Cogn Comput (2011) 3:264–278
DOI 10.1007/s12559-010-9093-9

Adaptive Gaze Control for Object Detection

G. C. H. E. de Croon · E. O. Postma · H. J. van den Herik

Received: 31 March 2010 / Accepted: 26 December 2010 / Published online: 15 January 2011

Abstract We propose a novel gaze-control model for detecting objects in images. The model, named ACT-DETECT, uses the information from local image samples in order to shift its gaze towards object locations. The model constitutes two main contributions. The first contribution is that the model's setup makes it computationally highly efficient in comparison with existing window-sliding methods for object detection, while retaining an acceptable detection performance. ACT-DETECT is evaluated on a face-detection task using a publicly available image set. In terms of detection performance, ACT-DETECT slightly outperforms the window-sliding methods that have been applied to the face-detection task. In terms of computational efficiency, ACT-DETECT clearly outperforms the window-sliding methods: it requires in the order of hundreds fewer samples for detection. The second contribution of the model lies in its more extensive use of local samples than previous models: instead of merely using them for verifying object presence at the gaze location, the model uses them to determine a direction and distance to the object of interest. The simultaneous adaptation of both the model's visual features and its gaze-control strategy leads to the discovery of features and strategies for exploiting the *local* context of objects. For example, the model uses the spatial relations between the bodies of the persons in the images and their faces. The resulting gaze control is a temporal process, in which the object's context is exploited at different scales and at different image locations relative to the object.

Keywords Gaze control · Computationally efficient object detection · Active vision · Evolutionary algorithms

G. C. H. E. de Croon (✉)
Micro Air Vehicle Lab, Control and Simulation Department, Technical University Delft, Delft, The Netherlands
e-mail: g.c.h.e.decroon@tudelft.nl

G. C. H. E. de Croon
Department of Artificial Intelligence, Radboud University Nijmegen, Nijmegen, The Netherlands

E. O. Postma · H. J. van den Herik
Tilburg Innovative Computing Center, Universiteit van Tilburg, Tilburg, The Netherlands
e-mail: e.o.postma@uvt.nl

H. J. van den Herik
e-mail: h.j.vdnherik@uvt.nl

Introduction

Humans typically detect an object quickly and accurately by exploiting its visual context. Violations of the context impair the speed and accuracy of the detection (cf. [5, 6, 18, 26]). To detect an object, humans exploit its (statistical) relations to other scene elements, which are due to the structure of the world [5, 6, 13, 18, 26, 37, 38, 50, 52, 55].

Object-Detection Methods

Remarkably, standard artificial object-detection methods typically *do* scan the entire image in search for objects of interest. For instance, window-sliding methods (e.g., [35, 42, 53, 66, 70, 74, 78]) employ *exhaustive search* to evaluate the presence of an object at all locations of an evenly spaced grid. They slide a window over the image, extracting a local sample at each grid point and classifying it either as an object or as a part of the background. Consequently, window-sliding methods extract a large number of local image samples.

Several object-detection methods were proposed in an attempt to limit the number of sample locations at which

object presence is evaluated. Typically, these methods proceed in two stages: (1) exhaustively scan the image to detect *interest points*, and (2) evaluate local samples only at the interest points. We briefly describe three methods employing these two stages: constellation-based methods, object-context methods, and region-of-interest methods.

First, constellation-based methods (cf. [10, 11, 20, 22, 23, 29, 41, 43, 47, 80, 81, 82]) detect objects by detecting constellations of their parts. For the first stage they calculate interest values for local samples at all points of an evenly spaced grid (typically processing the entire image). Examples include the calculation of the entropy of grey-scale values at multiple scales [31], and difference-of-Gaussian responses (in the SIFT-approach by [44]). Subsequently, at the interest points, the methods extract new local samples that are identified either as a part of the object or as a part of the background. An object is recognised if there is a constellation of recognised parts that is sufficiently similar to a learned constellation object model.

Second, object-context methods exploit the local spatial context of an object during scanning. These methods perform an initial scan to identify probable object context locations [4, 21, 27, 28, 35, 56, 67, 83], and subsequently apply an object classifier at these locations only, to verify the presence of an object.

Third, region-of-interest methods do not scan the entire image for object detection, but use a coarse global representation of the image in order to determine a region of interest [49, 76, 77]. The region of interest can then be (exhaustively) scanned in order to find the objects of interest [49].

In a different spirit from the object-detection methods mentioned earlier, [39, 40] reduce the number of sample evaluations by successively excluding regions of the image with a branch-and-bound scheme. In this manner the computational effort necessary for object detection can be significantly reduced.

Although the aforementioned object-detection methods limit the number of image samples evaluated with the object classification function, they do not exploit the coarse-scale and fine-scale scene elements to a degree that is comparable to human observers. In that respect, object-detection methods may incorporate ideas from gaze-control models to be discussed in the next section.

Gaze-Control Models

Visual attention mechanisms allow humans to search for objects efficiently. Many models for human visual attention have been introduced, e.g., [7, 8, 30, 57, 59, 61, 64, 73, 77]. Many of these models process the entire image to determine interest points that can be used by any computer vision algorithm for subsequent processing. A subset of visual attention models focuses on the processing of only local samples, for example, [3, 34, 36, 54, 62, 68, 75]. In a cornerstone paper on *active vision*,[1] [2] suggested to use such *gaze-control models* for the task of object detection. He argued that such models can use contextual cues to shift the gaze to probable object locations. For example, the corner of a table provides a contextual cue to the location of an object standing on the table (e.g., a mug).

Following this idea, it is clear that the main potential advantage of gaze control for object detection is a higher computational efficiency. Gaze-control models perform an *informed search* for an object. As a result, they can find objects by extracting far fewer samples than in an exhaustive search. Computational efficiency is important for object-detection tasks with time constraints, such as object detection on a miniature robotic platform (cf. [19, 65]), or object search in large image databases. Of course, the goal of gaze control is to obtain a higher computational efficiency, while retaining as good a detection performance as possible.

A small group of gaze-control models has been proposed purely for performing the task of object detection. Some studies [48, 62, 63, 72, 84] introduced gaze-control models for detecting a specific object in a visual scene. These models mainly shifted their gaze to the part of the visual scene that was most similar to the object. Other studies [24, 32, 46, 69] investigated gaze-control models that had to find black geometrical shapes (triangles or squares) in artificial images with a white background, under various amounts of noise. These studies emphasised that the gaze-control models should not follow a predefined gaze-control strategy, but should *adapt* their strategy to the object-detection task at hand. In more recent work, the gaze-control model in [12] extracted shape-based features at all positions in an image and combined this with bottom-up saliency and top-down cues in order to generate an attention map. The model built this map once and then shifted its gaze to the most promising locations in turn. The areas of attention overlapped for 92% with the eye movements of human observers, although this correspondence decreased after the first observation. Finally, in [79] a gaze-control model was presented that combined bottom-up and top-down cues with the modelling of temporal aspects of gaze control. The bottom-up visual saliency of the image was merged with an object location prior to form an attention map. Gaze control consisted of an iterative process of planning a gaze location, checking for object presence, and updating the attention map. Since the gaze-control model extracted visual features at all positions in

[1] Ballard himself used the term *animate vision*, to prevent confusion with active sensors, such as laser rangefinders. However, *active vision* remains the most commonly employed term.

the image and reasoned about all permutations of actions, it involved a considerable computational effort.

Although the gaze-control models mentioned earlier have contributed to the understanding of gaze control, none of them are competitive yet with existing window-sliding methods for object detection. Most of the models have been made for modelling and explaining human eye movements. For determining regions of interest, i.e., potential target locations of eye movements, they extract many different local features exhaustively from the image. Consequently, the computational load is large. As a case in point, in [79] processing a single image requires 212 s.

Contributions

In this article, we introduce a gaze-control model, named ACT-DETECT, which performs the task of object detection by processing local image samples.

In comparison with existing models of gaze control, ACT-DETECT makes two main contributions. The *first contribution* is that the model's setup makes it computationally highly efficient: in comparison with standard window-sliding methods only very few samples are extracted from an image to perform object detection. This computational efficiency does not imply that the detection performance is below par. On the face-detection task studied in this article, ACT-DETECT actually slightly outperforms existing object-detection methods. The *second contribution* of the model lies in its more extensive use of local samples than previous models: instead of merely using them for verifying object presence at the gaze location or for determining likely object locations, the model uses them to determine a direction and distance to the object of interest. The simultaneous adaptation of both the model's visual features and its gaze-control strategy leads to the discovery of features and strategies for exploiting the *local* context of objects. For example, in contrast to existing gaze-control models, the model can use the spatial relations between the bodies of the persons in the images and their faces. The resulting gaze control is a process over time, in which the object's context is exploited at different scales and at different distances from the object.

