A FREQUENCY-DOMAIN IMPLEMENTATION OF A SLIDING-WINDOW TRAFFIC SIGN DETECTOR FOR LARGE SCALE PANORAMIC DATASETS

I.M. Creusen[a,b,*], L. Hazelhoff[a,b], P.H.N. de With[a,b]

Cyclomedia Technology, Zaltbommel, The Netherlands
Eindhoven University of Technology, Eindhoven, The Netherlands
(i.m.creusen, l.hazelhoff, p.h.n.de.with)@tue.nl

KEY WORDS: Traffic sign detection, Sliding window, Object Detection, Frequency Domain

ABSTRACT:

In large-scale automatic traffic sign surveying systems, the primary computational effort is concentrated at the traffic sign detection stage. This paper focuses on reducing the computational load of particularly the sliding window object detection algorithm which is employed for traffic sign detection. Sliding-window object detectors often use a linear SVM to classify the features in a window. In this case, the classification can be seen as a convolution of the feature maps with the SVM kernel. It is well known that convolution can be efficiently implemented in the frequency domain, for kernels larger than a certain size. We show that by careful reordering of sliding-window operations, most of the frequency-domain transformations can be eliminated, leading to a substantial increase in efficiency. Additionally, we suggest to use the overlap-add method to keep the memory use within reasonable bounds. This allows us to keep all the transformed kernels in memory, thereby eliminating even more domain transformations, and allows all scales in a multiscale pyramid to be processed using the same set of transformed kernels. For a typical sliding-window implementation, we have found that the detector execution performance improves with a factor of 5.3. As a bonus, many of the detector improvements from literature, e.g. chi-squared kernel approximations, sub-class splitting algorithms etc., can be more easily applied at a lower performance penalty because of an improved scalability.

1 INTRODUCTION

The availability of large-scale collections of street-level panoramic photographs has led to new possibilities for surveying real-world objects. Several mobile mapping companies densely record panoramic photographs from driving vehicles, of which a few companies do this annually. This development is especially relevant for governmental institutions that are responsible for maintaining the roads and related infrastructure, such as traffic signs, street lights or traffic lights. The ability to view up-to-date photographs on a computer at the working place thereby avoiding multiple site visits and enabling a substantial cost reduction. It has become popular to regularly survey road-side objects such as traffic signs, because they can be damaged by collisions, deteriorated in color due to aging, be vandalised, stolen or become occluded by greenery. Additionally, situations occasionally arise where traffic signs keep being added over the years, leading to confusing and potentially dangerous traffic situations. Surveying traffic signs is a key example of a process that can become vastly more efficient by using street-level panoramic photographs combined with (partially) automated computer vision techniques. A system to automate this task needs to perform three functions: sign localization (in images with signs), sign classification (for traffic sign identity) and 3D positioning (to find 3D coordinates of the sign). As millions of panoramic photographs are involved in the search, the computational load of such systems is an important concern. In most systems, the computational load is highest at the sign localization stage, mainly because for every sign multiple pictures have to be analyzed. In this paper we have found a new method to reduce computional complexity of this large search, while preserving the quality of the sign localization with existing techniques.

Several complete traffic sign surveying systems that consider the entire pipeline from detection to classification and 3D positioning have been reported in literature. For example, the work of (Timofte et al., 2009) has many similarities to our proposed application, although the input consists of video rather than panoramic images. The use of video simplifies the detection problem due to increased redundancy in the data. Another example is the system described in (Hazelhoff et al., 2012) that deals with the previously described large-scale panoramic datasets taken at a 5-m intervals. For each of the individual stages of the pipeline there is much literature available, but since our focus is on sign localization, we will not describe the other stages. There is considerable prior work on traffic sign detection. A large portion of this research is focused on other applications, such as driver assistance systems or autonomous vehicles. These alternative applications have different constraints from our surveying application. The alternative systems often work in real-time on video streams, while supporting a small subset of traffic signs. In contrast, our application does not need to work in real-time, deals with high-resolution panoramic images taken every 5 meters and supports hundreds of traffic sign types. The real-time constraints of the systems related to vehicles often lead to algorithms based on the Viola and Jones detector, or highly specific algorithms created for this purpose. Some example systems can be found in (de la Escalera et al., 1997) and (Bahlmann et al., 2005). Another approach is taken in (Creusen et al., 2010), where a generic object detection algorithm is employed for traffic sign detection and extended to include color information. This approach is adopted in this paper, as it has some important advantages. The detector can easily be extended with support for additional traffic signs by providing new training samples. Additionally, the same algorithm can be used for detecting traffic signs as well as persons, license plates, street lights and other kinds of objects.

