A Real-time Visual Attention System Using Integral Images

Simone Frintrop[1], Maria Klodt[2], and Erich Rome[2]

[1] Institute of Computer Science III, Rheinische Friedrich-Wilhems-Universität, 53111 Bonn, Germany
[2]Fraunhofer Institut Intelligente Analyse und Informationssysteme (IAIS), Schloss Birlinghoven, 53754 Sankt Augustin, Germany

Abstract. Systems which simulate human visual attention are suited to quickly find regions of interest in images and are an interesting preprocessing method for a variety of applications. However, the scale-invariant computation of features in several feature dimensions is still too time consuming to be applied to video streams at frame rate which is necessary for many practical applications. As a consequence, current implementations of attention systems often make compromises between the accuracy and speed of computing a focus of attention in order to reduce the computation time. In this paper, we present a method for achieving fast, real-time capable system performance with high accuracy. The method involves smart feature computation techniques based on integral images. An experimental validation of the speed gain of our attention system VOCUS is provided, too. The real-time capability of the optimized VOCUS system has already been demonstrated in robotic applications.

1 Introduction

Computational attention systems have gained a lot of interest during the last years [14, 7, 10, 2]. Similar to the human visual system, they detect regions of interest in images: by "directing attention" to these regions, they restrict further processing to sub-regions of the image. Such guiding mechanisms are urgently needed, since the amount of information available in an image is so large that even the most performant computer cannot carry out exhaustive search on the data. This is eloquently explained in the work by Tsotsos [12, 13]: He proves that *unbounded visual search* (no target is given or it cannot be used to optimize search) is *NP-complete*[1].

Computational attention systems compute different features like intensity, color, and orientations in parallel to detect feature dependent saliencies. The saliency of a region is high when it has both a strong contrast to the environment and a high uniqueness, that means, the region differs considerably from the rest

[1] Problems that are *NP-complete* belong to the hardest ones in computer science. No polynomial algorithm is known for this class of problems and they are expected to require exponential time in the worst case [5].

of the image. Imagine one red and many blue balls on grass; the red ball has a higher saliency than the blue ones, because the uniqueness value is higher. Finally, the feature dependent saliencies are fused in a single saliency map which highlights regions of interest.

Compared to common interest region detectors like SIFT features [8] or Harris corners [9], these biologically motivated attention systems have several advantages. First, they are not restricted to a single feature dimension like for example corner or line features but consider several dimensions in parallel. This makes them more general and applicable to different environments. The weighting by uniqueness provides an intelligent way to fuse the feature dimensions by highlighting features which are most discriminative to the surrounding. Another advantage is that attention systems detect a limited amount of features in an image, usually between 5 and 20. This is in contrast to for example corner detectors which find thousands of features in texture-rich images. In applications like structure from motion detection and visual SLAM (Simultaneous Localization and Mapping) it is important to restrict processing to a limited amount of features to meet real-time constraints. Finally, the discriminativity of the attentional features is an important advantage for feature matching: whereas similar corners in texture-rich environments often can not be distinguished, the attention system picks especially those regions which are unique and thus better suited for matching.

Although very useful for the detection of regions of interest, current attention systems applied to video streams usually do not fulfill real-time constraints. The reason are time consuming feature computations on several scales and feature dimensions. Especially for applications on mobile robots, real-time performance is essential. Furthermore, when used as a pre-processing method for object recognition, it is consequently essential to be faster than the recognizer. Especially for very fast classifiers, this can be a challenging task.

However, there have been some approaches to speed up attention systems. Some groups utilized the parallel structure of the models: in [6], several CPU's share the computation of the feature maps and in [10], a dedicated hardware parallelizes the computations. On the other hand, the dependence on a special hardware makes a system less flexible. Especially for research purposes, it is important that a system can be implemented easily, adapted flexibly, and distributed to different platforms without additional hardware costs. A more flexible software solution to speed up the computations is realized in one of the best-known attention systems available: the Neuromorphic Vision Toolkit (NVT) of the group around Itti [7]. The computationally most expensive part, the feature computations, is realized with an approximation which trades off speed versus accuracy: instead of applying linear filters to each pyramid level, center-surround features are computed as differences between different pyramid levels (cf. sec. 3). In earlier versions of our attention system VOCUS, we assigned priority to accuracy, accepting a slower computation time [2], but for current applications in robotics we needed real-time performance.