The remainder of the article is structured as follows. The gaze-control model is described in "ACT-DETECT". The model is adapted to a face-detection task, which is explained in "Setup Face-Detection Experiment". Subsequently, in the "Performance Face-Detection Experiment", ACT-DETECT's performance is compared with that of standard window-sliding methods. Then, its visual features and gaze-control strategy are analysed in "Analysis Face-Detection Experiment". We compare the computational effort of ACT-DETECT with that of window-sliding methods in "Computational Efficiency". Subsequently, we discuss the results and draw conclusions in "Discussion and Conclusion".

ACT-DETECT

The gaze-control model ACT-DETECT consists of two modules: a feature extraction module and a controller module (see Fig. 1). ACT-DETECT operates on a multi-scale image representation (pyramid) consisting of s scaled versions of the input image. The multi-scale representation allows ACT-DETECT to perform a coarse-to-fine strategy and to use information on an object's location at different scales. The choice for the number of image scales, s, depends on the task (in the figure $s = 3$). The feature-extraction module extracts informative low-level visual features (e.g., oriented edges) from a local region of the multi-scale representation. The types and locations of the features are adaptable parameters. It outputs a numerical representation indicating the presence of these features within the local region. The controller module maps the numerical representations to actions (gaze shifts) by means of a nonlinear function with adaptable parameters. The adaptable parameters of ACT-DETECT are scale specific. So although the feature extraction and controller modules at each scale are identical, their parameters may be different.

The object-detection process in an image proceeds as follows. ACT-DETECT starts a detection sequence at a random location at the coarsest scale in the image pyramid. It then performs t gaze shifts (solid arrows) and moves down to the next, more detailed, scale (thick lines). At the next scale, these steps are repeated. Provided that the adaptive parameters have appropriate values, ACT-DETECT progressively refines its search for the object. The run ends when ACT-DETECT has made t gaze shifts on the finest scale. Since ACT-DETECT only takes local samples at each scale, it is in principle not necessary to construct the entire multi-scale representation for each image.

On the basis of findings in an earlier study [17], in this paper we employ $s = 2$ scales, the coarse scale and fine scale, respectively. We refer to the gaze-control model with

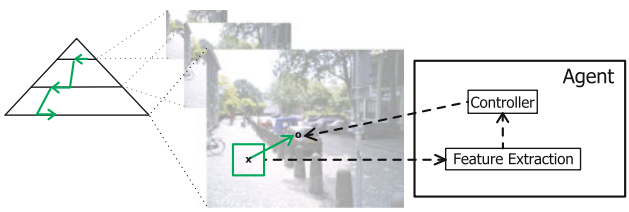

Fig. 1 Overview of ACT-DETECT. The *pyramid* on the *left* shows that ACT-DETECT exploits visual features on different image scales. The *right* part of the figure illustrates the core concept of ACT-DETECT: it has a closed loop of extracting visual features and performing gaze shifts. For a more detailed explanation we refer the reader to the text

the parameters for the coarse scale as the *coarse-scale model* and to the gaze-control model with the parameters for the finer scale as the *fine-scale model*.

The right part of Fig. 1 illustrates the core concept of gaze control for object detection: the model has a closed loop of visual inputs and gaze shifts. The figure shows for the most detailed scale how ACT-DETECT takes a local image sample from the gaze window (box), centred at the current gaze location (x). The model extracts visual features from this sample and passes the feature values to the model's controller, which maps the features to a relative gaze shift in the image to the new gaze location (o). In our experiments, ACT-DETECT makes $t = 5$ gaze shifts per scale.[2]

The final gaze location at the finest scale is considered a candidate object location. We use a standard object classifier to verify whether this location actually contains an object.

Below the implementation details are discussed. The details regard the feature extraction module (section "Feature Extraction Module"), the controller module (section "Controller Module"), and the object classifier that verifies the presence of an object at the final gaze location (section "Object Classifier"). Finally, we discuss the evolutionary algorithm that is used for optimising the adaptable parameters of the visual feature extraction and controller modules (section "Evolutionary Algorithm").

Feature Extraction Module

To facilitate comparison with existing window-sliding methods, the feature-extraction module employs features that are often used by such methods: the *integral features* introduced in [78]. The main advantage of these features is that they can be extracted with little computational effort, independent of their scale.

ACT-DETECT extracts n integral features, where n depends on the task. Each integral feature has a type and an area in the gaze window, both of which can be optimised for the task and the scale at hand. The top row of Fig. 2 shows the types of integral features that we use in our object-detection experiments. The features can occupy any area inside of the gaze window.

The bottom row of Fig. 2 (left) shows an example feature within the gaze window (surrounding box). It is of type 1 and spans a large part of the right half of the gaze window. The value of this feature is equal to the mean grey-value of all pixels in area A minus the mean grey-value of all pixels in area B (see Fig. 2 right). The example feature

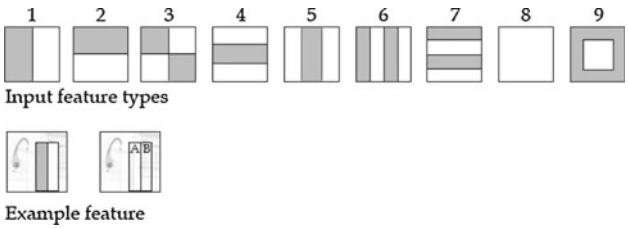

Fig. 2 Possible feature types (*top* part of the figure) and an example feature shown in the gaze window (*bottom* part of the figure)

will respond to vertical contrasts in the image, slightly to the right of the gaze location (in the centre of the gaze window).

The adaptable parameters of the feature extraction module consist of the types and coordinates of the integral features in the gaze window. These parameters are optimised using an evolutionary algorithm (see "Evolutionary Algorithm").

Controller Module

The controller module takes the n extracted features as input. It is a completely connected multilayer feedforward neural network, with $h = \lfloor n/2 \rfloor$ hidden neurons and $o = 2$ the output neurons. Both the hidden and output neurons have a sigmoid activation function:

$$a(z) = \tanh(z), \quad a(z) \in \langle -1, 1 \rangle, \tag{1}$$

where z is the weighted sum of the inputs to the neuron. The two output neurons, out_1 and out_2, encode for the gaze shift $(\Delta x, \Delta y)$ in pixels as follows:

$$\Delta x = \lfloor d_{\max} \times \text{out}_1 \rfloor, \tag{2}$$

$$\Delta y = \lfloor d_{\max} \times \text{out}_2 \rfloor. \tag{3}$$

The constant $d_{\max}$ represents the maximal displacement in the image in pixels. If a gaze shift brings the gaze window over the border of the image, the gaze location is reset to the closest possible valid location.

The adaptable parameters of the controller module consist of the neural network's weights. They are adapted to the object-detection task with an evolutionary algorithm (section "Evolutionary Algorithm").

Object Classifier

The verification whether the final gaze locations are located at object locations is performed by a Support Vector Machine (SVM) [33, 53, 60].[3] To train the classifier, we gathered image samples at the final gaze locations in the

[2] The number of time steps represents a trade-off between detection performance and computational efficiency: employing more time steps results in a higher performance, but also less computational efficiency.

[3] In previous studies [15, 16] we only used the first stage of a Viola and Jones classifier, i.e., a simple linear classifier. The classifier was trained on all locations in all training images.

training set. The inputs to the classifier are also integral features (see Fig. 2). The SVM was trained with the SMO-algorithm [58], with $c = 10$ and an RBF-kernel with $\gamma = 0.75$.[4] Please remark that since the image set described in "Setup Face-Detection Experiment" does not have large variations in object sizes, the classifier is applied only at one scale.

Evolutionary Algorithm

ACT-DETECT's parameters are optimised by means of a λ, μ-evolutionary algorithm [1, 51]. We employ an evolutionary algorithm mainly because (1) it is a semi-supervised optimisation method, which can thus find non-greedy action strategies, and (2) it can optimise multiple parts of the model at the same time. As explained earlier, we optimise both ACT-DETECT's visual features and controller, which may improve the performance on the task (see e.g., [45]).

For evaluation purposes, the image set associated with the detection task at hand is divided into two parts: half of the images is used for evolution and half of the images for testing. We first evolve the coarse-scale model parameters, starting from uniformly distributed locations. Since one run of ACT-DETECT can only lead to the detection of one object, we always perform $R = 10$ runs per image. The coarse-scale model is evolved to optimise the detection rate, i.e., the proportion of objects present in the training set that are detected by the ensemble of runs of ACT-DETECT. Then, we evolve the parameters of the fine-scale model, which always starts from the end locations of the already evolved coarse-scale model. The fine-scale model is evolved to minimise the number of false positives, i.e., the proportion of runs that do not end up at an object location. For the evolution of each model, we use a population size of $\lambda = 100$ and select the best $\mu = 25$ genomes to create a next generation. Evolution continues for $g = 300$ generations.

