximate the angle of a 2D vector by finding the maximum value of its dot product with vectors spaced uniformly along a unit circle. This can be done rapidly with binary search. We approximate gradient angle to 10 times the number of orientations and use linear interpolation to place each pixel in the appropriate channel.

Computation Time: The number of frames per second (fps) at which selected channels can be computed for 320×240 images, as tested on a standard PC, are: LUV color channels at 135 fps, gradient magnitude at 140 fps, and gradient histograms (with 6 bins) at 60 fps. Computing all 10 channel images can be performed at 40 fps, or 30-34 fps if pre-smoothing with $r \leq 8$ is used. Note that these rates do not include evaluation times for a classifier (although this is also quite fast).

Features: We generate a large pool of candidate features randomly rather than through careful design. A first-order feature is a sum over a rectangular region in a given channel; we generate candidates by randomly choosing both the channel index and the rectangle (enforcing a minimum area of 25 pixels). Higher-order features are randomly generated weighted sums of first-order features, each can span multiple channels. The number of possible first-order features in an $n \times n$ region is $O(n^4)$ and given higher-order feature this number grows rapidly; nevertheless, random sampling of features typically produces good results [7].

Boosting: Boosting offers a convenient, fast approach to learning given a large number of candidate features [11], and serves as the basis of the VJ object detection framework [27]. Here we use a variant known as a 'soft cascade' [29]. Instead of having multiple distinct cascade layers, a threshold is used after evaluation of every weak classifier. A single boosted classifier is trained on the entire data set and post-training a simple heuristic can be used to set the thresholds, resulting in a greatly simplified training procedure, see [29] for details.

We test with three variants of boosting: AdaBoost, RealBoost and LogitBoost [11]. Depth two decision trees are used as the weak classifier. Training a boosted classifier with 1000 weak classifiers given 20,000 training windows and 5,000 features takes about 5-10 minutes on a parallelized quad core machine (compared with 2 weeks for the original VJ classifier). The key optimization is that feature values are quantized and cached at the start of training so afterward training each weak classifier is fast.

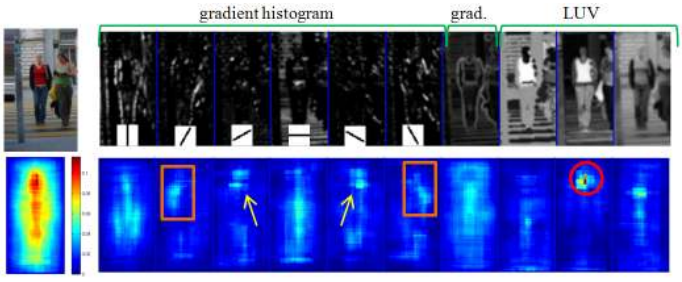

Figure 3: *Top*: Example image and computed channels. *Bottom*: Rough visualization of spatial support of trained classifier for all channels jointly (left) and separately for each channel type, obtained by averaging the rectangle masks of selected features. The orange rectangles highlight strong support for orientations slightly off from vertical for the region near the shoulders, likewise, the yellow arrows show support for orientations slightly off from horizontal for the side/top of the head. A surprising but intuitive result is a peak in the support of the 'U' color channel at the location of the head/face (red circle), apparently color is a strong, consistent cue for a person's face/head.

Full Image Detection: We use a sliding window over multiple scales. We use a step size of 4 pixels and a scale step of $2^{1/10}$. Non-maximal suppression (NMS) is used to suppress multiple nearby detections. Mean-shift NMS [5] is popular; however, it has multiple parameter settings that depend on other aspects of the detector such as step-size. Instead we use a simplified NMS procedure that suppresses the less confident of every pair of detections that overlap sufficiently according to the PASCAL criteria [19]. Although at worst the NMS procedure is $O(n^2)$ in the number of detections, in practice it can be made quite fast. Our NMS has a single parameter, the overlap threshold, which we set to .6. The overall system has a runtime of about 2s for multiscale detection in a 640×480 image, the fastest of all methods surveyed in [9].

4 Experiments

The majority of experiments are performed on the INRIA pedestrian dataset [5], we also show results on the recently introduced Caltech Pedestrian Dataset [9]. When comparing parameter settings we use ROC curves computed using *per-window* performance evaluation, thus avoiding NMS or other post processing. As is typical, we will frequently report detection rate at the reference point of 10^{-4} false positives per window (fppw). However, as shown in [9], the per-window measure is flawed and can fail to predict full image performance in certain cases. Therefore, we also evaluate our trained detectors on full images using the PASCAL criteria [19] in Sec. 4.2.