Many successful object detection algorithms are based on the sliding-window principle. One of the earliest examples of a detector of this type is the well-known Viola-Jones object detector (Viola and Jones, 2001). Features extracted from a rectangular window are used to classify the presence of an object in a region. The window is then slid across the image and each window is classified. Often, this is repeated on multiple scales to

*Corresponding author.

enable multi-scale detection. The computational load of object detection algorithms is usually concentrated in two stages: (1) feature extraction and (2) the sliding-window stage. In practice, the classification algorithm is frequently chosen to be a Support Vector Machine (SVM) with a linear kernel, because the enormous amount of windows does not allow for very complex classification algorithms, although other classifiers such as Adaboost can also be used. In this paper, we focus exclusively on optimization of the sliding-window stage of a detector using a linear classifier, such as for example the popular Histogram of Oriented Gradients approach (Dalal and Triggs, 2005).

Recently, many proposals have been published that improve the effectiveness of HOG-based detectors, either by extracting more features from the image, or by using multiple SVM kernels for a single object class. Both of these categories of improvements increase the computational load on the sliding window part of the object detector. An example is the proposal (Vedaldi and Zisserman, 2012) of a set of feature map transformations to approximate a Chi-Square kernel SVM, based on a linear kernel with an extended set of features, leading to a significant increase of the number of feature maps. Wijnhoven and De With (Wijnhoven and With, 2011) propose a method to improve performance by splitting classes into multiple sub-classes, which leads to improved performance at the cost of an increase in the number of classes. When decomposing an object into parts, i.e. a part-based model, such as in (Crandall et al., 2005), the number of classes also increases significantly. Another proposal (Creusen et al., 2012) to extract features from multiple color channels, also leads to a multiplication of the number of feature maps. This is just a selection, this list is not exhaustive. In all of the above proposals, an efficient sliding-window implementation is essential. These papers illustrate that an increase in computational efficiency of the sliding window stage can be used for two purposes: (1) improving the detection performance by applying one of the previously mentioned proposals or (2) to improve the scalability of the system towards large scale panoramic datasets.

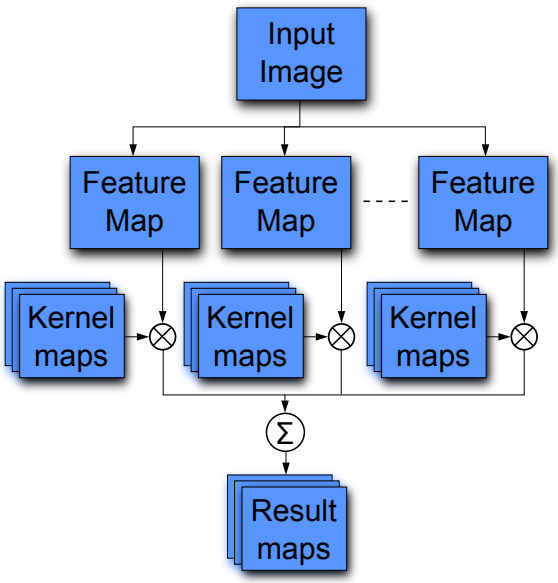


Figure 1: Overview of computations in a regular sliding-window process. The convolution is performed once for each class per feature map, and corresponding convolution results are summed up per class.