The genome representing the search space of the evolutionary algorithm is a vector of real values (double precision) in the interval $[-1, 1]$. The first part of the genome encodes for the visual features of ACT-DETECT. Each feature is represented by five values, one for the type (gene_1) and four for the two coordinates inside the gaze window (gene_2 to gene_5). The type of the feature is decoded as follows: $\text{type} = [|\text{gene}_1| \times 8] + 1$, where $[\cdot]$ is the round-function. The left coordinate of the feature in the window is decoded as: $\text{left} = \min(|\text{gene}_2|, |\text{gene}_3|) \times$ window_width. The right coordinate is then: $\text{right} = \max(|\text{gene}_2|, |\text{gene}_3|) \times$ window_width. We determine the top and bottom coordinates in the same manner, but with gene_4 and gene_5.

The second part of the genome encodes for the neural network weights. Each weight is directly represented by one value, which implies a weight range of $[-1,1]$. The probability for a mutation in the genome is $p_{\text{mut}} = 0.04$, and for one-point crossover with another selected genome is $p_{\text{co}} = 0.5$.

After the optimisation of the fine-scale model, we gather the image samples at the final gaze locations in the training set to evolve the object classifier. We use the same training parameters for the evolution of the object classifier as for the gaze-control models, but the genome only consists of double values encoding the visual features. The visual features of the object classifier are evolved to optimise the proportion of correctly classified samples.

Setup Face-Detection Experiment

In the *face-detection experiment*, ACT-DETECT is adapted to a face-detection task. We explain the task in "Face-Detection Task". In "Performance Face-Detection Experiment", ACT-DETECT's performance is compared with that of window-sliding methods. Then, in the section "Analysis Face-Detection Experiment", we analyse the evolved gaze strategy.

Face-Detection Task

The face-detection task consists of detecting frontal faces in the publicly available FGNET image set.[5] The choice for this set is motivated by the fact that the results can be compared with those reported in the literature, mainly obtained with variations of the Viola and Jones object detector [78].

The FGNET image set contains video sequences of a meeting room, recorded by two different cameras. The experiments involved the joint set of images from both cameras ('Cam1' and 'Cam2') in the first scene ('ScenA'). The set consists of 794 images of 720 × 576 pixels, which are converted to grey scale. Figure 3 shows five example images from the set. We use the ground truth data that is available online, in which only the faces with two visible eyes are labelled. For evolution, the image set is divided into two parts: half of the images is used for testing and half of the images for evolution. A two-folded test is used to obtain the results, with one evolution per fold.

For the face-detection task, ACT-DETECT extracts $n = 10$ features. Furthermore, the maximal displacement of the

[4] We used a publicly available implementation of this algorithm, which can be downloaded at http://theoval.sys.uea.ac.uk/~gcc/svm/toolbox/. For a more detailed explanation of the SVM and its parameters, we refer the reader to [9] and [58].

[5] The FGNET image set is available at (http://www-prima.inrialpes.fr/FGnet/).

Fig. 3 Five example images from the FGNET image set

gaze window in pixels is $d_{\max} = 360$ for the coarse-scale model and $d_{\max} = 240$ for the fine-scale model. All the other parameter settings have been discussed in "ACT-DETECT".

Performance Face-Detection Experiment

The evolutionary algorithm finds successful gaze-control strategies for the face-detection task. Figure 4 shows ten independent runs of the best instance of ACT-DETECT (first fold). The arrows represent the gaze shifts of ACT-DETECT. At time step $i = 0$, all runs are initialised at random locations in the image. The coarse-scale model then makes five gaze shifts, followed by another five gaze shifts of the fine-scale model. At the last time step ($i = 10$) seven out of ten runs have reached an object location. The local image samples at the final gaze locations are classified by the trained SVM. Circles indicate positive classifications, crosses negative classifications. The gaze shifts leading to a positive classification are blue for the coarse-scale model and red for the fine-scale model. Gaze shifts leading to a negative classification are grey. ACT-DETECT successfully detects all three faces in the image.

For classification tasks, it is common to use a Receiver Operating Characteristic plot (ROC-plot) for comparing the performances of different methods. A ROC-plot shows the relation between the true positive rate and the false positive rate. In object-detection tasks, this would correspond to the proportion of objects in the image set detected by the method, and the proportion of non-object locations classified as objects. Since the number of samples evaluated in an image can differ between methods, it is better to construct a Free-response Receiver Operating Characteristic plot (FROC-plot). It plots the detection rate against the average number of false positives per image. Of course, the goal of an object-detection method is to achieve a high detection rate and a low number of false positives.

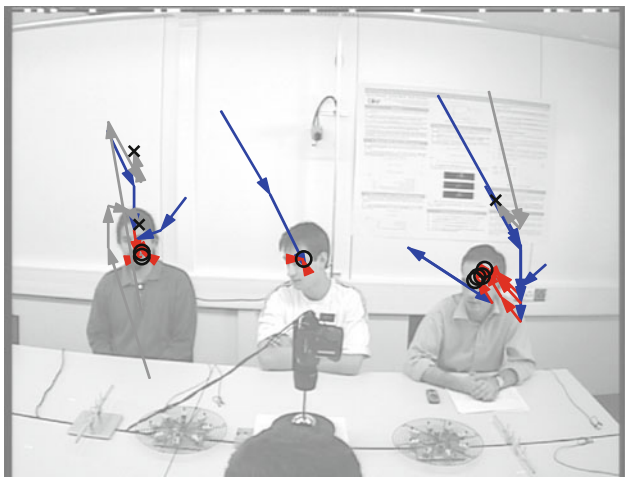

Fig. 4 Ten independent runs of ACT-DETECT on an image from the FGNET data set

For object-detection methods that mainly rely on a binary classifier, an FROC-plot can be constructed by varying the threshold of this classifier. Since both the gaze shifts and the object classification are of importance to the performance of ACT-DETECT, it is not evident how to construct an FROC-plot. Here, we mention three factors that are of influence. First, gaze control itself represents a choice for computational efficiency at the possible cost of the detection rate. The approach implies that parts of the image are skipped. Second, the fitness function is of influence on the FROC-plot. For example, the fitness function of the coarse-scale model puts an emphasis on the detection rate, while the fitness function of the fine-scale model does so on the number of false positives. Third, the number of independent runs is positively related to the detection rate and number of false positives. A higher number of runs results in more detections and false positives. We use the third factor to construct the FROC-plot of ACT-DETECT, since it is the easiest factor to vary. We use $R = \{1, 3, 5, 10, 20, 30, 50, 75, 100, 150, 200, 250, 300\}$ for generating the curve.

Figure 5 contains an FROC-plot of ACT-DETECT's results, averaged over the experiments with both folds (square markers, dotted line). In addition, the figure shows the results on the FGNET image set of the detector by Cristinacce and Cootes [14] (solid lines), of a Fröba–Küllbeck [25] detector (plus-markers) and a Viola and Jones [78] detector (circular markers). It also includes the results by Kruppa et al. of two Viola and Jones detectors trained on a separate image set and tested on the FGNET set [35] (dashed lines). The first of these detectors attempts to detect face regions in the image in the same manner as the detectors in [14]

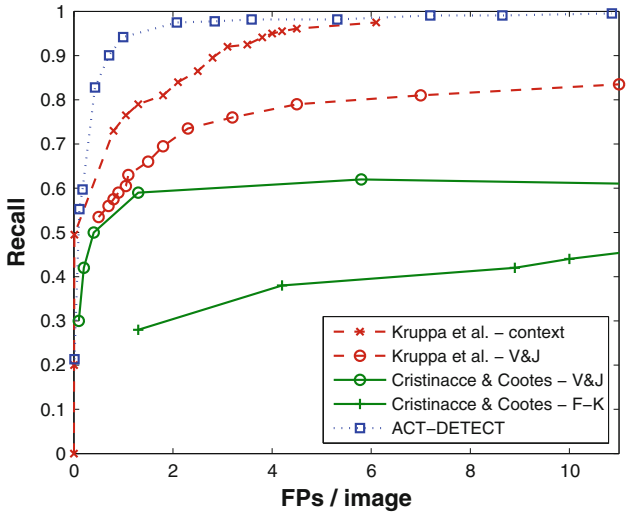

Fig. 5 FROC-plots of the different object-detection methods for the face-detection task

(circular markers). The second of these detectors attempts to detect a face by including a region around the face, containing head and shoulders (cross-markers).

Figure 5 leads to the observation that ACT-DETECT *has a better detection performance than the window-sliding methods on the* FGNET *data set*. This is surprising, since ACT-DETECT avoids large image regions in the detection process. The second best method is the one by [35], which takes an object's context into account. Detecting faces without considering context is difficult in the FGNET video-sequence, because the appearance of a face can change considerably from image to image [14]. However, the context of a face (such as head and shoulders) is rather fixed. This is why approaches that exploit this context [4, 35] have a more robust performance. The active object-detection method exploits context even to a greater extent than the methods studied in [4] and [35].