All parameters were kept constant except those explicitly being studied. By default, 8 channels were used, including gradient magnitude, grayscale, and 6 gradient histogram channels with no pre or post-smoothing. 5000 first-order features were randomly generated (no higher-order features were used by default). We trained AdaBoost on the resulting representation with 1000 depth-2 decision trees. Note that there are virtually no other parameters, either for the feature representation or learning algorithm. Negative windows were generated by training the default classifier with 5000 random negative windows, bootstrapping 5000 hard negative windows, and repeating the entire process a second time. The resulting 15,000 negatives were used for training in all subsequent experiments reported here.

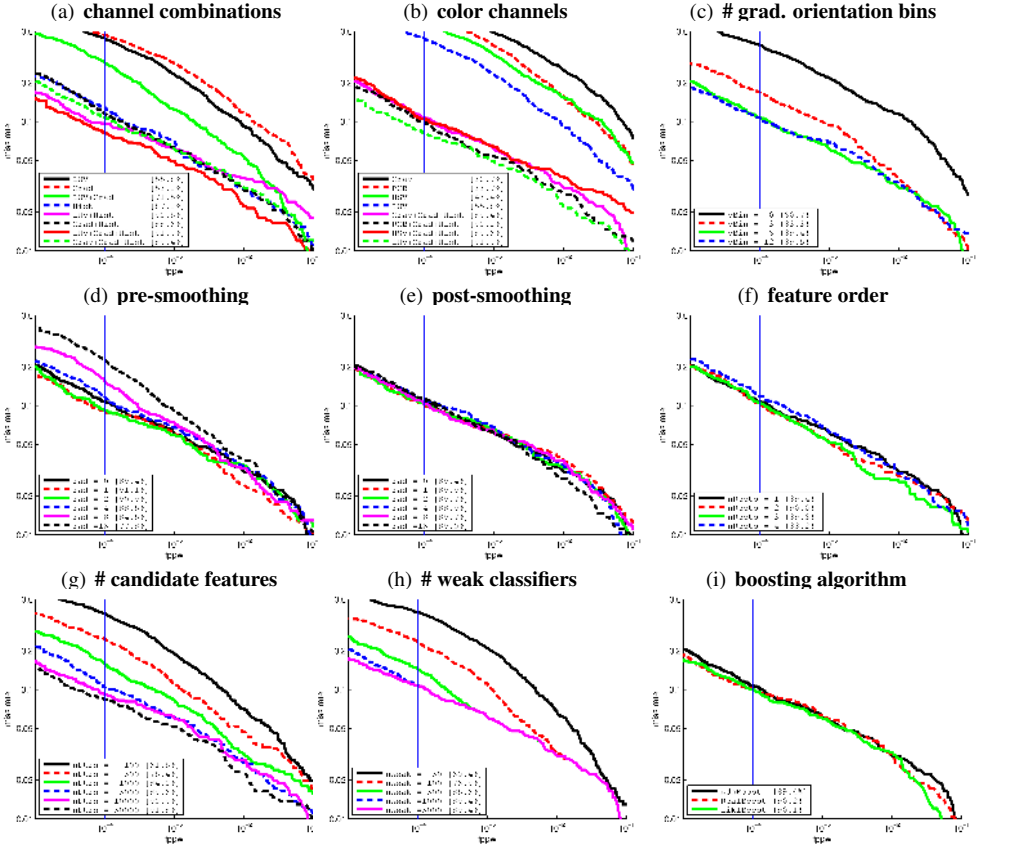

Figure 4: Evaluation of channel features, see text for details.

4.1 Channels and Features

Channels: In Fig. 4(a) and 4(b) we plot the performance of various channels types used alone or in conjunction. Not surprisingly, the most informative channels are the gradient histogram channels (Hist) achieving 87.2% detection performance at the reference point of 10^{-4} fppw. This is similar but slightly inferior to the 89% achieved by the HOG classifier with optimized parameter settings [5]. The number of bins for the histogram has little effect once at least 6 orientations were used (Fig. 4(c)). Combining Hist with gradient magnitude (Grad) and grayscale information (Gray) increases performance to 89.4% (the default configuration). However, Gray is not informative on its own, achieving only 30.7%. LUV stands out among the color spaces, achieving 55.8% on its own and 91.9% when combined with Grad and Hist. A visualization of the features used by LUV+Grad+Hist is shown in Fig. 3.