In this paper, we present a method for obtaining a fast, real-time capable attention system with high accuracy. This is done by performing smart feature computation techniques based on integral images. This technique, originally coming from the graphics community [1], has entered the computer vision area only recently [15]. Surprisingly, it has until now not yet entered the field of computational attention. The feature computations based on integral images are easily implementable and yield the same results as a filter-based approach as long as rectangular filters are considered. By applying these techniques to VOCUS, the speed of the system increases by a factor of 10. With a processing time of several milliseconds, VOCUS is capable of real-time performance. This was demonstrated in a visual SLAM scenario for mobile robotics [4].

We start by introducing our visual attention system VOCUS (sec. 2). Next follows the main part of the paper: we describe the integral images and how they are integrated into VOCUS to speed up the processing (sec. 3). In sec. 4 we present the results, and, finally, sec. 5 concludes the paper.

2 The Visual Attention System VOCUS

The visual attention system VOCUS (Visual Object detection with a CompUtational attention System) which used in this work is motivated from the human visual system and detects salient regions in images [2, 3]. VOCUS differs from most other attention systems by the ability to consider target knowledge (top-down information) to enable goal-directed search. It consists of a bottom-up and a top-down part; global saliency is determined from both cues (cf. Fig. 1). However, the optimizations proposed in this paper work on the bottom-up part and are therefore applicable to most current attention systems.

The structure of VOCUS' bottom-up part is based on one of the standard attention systems, the Neuromorphic Vision Toolkit (NVT) by Itti et al. [7] with some optimizations [2]. It detects salient image regions by computing image contrasts and uniqueness of a feature. For each of the features intensity, color, and orientation, we first compute an *image pyramid* with 5 layers. From the coarsest three levels of the pyramids, *scale maps* are computed. These represent saliencies on different scales for different feature types (like red, horizontal, etc.). The scale maps are fused into *feature maps* representing different feature types and these again are combined to *conspicuity maps*, one for each feature. Finally, the conspicuity maps are fused to a single *saliency map*, with the degree of brightness proportional to the degree of saliency. Some of the computed maps are displayed in Fig. 2.

The intensity and the color scale maps are created by *center-surround mechanisms.* These are inspired by the ganglion cells in the visual receptive fields of the human visual system, which respond to intensity contrasts between a center region and its surround. The cells are divided into two types: on-center cells respond excitatorily to light at the center and inhibitorily to light at the surround, whereas off-center cells respond inhibitorily to light at the center and excitatorily to light at the surround [11]. Accordingly, we determine two feature