Traditionally, sliding window detector are implemented in the pixel domain. For each position of the sliding window, a feature vector is extracted which is then classified by the classification algorithm. In case of a linear classifier, the classification can be considered as a series of convolutions of feature maps with their corresponding kernel maps. We show that by performing this convolution in the frequency domain and changing the order of operations, the processing time can be significantly reduced. Furthermore, we propose a way to limit the memory usage of the kernel maps, allowing them to be kept in memory, to avoid recomputation. Additionally, our approach allows the same set of transformed kernels to be used for all scales in a multiscale pyramid. It will be shown that the performance is much less sensitive to parameters like kernel size, number of classes and number of feature maps.

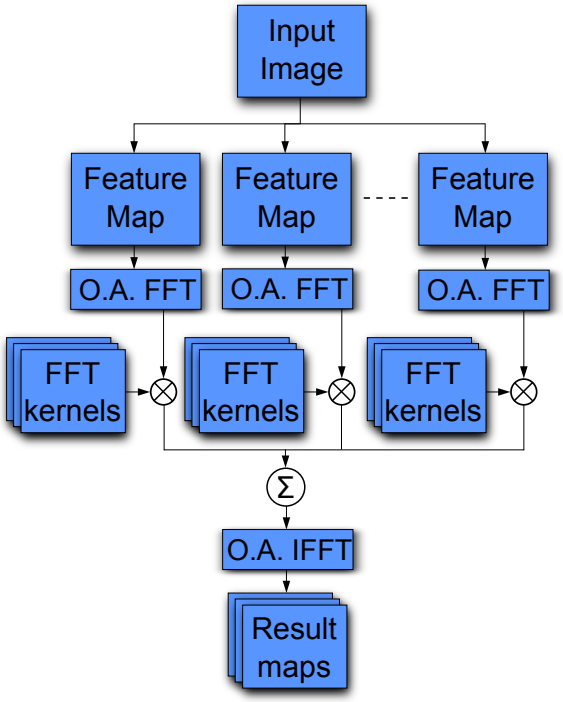


Figure 2: Schematic view of our proposed optimizations. Note that O.A. stands for Overlap-add. The FFT kernel maps can be stored in memory and do not need to be recomputed.

The remainder of this paper is structured as follows. Section 2 describes the sliding-window stage of a typical detector with a linear SVM. Our optimizations to the standard implementation are described in Section 3. In Section 4, the benchmark results of our implementation are compared to the standard implementation. Finally, Section 5 contains concluding remarks.

2 SYSTEM OVERVIEW

Typically, an object detection algorithm extracts a number of feature maps from the image. In the standard HOG algorithm, these features correspond to image gradients in a particular direction, contained within a group of pixels. However, these feature maps can contain other information such as color, texture or shape information, enabling the use of a mix of different types of feature maps. When training the SVM, information extracted from all these feature maps is concatenated into one large feature vector. After the SVM training process, the resulting coefficient vector can be split into separate kernel maps that correspond to each of the feature maps. In the case of a linear classifier, the sliding-window stage is equivalent to a *convolution* of the feature maps with their corresponding SVM kernel. Each class has a corresponding kernel for each of the feature maps, and the results of

the convolutions should be accumulated to obtain the resulting representation. The process is visualized in Figure 1.

The feature extraction stage is typically relatively slow, but is only performed once per scale. The convolution stage is faster than feature extraction, but still quite slow and is performed multiple times. It is performed for every object class on every feature map, and therefore it can easily become a computational bottleneck. The convolution operation in the pixel domain scales quadratically with respect to kernel size. The summation is also performed many times, but this is a much faster operation than the convolution.

3 EXECUTION OPTIMIZATIONS

It is well known that convolutions can be performed efficiently in the frequency domain for kernels larger than a certain size. The advantage of frequency-domain convolutions is that they are completely insensitive to the size of the kernel. However, the FFT and its inverse transformation are still expensive operations. If a naive frequency-domain approach is used, the gain in performance would be small, although this depends on the kernel size. The real advantage of using a frequency-domain approach is that this allows us to perform the final summation of the convolutions in the frequency domain, as it is a linear operation. This reduces the number of inverse FFT operations by a large amount, down to one per class per image, instead of once per feature map per class per image. This approach already achieves a significant reduction in the number of computations, but can be improved even further.