Scale: Pre-smoothing with a radius 1 binomial filter ($\sigma \approx .87$) improved performance from 89.4% to 91.1%, using a larger radius resulted in marked decreases in performance (Fig. 4(d)). Post-smoothing had little effect on performance (Fig. 4(e)).

Features: Using higher-order (randomly generated) features likewise had little effect on performance, raising detection rates only .6% over first-order features (Fig. 4(f)), so for efficiency first-order features are preferred. Using large numbers of candidate features can signficantly improve performance, with 30,000 features detection reaches 91.8% (Fig. 4(g)). The drawback of using more features is that training time increases proportionally.

Classifier: Increasing the number of weak classifiers improves performance up to about 1000 (Fig. 4(h)), although evaluating the classifier starts to take longer for larger values. Depth-2 trees gave the best performance, stumps (depth-1 trees) achieved only 83.2% detection (results not shown). The choice of boosting algorithm (Fig. 4(i)) plays almost no role in determining performance, demonstrating the stability of the overall framework.

4.2 Full Image Results

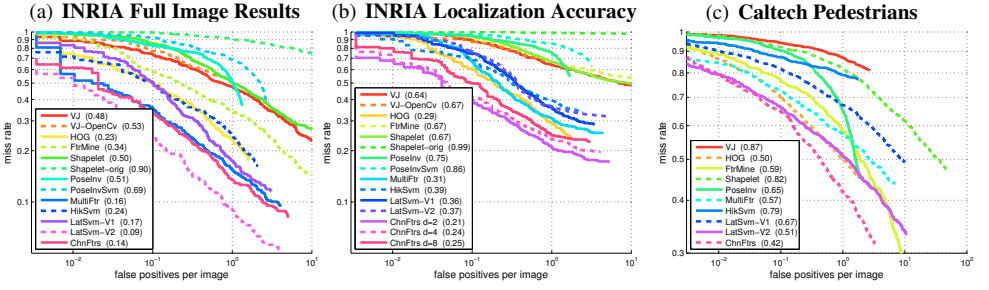

Figure 5: Full image results, see text for details.

For all subsequent experiments, we use a re-trained classifier. We changed 4 parameters from the default configuration above in order to maximize performance. We used: (1) LUV color channels, (2) pre-smoothing with $r = 1$, (3) 2000 weak classifiers, and (4) 30,000 candidate features for training. The resulting classifier achieves about 93% detection rate at 10^{-4} fppw, and almost 90% at 10^{-5} fppw. We refer to the resulting method as *ChnFtrs*.

INRIA results: Although typically results on the INRIA dataset are reported on cropped positives, the dataset also contains full images with the same pedestrians but within the original context. Dollár *et al*. [1, 9] evaluated 12 algorithms using these images and the PASCAL criteria. A number of algorithms that reported excellent per-window results (topping 95% detection) actually overfit to boundary effects and had fairly low per-image performance (see [9]). Per-image results of the 12 algorithms, along with the results of the proposed method (ChnFtrs) are shown in Fig. 5(a). ChnFtrs acheives 86% detection rate at 1 false positive per image (fppi), compared to a 77% detection rate for HOG. In fact, ChnFtrs matches or outperforms all but one other method ([10]) that has been evaluated on full images.

Localization Accuracy: One advantage of channel features is that a small step size d can be used for detection, while HOG windows have to be spaced at least $d \geq 8$ pixels apart (or recomputed at different offsets). Intuitively, using smaller d should result in better localization. To test this, we repeated the evaluation on the INRIA dataset, except we change the threshold in the PASCAL criteria to count a detection as correct if the area of intersection of the ground truth and detection is over 70% (instead of 50%). Results for ChnFtrs with d set to 2, 4 and 8 are shown in Fig. 5(a), note that all other methods are limited to $d \geq 8$. ChnFtrs with $d = 2$ achieves 79% detection, outperforming the next closest method by almost 10%.

Caltech Pedestrians: We conclude by showing results on Caltech Pedestrian Dataset [9] which contains almost half a million labeled bounding boxes and annotated occlusion information. Results for 50-pixel or taller, unoccluded or partially occluded pedestrians are shown in Fig. 5(c). Almost all classifiers shown were trained on the INRIA dataset, including ours, therefore, the results are a useful measure of the generalization power of the various methods. ChnFtrs significantly outperforms all other methods, achieving a detection rate of almost 60% at 1 fppi (compared to 50% for competing methods, including HOG).

5 Conclusion

The combination of diverse, informative channels along with the integral image trick for fast feature computation opened up the door to a much broader class of very effective features: the integral channel features examined closely in this work. As we have shown, integral channel features coupled with a standard boosting algorithm outperform existing features/methods for pedestrian detection. Eventually, we hope to extend these results to other object classes (specifically to the PASCAL VOC challenge). We would also like to explore additional families of channels.