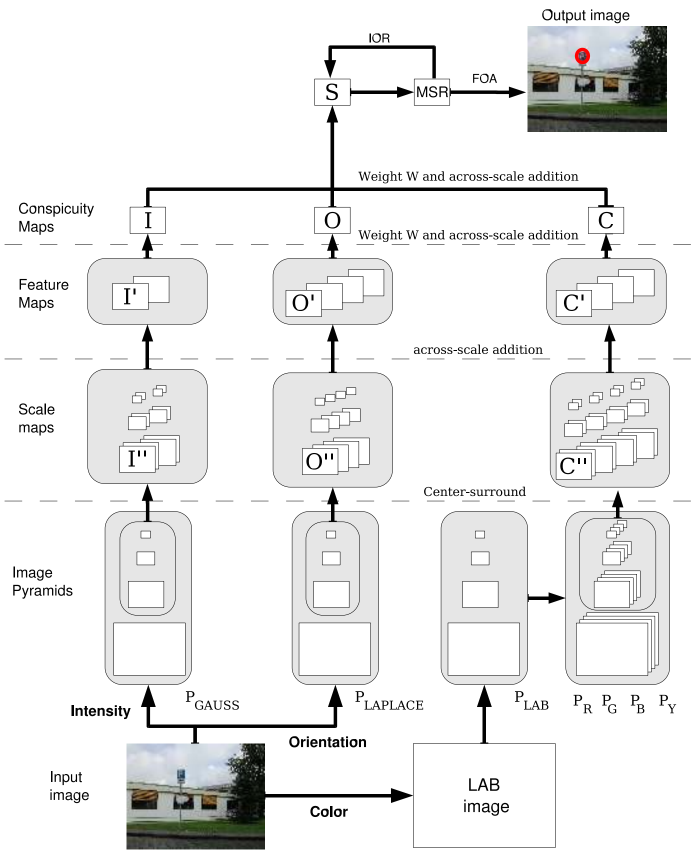


Fig. 1. The visual attention system VOCUS

types for intensity: the on-center difference responding strongly to bright regions on a dark background, and the off-center difference responding strongly to dark regions on a bright background. For two modes $i \,\epsilon\, \{(\text{on}), (\text{off})\}$, three levels s, and two surround sizes σ, this yields 12 intensity scale maps $I''_{i,s,\sigma}$. Similarly, we compute 24 color scale maps $C''_{\gamma,s,\sigma}$, with γ denoting the colors red, green, blue, and yellow. The 12 orientation scale maps $O''_{\theta,s}$, with $\theta \,\epsilon\, \{0°, 45°, 90°, 135°\}$, are determined with Gabor filters (details on the feature computations in [2, 3]).

The center-surround mechanisms are the most important part in this paper. They are most time consuming and therefore we apply the integral images here to achieve a speed-up. The details are explained in the next section.

Next, the scale maps are summed up for each feature type using *across-scale addition*: first, all maps are resized to scale s_2 whereby resizing scale s_i to

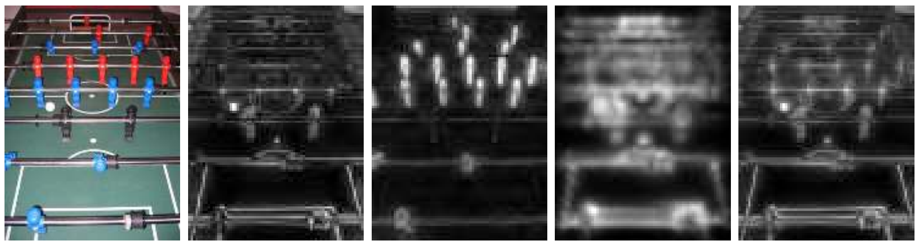


Fig. 2. Original image **(a)**, conspicuity maps for intensity **(b)**, color **(c)**, and orientation **(d)**, and the saliency map S_{bu} **(e)**

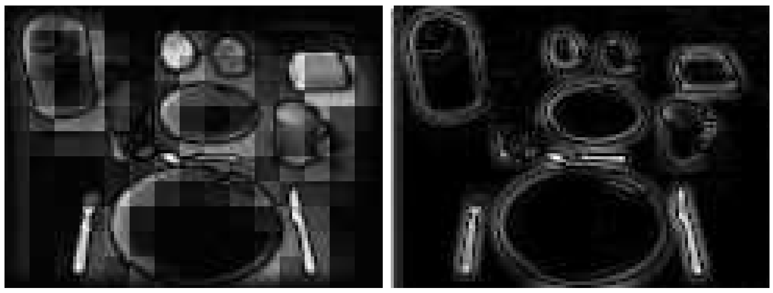
scale s_{i-1} is done by bilinear interpolation. After resizing, the maps are added up pixel by pixel. This yields 2 intensity maps I'_i, 4 orientation maps O'_θ and 4 color maps C'_γ. Next, we use a weighted sum to create 3 conspicuity maps $I = \sum_i \mathcal{W}(I'_i), C = \sum_\gamma \mathcal{W}(C'_\gamma)$, and $O = \sum_\theta \mathcal{W}(O'_\theta)$. These are finally fused to get the saliency map $S_{bu} = \mathcal{W}(I) + \mathcal{W}(O) + \mathcal{W}(C)$.

The weighting function $\mathcal{W}$ which is used to fuse the feature and conspicuity maps is called *uniqueness weight* and is maybe the most important part of computational attention systems: it emphasizes "important" maps, i.e. those with few peaks, enabling the detection of *pop-outs*. $\mathcal{W}$, applied to a feature map X, is defined as $\mathcal{W}(X) = X/\sqrt{m}$, where m is the number of local maxima that exceed a threshold. As a result of the weighting, the most salient region in the table soccer image in Fig. 2 is the ball, because it sticks out in the intensity channel.