A second issue is the efficient handling of kernel maps. Usually, when performing a frequency-domain convolution, the kernel is zero-padded to the same size as the feature map, and then transformed to the frequency domain. However, this means that the kernel maps consume a large amount of memory, and it becomes impractical to keep all the transformed kernels in memory. As a solution, we propose to use the overlap-add method, which is a common method of splitting a convolution of a large image by a small kernel into many smaller convolutions, see Figure 3. The kernels now only need to be zero-padded up to the size of the overlap-add region, and this makes it possible for them to be stored transformed in memory continuously during the computation. For example, an overlap-add region of 100×100 pixels means that each transformed kernel consumes about 78.1 kB of memory (stored as complex single-precision floating-point values), meaning that thousands of kernels can be kept in memory in a modern PC. This has the additional benefit that the memory usage of the sliding-window process is now completely decoupled from the image resolution, since tile-by-tile image processing is performed. This aspect is also beneficial for multi-scale processing, because all scales can be processed with the same set of transformed kernels.

There are only a few frequency-domain transformations that remain essential in normal operation of the system, and they occur on small tiles rather than on full-resolution feature maps. The feature maps still have to be transformed to the frequency domain, and the result maps require inverse transformation. All other operations are directly performed in the frequency domain as complex multiplications (for the convolutions) and additions. For a typical number of 48 feature maps and 20 object classes, the number of FFT and IFFT transformations has been reduced from 1028 in the naive FFT implementation to 68 with our proposed optimizations. This is an impressive reduction with a factor 15, although the overlap-add process does add some overhead. The final optimized process is visualized in Figure 2.

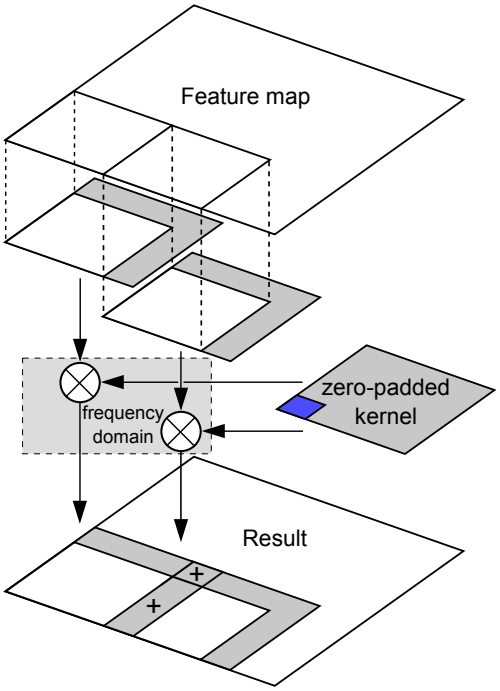


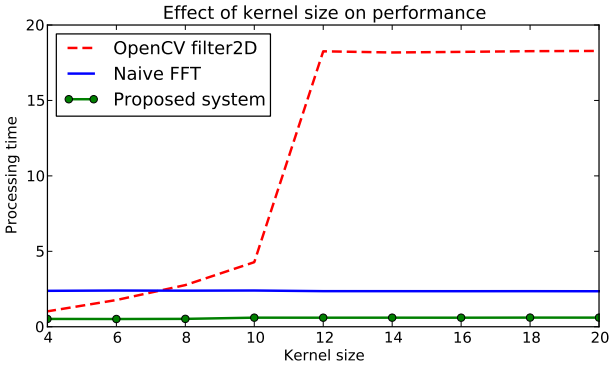
Figure 3: Overview of the overlap-add process for a single kernel. In our proposed system multiple kernels are used and their results are accumulated in the frequency domain.