Acknowledgments: S.B. work supported by NSF CAREER Grant #0448615 and ONR MURI Grant #N00014-08-1-0638.

References

[1] www.vision.caltech.edu/Image_Datasets/CaltechPedestrians/.

[2] B. Babenko, P. Dollár, and S. Belongie. Task specific local region matching. In *ICCV*, 2007.

[3] D.H. Ballard. Cortical connections and parallel processing: Structure and function. *Behavioral and Brain Sciences*, 9(1):67–90, 1986.

[4] J. F. Canny. A computational approach to edge detection. *PAMI*, Nov. 1986.

[5] N. Dalal and B. Triggs. Histogram of oriented gradient for human detection. In *CVPR*, 2005.

[6] P. Dollár, Z. Tu, and S. Belongie. Supervised learning of edges and object boundaries. In *CVPR*, 2006.

[7] P. Dollár, Z. Tu, H. Tao, and S. Belongie. Feature mining for image classification. In *CVPR*, 2007.

[8] P. Dollár, B. Babenko, S. Belongie, P. Perona, and Z. Tu. Multiple component learning for object detection. In *ECCV*, 2008.

[9] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *CVPR*, 2009.

[10] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008.

[11] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. Technical report, Stanford University, 1998.

[12] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36 (4):193–202, 1980.

[13] J. Gårding and T. Lindeberg. Direct computation of shape cues using scale-adapted spatial derivative operators. *IJCV*, 17(2):163–191, 1996.

[14] R. Haddad. A class of orthogonal nonrecursive binomial filters. *IEEE Transactions on Audio and Electroacoustics*, 19(3):296–304, Dec 1971.

[15] D. J. Heeger and J. R. Bergen. Pyramid-based texture analysis/synthesis. In *SIGGRAPH*, 1995.

[16] I. Laptev. Improvements of object detection using boosted histograms. In *BMVC*, 2006.

[17] J. Malik and P. Perona. Preattentive texture discrimination with early vision mechanisms. *Journal of the Optical Society of America A*, 7:923–932, May 1990.

[18] C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *ICCV*, 1998.

[19] J. Ponce, T.L. Berg, M. Everingham, D. Forsyth, M. Hebert, S. Lazebnik, M. Marszałek, C. Schmid, C. Russell, A. Torralba, C. Williams, J. Zhang, and A. Zisserman. Dataset issues in object rec. In *Towards Category-Level Object Rec.*, pages 29–48. Springer, 2006.

[20] F. Porikli. Integral histogram: A fast way to extract histograms in cartesian spaces. In *CVPR*, 2005.

[21] J. Puzicha, T. Hofmann, and J. Buhmann. Non-parametric similarity measures for unsupervised texture segmentation and image retrieval. In *CVPR*, 1997.

[22] L. G. Roberts. Machine perception of 3d solids. In J. T. Tippett et al., editor, *Optical and Electro-optical Information Processing*, pages 159–197. MIT Press, 1965.

[23] E. Seemann, B. Leibe, K. Mikolajczyk, and B. Schiele. An evaluation of local shape-based features for pedestrian detection. In *BMVC*, 2005.

[24] Z. Tu. Probabilistic boosting-tree: Learning discriminative models for classification, recognition, and clustering. In *ICCV*, 2005.

[25] Z. Tu, K.L. Narr, P. Dollár, I. Dinov, P.M. Thompson, and A.W. Toga. Brain anatomical structure segmentation by hybrid discriminative/generative models. *Transactions on Medical Imaging*, 27(4):495–508, April 2008.

[26] O. Tuzel, F. Porikli, and P. Meer. Human detection via classification on riemannian manifolds. In *CVPR*, 2007.

[27] P. Viola and M. Jones. Robust real-time object detection. *IJCV*, 57(2):137–154, 2004.

[28] Christian Wojek and Bernt Schiele. A performance evaluation of single and multi-feature people detection. In *DAGM*, 2008.

[29] Cha Zhang and Paul Viola. Multiple-instance pruning for learning efficient cascade detectors. In *NIPS*, 2007.

[30] W. Zhang, J. Sun, and X. Tang. Cat head detection - how to effectively exploit shape and texture features. In *ECCV*, 2008.

[31] Q. Zhu, M.-C. Yeh, K.-T. Cheng, and S. Fast human detection using a cascade of histograms of oriented gradients. In *CVPR*, 2006.