In top-down mode, the system aims to detect a target, i.e., input to the system is the image and some target information, provided as a feature vector $\boldsymbol{v}$. This vector is learned from a region which is provided manually or automatically. In *search mode*, the system multiplies the feature and conspicuity maps with the weights of $\boldsymbol{v}$. The resulting maps are summed up, yielding the *top-down saliency map* S_{td}. Finally, S_{bu} and S_{td} are combined by: $S = (1-t) * S_{bu} + t * S_{td}$, where t determines the contributions of bottom-up and top-down (details in [2]).

3 Fast Feature Computation using Integral Images

The most time consuming part in VOCUS are the feature computations, since filters of different sizes are applied to several layers of the image pyramids. As mentioned earlier, Itti et al. [7] use an approximation to achieve faster feature computations: instead of applying linear filters to each pyramid level, center-surround features are computed as differences between pyramid levels: the center is a pixel at scale $c \in \{2, 3, 4\}$ and the surround is the corresponding pixel at scale $s = c + \delta$, with $\delta \in \{3, 4\}$. As a result, c is usually not exactly in the center of the surrounding region s, but is one of the pixels in a squared region. This leads to transition effects at the borders of the squares as illustrated in Fig. 3), left.



(a) Intensity map of NVT (b) Intensity map of VOCUS

Fig. 3. Two intensity maps of a breakfast table scene, computed by the NVT [7] (left) and by our system VOCUS (right). The square-textured structure in the left image resulting from taking the difference between two scales can be seen clearly, the right image shows a more accurate solution.

In VOCUS, we attached more importance to the accuracy of the feature computations, therefore we computed the features with rectangular filters which considered the exact surrounding region. The result is shown in Fig. 3), right. Even more accurate were circular filters but since we found no major difference in quality, we used the simpler rectangular version. Although more accurate, the filter computations slowed down the system considerably, since the time complexity depends on the filter size which was up to 15 × 15 pixels. In the following, we describe how a fast computation, independent on the filter size, can be achieved without losing any accuracy in comparison to the common linear filter approach.

The fast feature computations are achieved with *integral images* [15], also known as summed area tables [1]. The advantage of an integral image is that when it is once created, the sum and mean of the pixel values of an area of arbitrary size can be computed in constant time. An integral image I is an intermediate representation for the image and contains the sum of gray scale pixel values of image N with height y and width x, i.e.,

$$I(x,y) = \sum_{x'=0}^{x} \sum_{y'=0}^{y} N(x',y'). \tag{1}$$

A visualization is depicted in Fig. 4, left. The integral image is computed recursively by the formula:

$$I(x,y) = I(x,y-1) + I(x-1,y) + N(x,y) - I(x-1,y-1) \tag{2}$$

with $I(-1,y) = I(x,-1) = I(-1,-1) = 0$. The computation requires only one scan over the input data. This intermediate representation $I(x,y)$ allows the

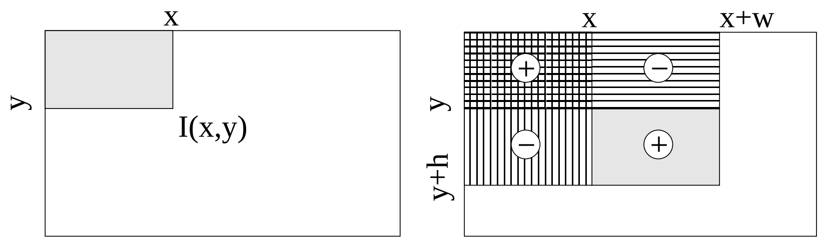


Fig. 4. Left: The integral image contains at $I(x, y)$ the sum of the pixel values in the shaded region. Right: the computation of the average value in the shaded region is based on four operations on the four depicted rectangles according to eq. 3.

computation of a rectangle feature value at (x, y) with height h and width w using four references (see Fig. 4 (right)):

$$\begin{aligned} F(x, y, h, w) = I(x + w, y + h) - I(x, y + h) \\ -I(x + w, y) + I(x, y). \end{aligned} \tag{3}$$

In VOCUS, we apply integral images for the features intensity and color, since here we can replace the rectangular center-surround computations without losing any information. It is also possible to use them for the orientation feature, but since these are realized by Gabor filters, rectangular features would only achieve approximate results. We leave an evaluation of the quality of this approximation and the decision whether the loss is acceptable to future work. Here we concentrate on changes which do not change the system behaviour.

We compute respectively one integral image for the scales $2 - 4$ of the intensity and color image pyramids. This was a solution which enabled to keep the main structure of the system. In a future redesign of the system, we consider to replace the image pyramids by integral images, since the feature computation across scales can also be performed on a single integral image.