It is interesting to note the difference between the Viola and Jones classifiers used in [14] and [35]. The difference can be explained by at least three factors: a different training set, different settings of the training parameters for the Viola and Jones classifier, and different ground truth data. In contrast to the ground truth data available online, [35] also labelled profile faces.

Disregarding small differences between the experiments, the results show that ACT-DETECT performs at least as good as the window-sliding methods on the FGNET face-detection task.

Analysis Face-Detection Experiment

In this section, we investigate how ACT-DETECT selects sensible gaze shifts. The analysis focuses on the best-evolved instance of ACT-DETECT on the first fold, with an emphasis on the coarse-scale model. We first study its evolved visual features (section "Visual Features") and then its mapping from visual inputs to gaze shifts in the image (section "Gaze Shifts").

Visual Features

The evolved visual features capture properties of the object and its context. Figure 6 shows the $n = 10$ evolved input features for the coarse-scale model. The features are projected on an image from the training set. They are shown at their locations and with their sizes within the gaze window (white box). The white cross indicates the centre of the gaze window.

In order to interpret the evolved features, we extract them at all possible gaze locations within the image and store their values. We scale these values to obtain *feature responses* in the interval [0,1]. Figure 7 shows a (darkened) image in the background and the feature responses on all possible gaze locations on the foreground. There are no feature responses in the border of the image, since the gaze window cannot go over the border of the image. High intensity regions mark high responses, low intensity regions low responses.

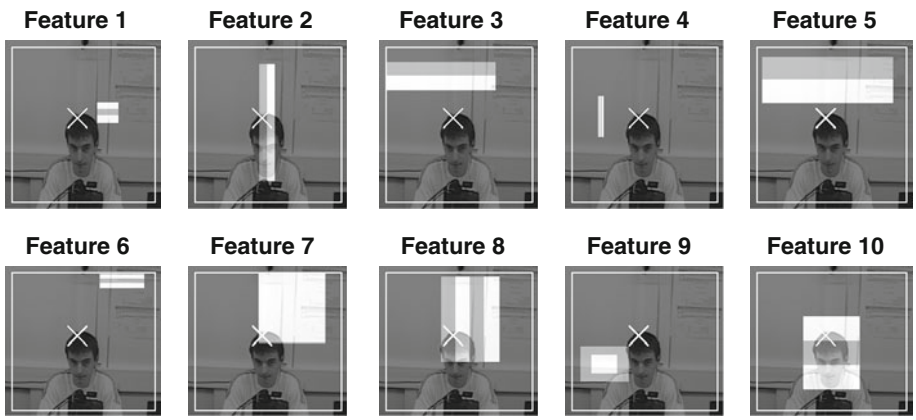

Fig. 6 The ten evolved features for the face-detection task, shown within the gaze window (*white box*) of the coarse-scale model. The *white cross* indicates the centre of the gaze window

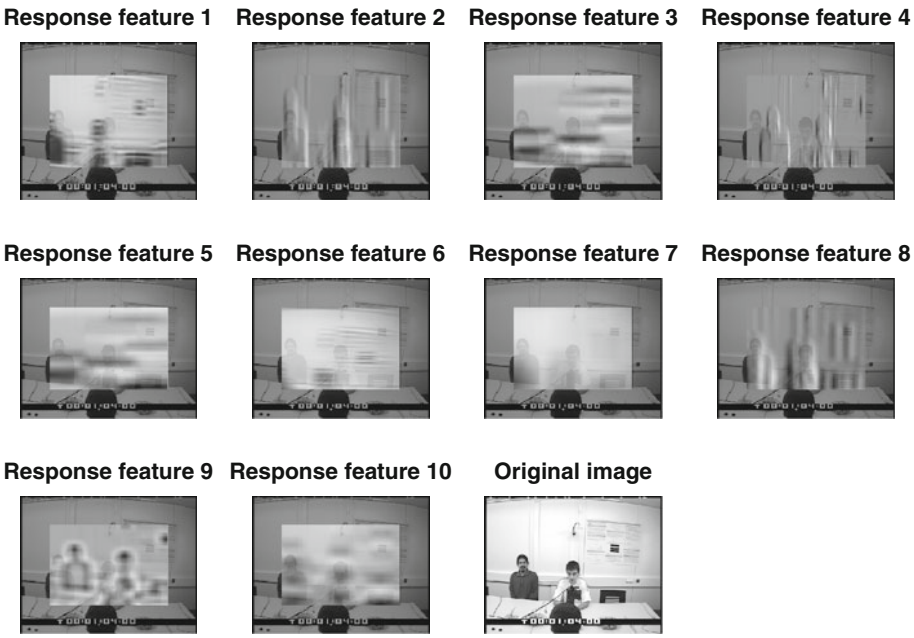

Fig. 7 Responses of the ten features shown in Fig. 6 in different parts of the image. The evolved features have been extracted at all possible gaze locations in an image (labelled 'original image'). Per feature, we show a darkened version of the image as the background, and the feature responses on the foreground. The feature responses are scaled to [0,1] and represented with *grey*-values. High intensity represents a high response, low intensity a low response. We show the response at the gaze location, i.e., the centre of the gaze window when the feature was extracted

Contextual and Object Features

Figures 6 and 7 illustrate that the model uses contextual features to find the object. Let us take feature 8 as an example. This feature represents vertical contrasts to the top right of the current gaze location (Fig. 6). We can see in Fig. 7 that this feature has a high response when the gaze location is on the right part of a person's body. This can be used by ACT-DETECT to locate the face.[6] Besides contextual features, ACT-DETECT also exploits features that seem to be more object-related. A telling example is feature 9, a centre-surround feature. Figure 7 shows that the high responses of the feature appear like ghost silhouettes shifted upwards and to the right with respect to the two persons. The shift reflects the fact that feature 9 (see Fig. 6) is located to the bottom left of the gaze location. There are only a few locations in Fig. 7 where the response of feature 9 is low (the dark regions), including those locations in which the gaze window is situated above and to the right of a head.

Information in the Visual Inputs

Apparently, ACT-DETECT exploits the visual features as detected at non-object locations to estimate the location of the sought-for object. To show that the evolved visual features indeed carry information about the location of an object, we determined the relation between visual samples on the one hand, and the corresponding horizontal and vertical displacements to an object on the other hand. Local input samples were gathered by extracting the evolved features at 30 random gaze locations for each image in a set of 145 training images (leading to 4,350 input samples) and storing the relative distance from each gaze location to the closest object.

The information in the input samples about the required displacements is determined using the mutual information measure. To facilitate the calculation of this measure, the continuous-valued input feature vectors are transformed to a discrete set by using k-means clustering. In this manner, the input features can be modelled as a discrete variable C with cardinality $|C| = k$. After clustering, every sample was mapped to the nearest cluster centroid and the relative distance to the target was stored for that cluster. This allowed us to determine the mutual information I between the clusters C and the horizontal displacement ΔX and the vertical displacement ΔY to the closest target: $I(C;\Delta X)$ and $I(C;\Delta Y)$. The values of ΔX and ΔY were discretised by binning them in 10 evenly spaced bins, ranging from their minimum to their maximum value. The mutual information, expressed in bits, could then be calculated according to the formula:

$$I(C;\Delta X) = H(\Delta X) - H(\Delta X \mid C), \tag{4}$$

where H is the Shannon entropy [71]. The variables involving ΔY are calculated with analogous formulas. As can be seen from Eq. 4, I is a symmetric measure. In the current context, it expresses how much information is gained about the relative object location in X- or Y-

[6] Other examples of contextual features are feature 4, which has a high response when ACT-DETECT is right of a person's body, and feature 7, which has a higher response on the wall than further down in the image where the persons are seated.

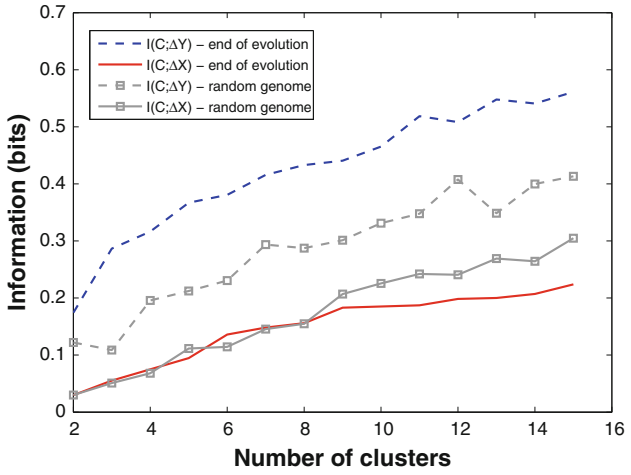

Fig. 8 The relation between the number of clusters and the mutual information of the clustering with the horizontal distance (ΔX, solid line) and vertical distance (ΔY, dashed line) to the closest object. The mutual information is expressed in bits and shown for the evolved instance of ACT-DETECT (*red* and *blue* lines) and an instance of ACT-DETECT with a random genome (*grey lines* with *square* markers). (Color figure online)

direction, if it is known to which cluster a local sample belongs. ACT-DETECT uses a neural network to map input feature vectors to actions. The network does not divide the input space into discrete subspaces, but maps the inputs to displacements with a continuous function. Since it is unknown which number of clusters best approximates the way in which ACT-DETECT interprets the input space, various numbers of clusters were used: $k = \langle 2, 3, 4, \ldots, 15 \rangle$.