4 EXPERIMENTAL RESULTS

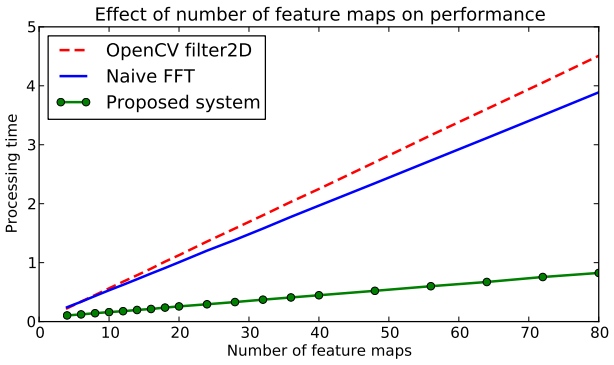
To evaluate the performance improvement of our proposed technique for performing the sliding-window stage in the frequency-domain, we compare our system to a normal pixel-domain implementation, for which the filter2D function from OpenCV (Bradski, 2000) is used. Please note that the filter2D function automatically switches to a frequency-domain implementation for the kernels in our experiment larger than 10×10 pixels. Additionally, we compare the performance to a naive frequency-domain implementation, in which for each convolution two transformations and one inverse transformation is performed. For the frequency transforms, the FFTW library (Frigo and Johnson, 2005) is used, and the complex multiplication and additions are implemented using SSE-optimized functions. All experimental execution times reported are the average over 50 executions, and the indicated measured time consists exclusively of the sliding-window stage of the detector, so the time for feature extraction is ignored. The image is processed at a single scale.

The results are shown in Figure 4. Figure 4(a) shows that the FFT-based implementations are not significantly affected by the kernel size. The processing time of our proposed system increases only slightly because the overlap-add method becomes less efficient for larger kernels (more padding must be used). In Figure 4(b) we show the impact of the number of feature maps on the performance of the sliding-window stage. As expected, the processing time increases linearly with the number of feature maps for all apporaches, however we can observe that the processing time of our proposed method increases at a much slower pace compared with the regular and naive FFT implementations. The same can be concluded about the number of classes in Figure 4(c), where the performance of our proposed system scales much better than the other implementation methods.

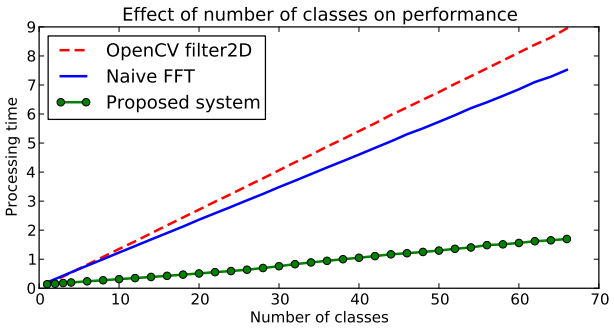
For a typical configuration of 20 classes, 48 feature maps and 8×8 kernels, the processing time is reduced by a significant factor of 5.3. The previous benchmark results are for only one part



(a) Effect of kernel size on performance. Square kernels are used with sizes equal to the number on the x-axis. The number of feature maps is set to 48, and the number of classes to 20. Note that the filter2D implementation automatically switches to an FFT-based implementation for kernels larger than 10×10 pixels.



(b) Effect of the number of feature maps on performance. The kernel is set to 8×8 pixels, and the number of classes to 20.



(c) Effect of the number of classes on performance. The kernel size is set to 8×8, and the number of feature maps to 48.

Figure 4: Experimental results of our benchmarks. Experiments are performed on a Windows-7 PC with a CPU Core i7 920, single-threaded. Processing times are in seconds, for feature maps of 600×300 pixels.

of the detection algorithm, the sliding window stage and only on a single processing scale. In order to validate how big the performance gain is in a more realistic setting, we have also performed a benchmark of the whole detector, including reading 20 MPixel images from the network, preprocessing, feature extraction on 45 image scales, mean shift clustering of results and writing them to disk. We have found that in this scenario, the performance gain is obviously less large than when measuring the sliding window stage in isolation, but still amounts to a reduction of 28% in the total time. We would like to emphasize that the detection results are identical with our proposed sliding window system compared with the traditional pixel domain implementation, therefore we have not included performance figures.