The central element of the feature computation in VOCUS is the center-surround difference algorithm. It computes the difference between a pixel value and its surrounding area of a certain radius r. The on-off difference highlights bright pixels on dark background whereas the off-on difference highlights dark pixels on bright background.

Before optimizing, the surrounding area σ was directly computed as the mean of the pixel values by summing up the values and dividing by the number of pixels. The time-consuming part, the summing up, is replaced now by integral images: the surrounding area σ can be computed for an arbitrary radius r straightforward by only slightly modifying eq. 3:

$$\begin{aligned} \sigma(x, y) = I(x + r, y + r) + I(x - r, y - r) - \\ I(x + r, y - r) - I(x - r, y + r) - N(x, y). \end{aligned} \tag{4}$$

This is basically the same equation as eq. 3, the only differences are that the reference pixel (x, y) is now in the center of the region and the value of the center pixel $N(x, y)$ is subtracted. The on-off difference is then just the difference of the current pixel and the mean pixel value in the surround:

$$\text{on-off-difference}(x, y) = N(x, y) - \frac{\sigma(x, y)}{s} \tag{5}$$

where $s = (2r+1)^2 - 1$ is the number of pixels inside the surround. The off-on difference can be computed analogous. Both, on-off as well as off-on differences, are used to compute the intensity feature maps. For the color maps, only the on-off difference is needed, because the opponent color is represented by an own color map (green vs. red and yellow vs. blue) (details in [2]).

4 Results

In this section, we describe the optimization process and the time savings we achieved. Before starting with the optimizations, the center-surround difference was computed by rectangular filters, that means the surround was computed directly as the sum of the pixels in the surrounding region. This yielded accurate results as displayed in Fig. 3, but was too slow for real-time performance (cf. first row of Table 1). First, we analyzed the runtime behaviour of our attention system VOCUS with a profiler on three different image sizes ($200 \times 150, 400 \times 300$, and 800×600). It turned out that by far the most time was spent within pixel access functions. This had two reasons: first, the pixel access function was too slow and second, the function was called extremely often: several million times for the 400×300 image.

We decided to optimize the runtime of VOCUS in two steps. In the first step, we applied standard optimization methods, e.g. to speed up the pixel access function. In the second step, we introduced integral images as explained in sec. 3 to reduce the number of pixel accesses. This is the main part of the paper. This separation enables to exactly determine which speed-up factor results from the integral images and which from other optimizations.

In the first optimization step, we started by speeding up the pixel access function. VOCUS uses the open source computer vision library `OpenCV`, which allows fast image processing algorithms. However, the pixel access functions `cvGet2D` and `cvSet2D` are computationally expensive. As suggested in the `OpenCV` documentation, we changed the way of accessing the pixel values by directly accessing the raw data array. Furthermore, we made use of precomputing variables in loops, and the separation of filter kernels, which means that a 2D convolution is replaced by two 1D convolutions. As can be seen in the second row of table 1, the first optimization step yields a runtime speed up of factor 5.

After standard optimizations, it becomes usually more and more difficult to achieve a significant time speed up. An analysis with the profiler revealed that the function to compute the center-surround differences, which are used for the

Image size	200 × 150	400 × 300	800 × 600
Before optimizing	0.650	2.720	11.220
Standard optimizations	0.120	0.510	2.070
Optimized with integral images	0.010	0.050	0.190

Table 1. Performance of our attention system VOCUS (times in seconds), computed on a 2.8 GHz PC. The first optimization step (2nd row) uses standard optimizations, e.g., a faster pixel access method. The second optimization step (3rd row) differs from the first optimization step only by replacing the filter-based center-surround computation by computations based on integral images.

computation of the intensity and color maps, took still 85% of the total runtime. At this point, we replaced the center-surround computations by integral images as described in sec. 3. These optimizations resulted in an additional time speed up of factor 10 compared to the optimized code with normal filters (cf. Table 1). The function to compute the center-surround differences, in which most of the pixel accesses took place, was reduced to 18 % of the total runtime.

After these optimizations, VOCUS is real-time capable and useful for practical applications in computer vision and robotics. We have demonstrated this in [4], where VOCUS was used on a mobile robot in the context of visual SLAM (Simultaneous Localization and Mapping).