Figure 8 shows the calculated mutual information for the different numbers of clusters. The mutual information is shown for the evolved instance of ACT-DETECT (red and blue lines) and an instance of ACT-DETECT with a random genome (grey lines with square markers). The mutual-information results lead to three observations. The first observation is that the inputs seem to have more information on the vertical distance than on the horizontal distance to the closest target object ($I(C;\Delta Y) > I(C;\Delta X)$). This is likely due to the fact that the visual environment in the image set is more organised in the vertical direction than in the horizontal direction. The second observation is that the evolved visual features capture more information on ΔY than the features corresponding to a random genome, while the information of ΔX is comparable. Although the information of only one random genome is shown in Fig. 8, this finding is valid for most of the random genomes. More information in the visual features can lead to a higher performance on the detection task. The third observation is that the information of the clustering on ΔX and ΔY grows with an increasing number of clusters. With a further increase in the number of clusters, the mutual information will continue to grow up to the point where the number of clusters equals the number of samples. It is important to remark that (a) the neural network of ACT-DETECT is not able to make such a fine subdivision of the input space, and (b) that such a fine subdivision of the input space would have negative effects on the generalisation to unseen local samples.

The results of our information-theoretic analysis confirm that local image samples contain information about the global location of the target object. This explains how ACT-DETECT achieves its detection performance despite the limited number of visual samples considered.

Spatial Distribution Relative to Closest Object

To gain further insight into how typical image samples map onto appropriate gaze shifts, we visualise the relative spatial distribution of each cluster's inputs with respect to the closest object. Figure 9 shows these spatial distributions for $k = 6$ clusters (the order of the clusters is

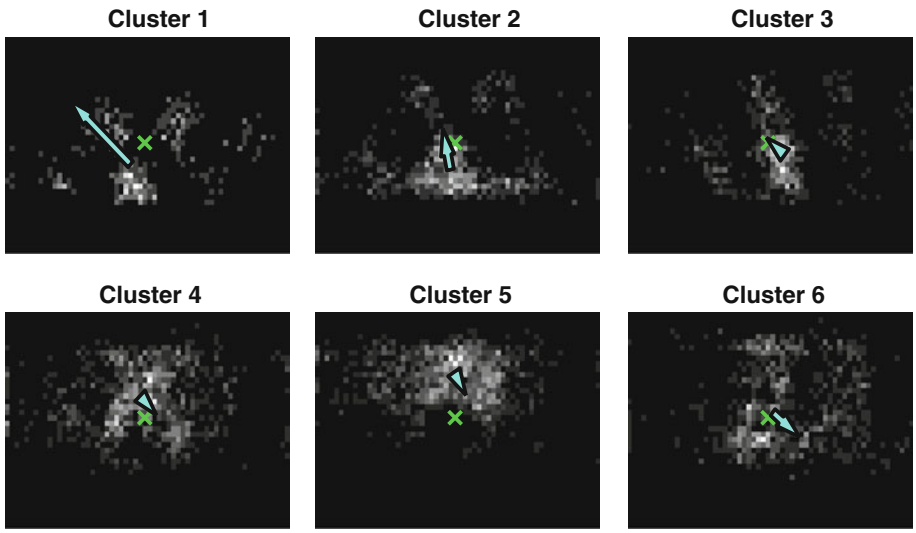

Fig. 9 Spatial distribution of six visual input clusters relative to the closest object. High occurrence is represented with high intensity, low occurrence with low intensity. The location of the closest object is in the centre of each inset (shown with a *green cross*). The *blue arrows* originate at the median of all locations in the cluster and represent the actions taken by ACT-DETECT if it receives the cluster centroid as visual input. (Color figure online)

irrelevant). We represent the closest target object with a green cross in the centre of every inset, and high intensity regions indicate regions where the input cluster has a high probability of occurring (low intensity regions indicate the opposite). The figure shows that the clusters of inputs have different relative spatial distributions with respect to the location of the target object.

Gaze Shifts

To show that ACT-DETECT employs a non-greedy gaze-shifting strategy, we now turn to the mapping of visual inputs to gaze shifts.

Figure 9 shows that ACT-DETECT *exploits the relative spatial distributions of the visual inputs, but does not follow a purely greedy strategy*. The blue arrows in the figure represent the direction and size of the gaze shifts taken by ACT-DETECT when it is provided with the cluster centroids as inputs. The arrows originate from the median relative location of the corresponding cluster of sensory inputs. Clearly, the actions depend on the relative spatial distributions of the sensory input clusters. However, ACT-DETECT does not seem to apply a greedy action policy: the ideal greedy action would shift the gaze from any location directly to the object location. For some clusters, the gaze shifts deviate from the ideal greedy action. The most remarkable deviation is that of cluster 1 for which ACT-DETECT makes a large gaze shift to the top left. This shift seems to make sense in the context of the rest of ACT-DETECT's behaviour: for the clusters above the object locations it always goes slightly more to the right.

Figure 10 shows the actions of the coarse-scale model in two different images. The arrows in the figure represent ACT-DETECT's gaze shifts at a 12 × 11 grid in the images. Most of the time ACT-DETECT makes large movements to the top left for visual inputs that occur below the faces and smaller movements to the bottom right for those that occur above the faces. This behaviour attempts to bring ACT-DETECT close to a face, where it gets caught into a local behavioural attractor. Please remark that the non-greedy behaviour of ACT-DETECT is due to the fitness function used in evolution: the fitness only depends on the locations of ACT-DETECT at the end of the run.

The fine-scale model always starts at the final locations of the coarse-scale model. It exploits object context at a finer scale. Figure 11 shows the actions of the fine-scale model on a grid of 20 × 21 in the image. From this figure, it is clear that ACT-DETECT can approach objects on a finer scale, but that it works best for locations close to the objects. The arrows in the figure typically point upwards for locations above the faces in the image. This implies that if the fine-scale model started at such a location, it would end up at the top border of the image. It also leads to the

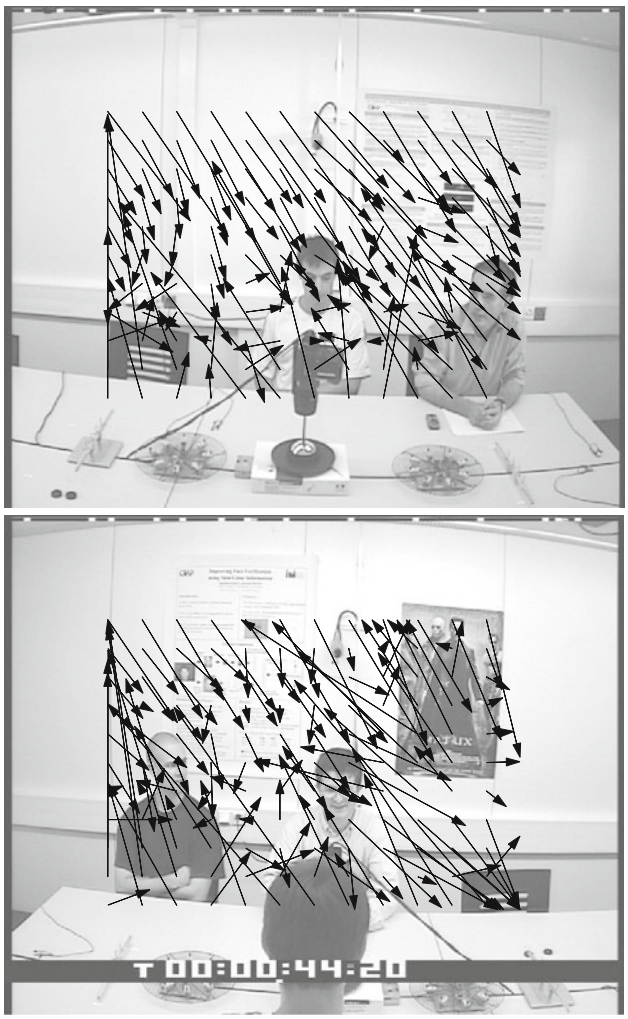

Fig. 10 Actions taken by the coarse-scale model on a 12 × 11 grid in two example images

observation that the fine-scale model generally moves slightly up. This is an indication that the coarse-scale model has a tendency to end up slightly below the faces in the image. Since the fine-scale model is evolved for starting its gaze behaviour at the final locations of the coarse-scale model, it compensates for possible small deviations between these locations and the object locations.