5 CONCLUSIONS

We have proposed an optimized implementation of the sliding-window stage of an object detection algorithm. This implementation significantly improves the scalability of the object detection algorithm with respect to the number of classes, the number of feature maps and the kernel size. This proposal can be used to achieve two goals. First, it gives an increase in computational performance leading to faster processing times, which makes applications on large-scale datasets such as country-wide datasets of panoramic images more feasible. Second, for the same computation time, our technique enables improved detection performance. The ways for implementing this approach have been published already in literature, such as by increasing the number of feature maps or by expanding the number of classes. By performing the convolutions in the frequency domain, and switching the order of operations, multiple transformations can be eliminated. Finally, when using the overlap-add method to split the large convolutions in many smaller convolutions, the memory usage of the kernel maps is significantly decreased, so that they can be kept in memory and recomputations are avoided. The only remaining transformations in the system are the forward transformations of the feature maps and the inverse transformations of the result maps, which is a factor 15 less than a naive FFT implementation, for the presented case of 20 classes and 48 feature maps.

Our benchmarks show that for a typical configuration of 20 classes and 48 feature maps, the processing time is reduced by a factor of 5.3. Further experiments show that the system is relatively insensitive to variations in kernel size, number of feature maps and number of classes. This means that many of the detector improvements from literature, such as color planes, chi-squared kernel approximations, sub-class splitting algorithms or shape models, now carry a significantly smaller performance penalty.

REFERENCES

Bahlmann, C., Zhu, Y. and Ramesh, V., 2005. A system for traffic sign detection, tracking, and recognition using color, shape and motion information. In: Proc. of the IEEE Symposium on Intelligent Vehicles, pp. 255–260.

Bradski, G., 2000. The opencv library. Dr. Dobb's Journal of Software Tools.

Crandall, D., Felzenszwalb, P. and Huttenlocher, D., 2005. Spatial priors for part-based recognition using statistical models. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Vol. 1, pp. 10–17.

Creusen, I., Hazelhoff, L. and de With, P. H. N., 2012. Color transformation for improved traffic sign detection. In: Proc. of the IEEE International Conference on Image Processing (ICIP), Orlando, USA, pp. 457–460.

Creusen, I., Wijnhoven, R., Herbschleb, E. and de With, P., 2010. Color exploitation in hog-based traffic sign detection. In: Proc. of the IEEE International Conference on Image Processing (ICIP), Hong Kong, China, pp. 2669–2672.

Dalal, N. and Triggs, B., 2005. Histogram of oriented gradients for human detection. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Vol. 1, pp. 886–893.

de la Escalera, A., Moreno, L., Salichs, M. and Armingol, J., 1997. Road traffic sign detection and classification. IEEE Transactions on Industrial Electronics 44(6), pp. 848–859.

Frigo, M. and Johnson, S. G., 2005. The design and implementation of fftw3. Proceedings of the IEEE 93(2), pp. 216–231.

Hazelhoff, L., Creusen, I. M. and de With, P. H. N., 2012. Robust detection, classification and positioning of traffic signs from street-level panoramic images for inventory purposes. In: Proc. of the Workshop on Applications of Computer Vision (WACV), Breckenridge, USA, pp. 313–320.

Timofte, R., Zimmermann, K. and Gool, L. V., 2009. Multi-view traffic sign detection, recognition, and 3d localisation. In: Proc. of the Workshop on Applications of Computer Vision (WACV), pp. 1–8.

Vedaldi, A. and Zisserman, A., 2012. Efficient additive kernels via explicit feature maps. IEEE Transactions on Pattern Analysis and Machine Intelligence 34(3), pp. 480–492.

Viola, P. and Jones, M., 2001. Rapid object detection using a boosted cascade of simple features. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Vol. 1, pp. 511–518.

Wijnhoven, R. G. J. and With, P. H. N. D., 2011. Unsupervised sub-categorization for object detection: Finding cars from a driving vehicle. In: Proc. IEEE International Conference on Computer Vision (ICCV), pp. 2077–2083.