5 Conclusion

Computational visual attention systems determine salient image regions usually by applying linear filters to different scales and feature dimension of an input image. The computational effort of these techniques is usually too high to meet real-time constraints; standard optimization techniques can yield a fair but limited improvement. In this article, we have presented a method for further improving the general performance of a computational visual attention system by a typical factor of 10. The method uses integral images for the feature computations and reduces the number of necessary pixel accesses significantly, since it enables the computation of arbitrarily sized feature values in constant time. In contrast to other optimizations which approximate the feature values, our method is accurate and provides the same results as the filter-based methods. The computation of regions of interest can now be performed in real time for reasonable initial image resolutions (half VGA) and thus allow their use in a variety of applications. Attention can now be used for feature tracking or for preselecting landmark features for visual SLAM.

So far, the optimization has been applied only to the color and intensity feature dimensions. We plan to use integral images for the computation of the orientation feature, too. Since the currently used Gabor filters are more accurate than the rectangular features necessary for the integral images, the loss of accuracy has to be traded off against the speed-up. Another possibility for improvement is to make the image pyramids obsolete and perform all required computations on a single integral image per feature dimension. Scale-invariant feature computations could then be achieved by applying differently sized rectangular features to the integral images. The expected improvements will allow frame rate computations at even higher initial image resolutions.

References

1. Crow, F. C. Summed-area tables for texture mapping. *Proceedings of SIGGRAPH* **18** (3, 1084) 207–212.
2. Frintrop, S. VOCUS: A Visual Attention System for Object Detection and Goal-directed Search. PhD thesis Rheinische Friedrich-Wilhelms-Universität Bonn Germany (2005). Published 2006 in Lecture Notes in Artificial Intelligence (LNAI), Vol. 3899, Springer Verlag Berlin/Heidelberg.
3. Frintrop, S., Backer, G. and Rome, E. Goal-directed Search with a Top-down Modulated Computational Attention System. In: Proc. of the Annual meeting of the German Association for Pattern Recognition (Jahrestagung der Deutschen Arbeitsgemeinschaft für Mustererkennung) DAGM 2005 Lecture Notes in Computer Science (LNCS) Springer (2005) 117–124.
4. Frintrop, S., Jensfelt, P. and Christensen, H. Attentional Landmark Selection for Visual SLAM. In: Proc. of the International Conference on Intelligent Robots and Systems (IROS '06) (2006).
5. Garey, M. and Johnson, D. S. *Computers and Intractability, A Guide to the Theory of NP-Completeness* Freeman San Francisco 1979.
6. Itti, L. Real-Time High-Performance Attention Focusing in Outdoors Color Video Streams. In: Proc. SPIE Human Vision and Electronic Imaging IV (HVEI '02), B. Rogowitz and T. N. Pappas (Eds.) (2002) 235–243.
7. Itti, L., Koch, C. and Niebur, E. A Model of Saliency-Based Visual Attention for Rapid Scene Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **20** (11, 1998) 1254–1259.
8. Lowe, D. G. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision (IJCV)* **60** (2, 2004) 91–110.
9. Mikolajczyk, K. and Schmid, C. Indexing Based on Scale Invariant Interest Points. In: Proc. of ICCV (2001) 525–531.
10. Ouerhani, N. and Hügli, H. Real Time Visual Attention on a Massively Parallel SIMD Architecture. *Int. Journal of Real-time Imaging, Elsevier Computer Science* **9** (3, 2003) 189–196.
11. Palmer, S. E. *Vision Science, Photons to Phenomenology* The MIT Press Cambridge, MA 1999.
12. Tsotsos, J. K. Analyzing Vision at the Complexity Level. *Behavioral and Brain Sciences* **13** (3, 1990) 423–445.
13. ———Complexity, Vision, and Attention. In: Vision and Attention, M. Jenkin and L. R. Harris (Eds.) Springer Verlag 2001 chapter 6.
14. Tsotsos, J. K., Culhane, S. M., Wai, W. Y. K., Lai, Y., Davis, N. and Nuflo, F. Modeling Visual Attention via Selective Tuning. *Artificial Intelligence* **78** (1-2, 1995) 507–545.
15. Viola, P. and Jones, M. J. Robust Real-Time Face Detection. *International Journal of Computer Vision (IJCV)* **57** (2, 2004) 137–154.