Computational Efficiency

The main benefit of applying ACT-DETECT to object detection is its computational efficiency. In this section, we first make a general comparison of the computational effort of ACT-DETECT and that of window-sliding methods (section "Window-Sliding Methods"). Then, we make a tentative comparison with recently developed methods, which significantly reduced the computational effort with respect to

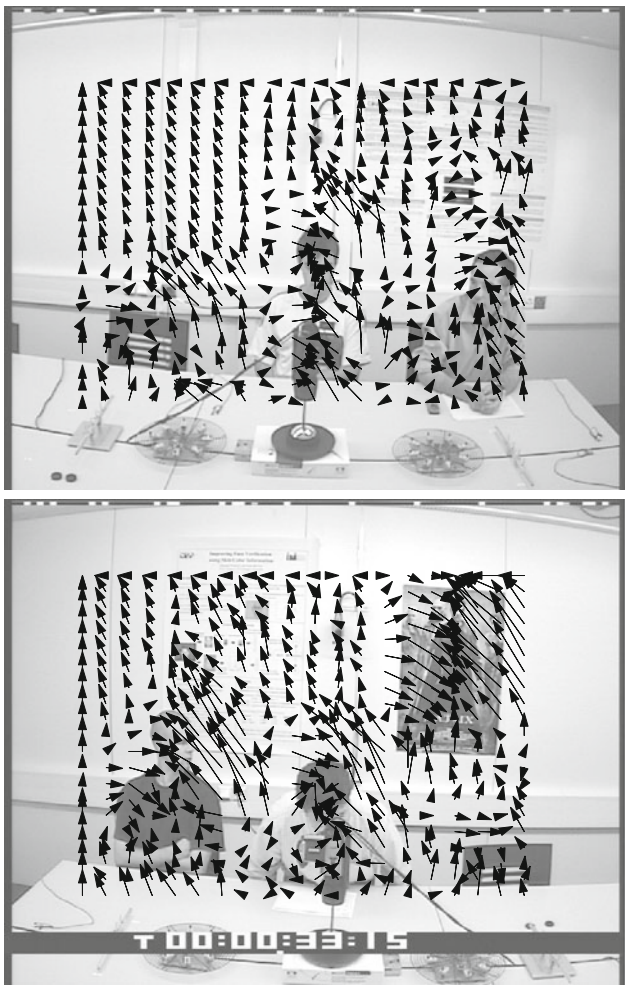

Fig. 11 Actions taken by the fine-scale model on a 20 × 21 grid in two example images

window-sliding methods (section "Efficient Subwindow Search").

Window-Sliding Methods

The comparison between ACT-DETECT and window-sliding methods is possible because both ACT-DETECT and the window-sliding methods extract features from a local input window. The computational costs C of a window-sliding method (WS) and of ACT-DETECT (AD) can be expressed as follows.

$$C_{\mathrm{WS}} = G_{\mathrm{H}} G_{\mathrm{V}} (F_{\mathrm{WS}} + Cl) + P \tag{5}$$

$$C_{\mathrm{AD}} = R(S \times T)(F_{\mathrm{AD}} + Ct) + R(F_{\mathrm{WS}} + Cl) + P \tag{6}$$

The variables G_{H} and G_{V} are the number of horizontal and vertical grid points, respectively. Furthermore, F_{WS} is the number of operations necessary for feature extraction in the window-sliding approach, Cl stands for the classifier, and P for preprocessing. For ACT-DETECT, R is the number of independent runs, S the number of scales, and T the number of local samples extracted per scale. F_{AD} is the number of operations necessary for feature extraction, and Ct for the controller that maps the features to gaze shifts. The cost of ACT-DETECT includes $R(F_{\mathrm{WS}} + Cl)$, since the presence of an object is verified at the final gaze location.

ACT-DETECT is computationally more efficient than the window-sliding method. The main reason for this is that ACT-DETECT extracts far fewer local samples, i.e., $(R(S \times T) + R) \ll G_{\mathrm{H}} G_{\mathrm{V}}$, while its feature extraction and controller do not take much more computational effort than the feature extraction and classifier of the window-sliding method. For example, in the FGNET-task a window-sliding method that verifies the presence of an object at every point of a grid with a step size of two pixels will extract $335 \times 248 = 83{,}080$ local samples (based on the image size, the average face size of 50×80 pixels, and the largest step size mentioned in [78]). In contrast, in the case of $R = 20$, $S = 2$, and $T = 5$, ACT-DETECT extracts $R(S \times T + 1) = 20 \times 11 = 220$ local samples. Under these conditions, the window-sliding method extracts $83{,}080/220 \approx 378$ times more local samples than ACT-DETECT.

Further speed-ups can be attained by reducing the number of time steps used by ACT-DETECT. The best results will be obtained by evolving ACT-DETECT for a smaller number of time steps T, but here we apply the instance of ACT-DETECT evolved for $T = 5$ to other numbers of time steps. Figure 12 shows the FROC curves when it is applied to $T = 7$ (red solid line), $T = 5$ (blue dotted line), $T = 2$ (green dashed line), and $T = 1$ (black dashed-dotted line). Please remark that $T = 1$ means that ACT-DETECT makes one

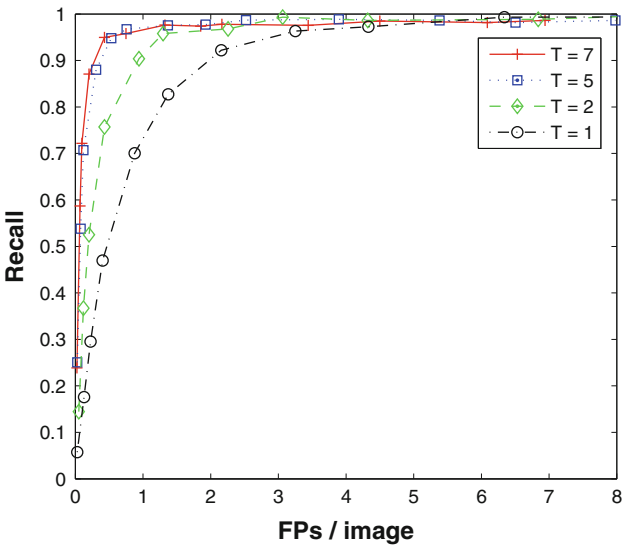

Fig. 12 FROC curves for $T = 7$ (*red solid line*), $T = 5$ (*blue dotted line*), $T = 2$ (*green dashed line*), and $T = 1$ (*black dashed-dotted line*). (Color figure online)

shift with the coarse-scale model, one shift with the fine-scale model, and a final check with the object classifier. With $R = 20$, this adds up to 60 samples extracted from the image, resulting in a recall of 70% with on average 0.88 false positives per image. The results in the figure show that ACT-DETECT's performance degrades gracefully with a decreasing number of time steps.

Please note that for the comparison between ACT-DETECT and window-sliding methods we did not take object detection at different scales into account. The number of scales at which an object can occur would imply a new multiplication factor for the computational costs, which is most disadvantageous for the window-sliding method. The straightforward way to detect objects at multiple scales for ACT-DETECT is to apply the object classifier at the final step on different scales. The question then becomes whether the same visual features and behaviours of the coarse-scale and fine-scale model are evenly adequate for objects of different scales. The matter of the detection of differently scaled objects is left for future work.

Efficient Subwindow Search

Some recent studies have already considerably improved the computational efficiency over window-sliding methods. For example, as mentioned in the introduction, in [40] a branch-and-bound scheme, Efficient Subwindow Search (ESS), is used to efficiently maximise a classifier function over all subregions in an image. The branch-and-bound scheme does not deteriorate the performance with respect to a window-sliding method, but dramatically reduces the number of function evaluations necessary in an image. The authors mention that for image sizes in the range of 500×333 to 640×480, ESS performs on average less than 20,000 evaluations of the bound function per image. A direct comparison of this method's computational effort with ACT-DETECT is difficult, since on the one hand ESS' preprocessing seems more expensive, while on the other hand ESS already evaluates objects at different scales. Still, we can make a tentative comparison based on the assumption that ACT-DETECT's object classifier will be applied at $N = 5$ different scales. This would lead to the extraction of $R((S \times T) + N) = 20 \times (10 + 5) = 300$ samples. This number is $\sim$67 times smaller than that of the branch-and-bound method (ignoring any differences in preprocessing).

Discussion and Conclusion

Our study revealed that the exploitation of an object's visual context enhances the computational efficiency of detecting the object considerably. The implication is that current object-detection methods could be improved in terms of efficiency by adopting the local-context guided sampling of ACT-DETECT. The extent to which this is true depends on the generalisability of the results, in particular the degree to which local-context guided sampling can be successful on other tasks than the FGNET face-detection task. The success on other tasks depends mainly on the presence and reliability of the visual cues at non-object locations to the object location. In the FGNET task, the brightness and absence of texture on the walls can be used as reliable visual cues for the face-detection task: the model should typically shift its gaze downwards. In other tasks, visual cues may be quite different but still informative. We expect that most object-detection tasks of interest have visual cues that can be exploited by ACT-DETECT. Further experiments are needed to bear out this expectation.

We conclude that object-detection algorithms can exploit local visual context to improve their efficiency and that the success of such algorithms depends on (1) the presence of visual cues at non-object locations to the object location, and (2) the capability of the gaze-control model to exploit these cues.

Future work will address the generalisability of the gaze-control model to a wide variety of detection tasks. These tasks will have a larger variation both with respect to the environment as to the object types and scales. In addition, it will be investigated whether employing a gaze-control model such as ACT-DETECT can reduce the need for large training data sets as required by classification-based methods. While one object in one image only gives one positive example for a classification-based method (and many negative ones), it provides a rich training set for a gaze-control model, consisting of the visual feature values at all image locations.

Acknowledgments This research was carried out within the ToKeN VindIT project (grant number 634.000.018), funded by the Netherlands Organisation for Scientific Research (NWO). Furthermore, we thank Laurens van der Maaten for commenting on earlier versions of this article. Finally, we thank the reviewers for their insightful comments.

References

1. Bäck T. Evolutionary algorithms in theory and practice. Oxford: Oxford University Press; 1996.
2. Ballard DH. Animate vision. Artif Intell. 1991;48(1):57–86.
3. Bandera C, Vico FJ, Bravo JM, Harmon ME, Barid LC III. Residual q-learning applied to visual attention. In: Saitta L,

editor. 13th International conference on machine learning (ICML 1996). Italy: Bari; 1996. p. 20–7.
4. Bergboer NH, Postma EO, van den Herik HJ. A context-based model of attention. In: López de Mánteras R, Saitta L, editors. 16th European conference on artificial intelligence (ECAI 2004). Valencia, Spain: IOS Press, 2004. p. 927–31.
5. Biederman I, Glass AL, Stacy WJ. Searching for objects in real-world scenes. J Exp Psychol. 1973;97(1):22–7.
6. Biederman I, Mezzanotte RJ, Rabinowitz JC. Scene perception: detecting and judging objects undergoing relational violations. Cogn Psychol. 1982;l4(2):l43–77.
7. Boccignone G, Ferraro M. Modelling gaze shift as a constrained random walk. Phys. A. 2004;331(1):207–18.
8. Brockmann D, Geisel T. The ecology of gaze shifts. Neurocomputing. 2000;32–33:643–50.
9. Burges C. A tutorial on support vector machines for pattern recognition. Data Min Knowl Discov. 1998;2(2):121–67.
10. Burl MC, Weber M, Perona P. A probabilistic approach to object recognition using local photometry and global geometry. In: Burkhardt H, Neumann B, editors. 5th European conference on computer vision (ECCV 1998), Freiburg. vol 2. Springer; 1998. p. 628–41.
11. Carbonetto P, Dorkó G, Schmid C, Kück H, de Freitas N. Learning to recognize objects with little supervision. Int J Comput Vis. 2008;77(1–3):219–37.
12. Chikkerur S, Tan C, Serre T, Poggio T. An integrated model of visual attention using shape-based features. 2009. Tech. rep., MIT CSAIL, CBCL-278.
13. Chung MM, Jiang Y. Contextual cueing: implicit learning and memory of visual context guides spatial attention. Cogn Psychol. 1998; 36(1):28–71.
14. Cristinacce D, Cootes T. A comparison of two real-time face detection methods. In: 4th IEEE international workshop on performance evaluation of tracking and surveillance, Graz. 2003. p. 1–8.
15. de Croon G. Active object detection. In: Ranchordas A, Araújo H, Vitrià J, editors. 2nd International conference on computer vision theory and applications (VISAPP 2007), Barcelona, Institute for Systems and Technologies of Information, Control and Communication (INSTICC). 2007. p. 97–103.
16. de Croon G, Postma E. Sensory-motor coordination in object detection. In: Abbass H, editor. 1st IEEE symposium on artificial life, Honolulu. 2007. p. 147–54.
17. de Croon G, Postma EO. Active object detection. In: Schobbens PY, Vanhoof W, Schwanen G, editors, Belgian–Dutch AI Conference (BNAIC 2006). Belgium: Namur; 2006. p. 107–14.
18. de Graef PC, Christiaens D, d'Ydevalle G. Perceptual effects of scene context on object identification. Psychol Res. 1990; 52(4):317–29.
19. de Wagter C, Mulder J. Towards vision-based uav situation awareness. In: AIAA guidance, navigation, and control conference, San Francisco. 2005. AIAA–2005–5872.
20. Dorkó G, Schmid C. Selection of scale-invariant parts for object class recognition. In: 9th IEEE international conference on computer vision (ICCV 2003). vol 1. Los Alamitos: IEEE Computer Society; 2003. p. 634–9.
21. Elder J, Prince S, Hou Y, Sizintsev M, Olevskiy E. Pre-attentive and attentive detection of humans in wide-field scenes. Int J Comput Vis. 2007;72(1):47–66.
22. Fergus R, Perona P, Zisserman A. Object class recognition by unsupervised scale-invariant learning. In: Computer vision and pattern recognition (CVPR 2003). vol 2. Madison: IEEE Computer Society; 2003. p. 264–71.
23. Fergus R, Perona P, Zisserman A. Weakly supervised scale-invariant learning of models for visual recognition. Int J Comput Vis. 2007;71(3):273–303.
24. Floreano D, Kato T, Marocco D, Sauser E. Coevolution of active vision and feature selection. Biol Cybern. 2004;90(3):218–28.
25. Fröba B, Küllbeck C. Robust face detection at video frame rate based on edge orientation features. In: 5th IEEE international conference on automatic face and gesture recognition 2002. Los Alamitos: IEEE Computer Society; 2002. p. 342–7.
26. Henderson JM, Weeks PA Jr, Hollingworth A. Effects of semantic consistency on eye movements during scene viewing. J Exp Psychol Hum Percept Perform. 1999;25(1):210–28.
27. Hoiem D, Efros AA, Hebert M. Geometric context from a single image. In: Ma S, Shum HY, editors, 10th IEEE international conference on computer vision (ICCV 2005), Bejing. vol 1. Washington: IEEE Computer Society; 2005. p. 654–61.
28. Hoiem D, Efros AA, Hebert M. Putting objects in perspective. In: IEEE conference on computer vision and pattern recognition (CVPR 2006), New York. vol 2. Los Alamitos: IEEE Computer Society; 2006. p. 2137–44.
29. Holub AD, Welling M, Perona P. Combining generative models and fisher kernels for object recognition. In: Ma S, Shum HY, editors, 10th IEEE international conference on computer vision (ICCV 2005), Bejing. Washington: IEEE Computer Society; 2005. p. 136–43.
30. Itti L, Koch C, Niebur E. A model of saliency-based visual attention for rapid scene analysis. IEEE Trans Pattern Anal Mach Intell. 1998;20(11):1254–9.
31. Kadir T, Brady M. Scale, saliency and image description. Int J Comput Vis. 2001;45(2):83–105.
32. Kato T, Floreano D. An evolutionary active-vision system. In: Congress on evolutionary computation (CEC 2001), Seoul. vol 1. IEEE Computer Society; 2001. p. 107–14.
33. Kim TK, Kittler J. Composite support vector machines with extended discriminative features for accurate face detection. IEICE Trans Inf Syst 2005;E88-D(10):2373–9.
34. Klarquist WN, Bovik AC. Fovea: a foveated vergent active stereo vision system for dynamic three-dimensional scene recovery. IEEE Trans Robot Autom. 1998;14(5):755–70.
35. Kruppa H, Castrillon-Santana M, Schiele B. Fast and robust face finding via local context. In: Joint IEEE international workshop on visual surveillance and performance evaluation of tracking and surveillance (VS-PETS 2003), Nice. IEEE Computer Society; 2003. p. 157–64.
36. Lacroix J, Postma E, van den Herik H. Modeling visual classification using bottom-up and top-down fixation selection. In: 29th Annual conference of the cognitive science society (CogSci 2007), Nashville; 2007. p. 419–24.
37. Lambert A, Duddy M. Visual orienting with central and peripheral precues: deconfounding the contributions of cue eccentricity, cue discrimination and spatial correspondence. Vis Cogn. 2002;9:303–36.
38. Lambert A, Roser M, Wells I, Heffer C. The spatial correspondence hypothesis and orienting in response to central and peripheral spatial cues. Vis Cogn. 2002;13:65–88.
39. Lampert C. An efficient divide-and-conquer cascade for nonlinear object detection. In: IEEE computer society conference on computer vision and pattern recognition; 2010.
40. Lampert C, Blaschko M, Hofmann T. Beyond sliding windows: object localization by efficient subwindow search. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR 2008). IEEE Computer Society; 2008. p. 1–8.
41. Leibe B, Leonardis A, Schiele B. Robust object detection with interleaved categorization and segmentation. Int J Comput Vis Special Issue Learn Recognit Recognit Learn. 2008;77(1–3): 259–89.
42. Li SZ, Zhang Z. Floatboost learning and statistical face detection. IEEE Trans Pattern Anal Mach Intell 2004;26(9):1112–23.

43. Loeff N, Arora H, Sorokin A, Forsyth D. Efficient unsupervised learning for localization and detection in object categories. In: Weiss Y, Schölkopf B, Platt J, editors, Advances in neural information processing systems (NIPS 2005). Vancouver: MIT Press; 2005. p. 811–8.
44. Lowe DG. Distinctive image features from scale-invariant keypoints. Int J Comput Vis. 2004;60(2):91–110.
45. Macinnes I, Di Paolo E. The advantages of evolving perceptual cues. Adapt Behav. 2006;14(2):147–56.
46. Marocco D, Floreano D. Active vision and feature selection in evolutionary behavioral systems. In: 7th International conference on simulation of adaptive behavior, from animals to animats (SAB 2002), Edinburgh. Cambridge: MIT Press; 2002. p. 247–55.
47. Mikolajczyk K, Leibe B, Schiele B. Multiple object class detection with a generative model. In: IEEE conference on computer vision and pattern recognition (CVPR'06), New York. IEEE Computer Society; 2006. p. 26–36.
48. Minut S, Mahadevan S. A reinforcement learning model of selective visual attention. In: 5th International conference on autonomous agents (Agents 2001), Montreal. New York: ACM Press; 2001. p. 457–64.
49. Murphy K, Torralba A, Eaton D, Freeman W. Object detection and localization using local and global features. In: Ponce J, Hebert M, Schmid C, Zisserman A, editors, Toward category-level object recognition (Sicily workshop on object recognition). Berlin: Springer; 2006. p. 382–400.
50. Neider MB, Zelinski GJ. Scene context guides eye movements during visual search. Vis Res. 2006;46(5):614–21.
51. Nolfi S, Floreano D. Evolutionary robotics: the biology, intelligence, and technology of self-organizing machines. Cambridge: MIT Press/Bradford Books; 2000.
52. Oliva A, Torralba A, Castelhano MS, Henderson JM. Top-down control of visual attention in object detection. In: 10th IEEE international conference on image processing (ICIP 2003), Barcelona. vol 1. IEEE Computer Society; 2003. p. 253–6.
53. Osuna E, Freund R, Girosi F Training support vector machines: an application to face detection. In: Computer vision and pattern recognition (CVPR 1997), San Juan. Los Alamitos: IEEE Computer Society; 1997. p. 130–6.
54. Paletta L, Fritz G, Seifert C. Q-learning of sequential attention for visual object recognition from informative local descriptors. In: de Raedt L, Wrobel S, editors, 22nd International conference on machine learning (ICML 2005), Bonn. ACM Press; 2005. p. 649–56.
55. Palmer ST. The effects of contextual scenes on the identification of objects. Mem Cogn. 1975;3(5):519–26.
56. Perko R, Leonardis A. Learning visual context for object detection. In: 16th IEEE electrotechnical and computer science conference (ERK 2007), Portorož. vol B. IEEE Computer Society; 2007. p. 163–6.
57. Peters RJ, Itti L. Beyond bottom-up: incorporating task-dependent influences into a computational model of spatial attention. In: 12th IEEE computer society conference on computer vision and pattern recognition (CVPR 2007), Minneapolis. IEEE Computer Society; 2007. p. 1–8.
58. Platt J. Using sparseness and analytic QP to speed training of support vector machines. In: Kearns M, Solla S, Cohn D, editors, Advances in neural information processing systems (NIPS 1999). vol 11. Denver: MIT Press; 1999 p. 557–63.
59. Privitera CM, Stark LW. Algorithms for defining visual regions-of-interest: comparison with eye fixations. IEEE Trans Pattern Anal Mach Intell. 2000;22(9):970–82.
60. Qi Y, Doermann D, DeMenthon D. Hybrid independent component analysis and support vector machine learning scheme for face detection. In: IEEE international conference on acoustics, speech, and signal processing (ICASSP 2001), Salt Lake City. vol 3. Los Alamitos: IEEE Computer Society; 2001. p. 1481–4.
61. Rajashekar U, Cormack L, Bovik A. Image features that draw fixations. In: 10th IEEE international conference on image processing (ICIP 2003), Barcelona. vol 3. IEEE Computer Society; 2003. p. 313–6.
62. Rao RPN, Zelinsky GJ, Hayhoe MM, Ballard DH. Modeling saccadic targeting in visual search. In: Touretzky D, Mozer M, Hasselmo M, editors, Advances in neural information processing systems (NIPS 1995). vol 8. MIT Press; 1995. p. 830–6.
63. Rao RPN, Zelinsky GJ, Hayhoe MM, Ballard DH. Eye movements in visual cognition: a computational study. Tech. Rep. 97.1, Department of Computer Science, University of Rochester; 1997.
64. Rasolzadeh B, Björkman M, Hayman E, Eklundh J. An attentional system combining top-down and bottom-up influences. In: International cognitive vision workshop, in conjunction with ECCV 2006, Graz; 2006.
65. Roberts JF, Stirling T, Zufferey JC, Floreano D. Quadrotor using minimal sensing for autonomous indoor flight. In: 3rd US–European competition and workshop on micro air vehicle systems (EMAV 2007), Toulouse; 2007.
66. Rowley HA, Baluja S, Kanade T. Neural network-based face detection. IEEE Trans Pattern Anal Mach Intell 1998;20(1):23–38.
67. Russakovsky O, Ng A. A steiner tree approach to efficient object detection. In: IEEE computer society conference on computer vision and pattern recognition; 2010.
68. Rybak I, Gusakova V, Golovan A, Podladchikova L, Shevtsova N. A model of attention-guided visual perception and recognition. Vis Res 1998;38(15–16):2387–400.
69. Schmidhuber J Reinforcement learning in markovian and non-markovian environments. In: Lippman D, Moody J, Touretzky D, editors, Advances in neural information processing systems (NIPS 1990). Denver: Morgan Kaufmann; 1990. p. 500–6.
70. Schneiderman H, Kanade T. A statistical approach to 3d object detection applied to faces and cars. In: IEEE conference on computer vision and pattern recognition (CVPR 2000). vol 1. Hilton Head Island: IEEE Computer Society; 2000. p. 746–51.
71. Shannon C. A mathematical theory of communication. Bell Syst Tech J. 1948;27:379–423, 623–56.
72. Smeraldi F, Capdevielle N, Bigün J. Facial features detection by saccadic exploration of the gabor decomposition and support vector machines. In: 11th Scandinavian conference on image analysis (SCI 1999), Kangerlussuaq. vol 1; 1999. p. 39–44.
73. Sun Y, Fisher R. Object-based visual attention for computer vision. Artif Intell. 2003;146(1):77–123.
74. Sung K, Poggio T. Example-based learning for view-based human face detection. IEEE Trans Pattern Anal Mach Intell. 1998;20(1):39–51.
75. Terzopoulos D, Rabie T, Grzesczuk R. Perception and learning in artificial animals. In: 5th International conference on the synthesis and simulation of living systems (Artificial Life V), Nara. Cambridge: MIT Press; 1996. p. 346–53.
76. Torralba A. Contextual priming for object detection. Int J Comput Vis. 2003;53(2):169–91.
77. Torralba A, Oliva A, Castelhano M, Henderson JM. Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search. Psychol Rev. 2006;113(4):766–86.
78. Viola P, Jones MJ. Robust real-time object detection. Tech. rep., Cambridge Research Laboratory; 2001.
79. Vogel J, de Freitas N. Target-directed attention: sequential decision making for gaze planning. In: IEEE international conference on robotics and automation (ICRA 2008); 2008. p. 2372–9.

80. Weber M. Unsupervised learning of models for object recognition. PhD thesis. Pasadena: California Institute of Technology; 2000.
81. Weber M, Welling M, Perona P. Towards automatic discovery of object categories. In: IEEE conference on computer vision and pattern recognition (CVPR 2000). Hilton Head Island: IEEE Computer Society; 2000a. p. 101–9.
82. Weber M, Welling M, Perona P. Unsupervised learning of models for recognition. In: 6th European conference on computer vision (ECCV 2000), Dublin. vol 1. Springer; 2000b. p. 18–32.
83. Wolf L, Bileschi S. A critical view of context. Int J Comput Vis 2006;69(2):251–61.
84. Young SS, Scott PD, Bandera C. Foveal automatic target recognition using a multiresolution neural network. IEEE Trans Image Process. 1998;7(8):1122–35.