Attention-based Dynamic Visual Search Using Inner-Scene Similarity: Algorithms and Bounds

Tamar Avraham and Michael Lindenbaum

Computer Science Department, Technion - I.I.T.

Haifa, Israel 32000

tammya, mic@cs.technion.ac.il

June 2, 2005

Abstract

A visual search is required when applying a recognition process on a scene containing multiple objects. In such cases, we would like to avoid an exhaustive sequential search. This work proposes a dynamic visual search framework based mainly on inner-scene similarity. Given a number of candidates (e.g., sub-images), we hypothesize is that more visually similar candidates are more likely to have the same identity. We use this assumption for determining the order of attention. Both deterministic and stochastic approaches, relying on this hypothesis, are considered. Under the deterministic approach, we suggest a measure similar to Kolmogorov's epsilon-covering that quantifies the difficulty of a search task. We show that this measure bounds the performance of all search algorithms and suggest a simple algorithm that meets this bound. Under the stochastic approach, we model the identity of the candidates as a set of correlated random variables and derive a search procedure based on linear estimation. Several experiments are presented in which the statistical characteristics, search algorithm, and bound are evaluated and verified.

Contents

1 Introduction **4**

2 Related Work **6**

3 Framework **9**

3.1 The context – visual search involving candidate selection and classification . . 9

3.2 Sources of information for directing the search 9

3.3 Models of inner-scene visual similarity 10

3.4 Dynamic vs. Static Search 10

3.5 Algorithms 11

3.6 Measures of performance 11

4 Deterministic Analysis of Visual Search Performance **12**

4.1 Introduction 12

4.2 Notations 12

4.3 A measure for search difficulty combining target isolation and candidate scattering 13

4.4 Lower-bounds on search algorithms 14

4.4.1 No knowledge 15

4.4.2 Minimal knowledge: $d_T > d_0$ and the number of isolated points 15

4.4.3 Bounded metric space 15

4.4.4 Knowledge: $d_T > d_0$ and $c_{d_0}(X)$ 16

4.5 A simple search algorithm and its upper-bounds 16

4.5.1 Simple search algorithms 16

4.5.2 Bounded metric space 17

4.5.3 $c_{d_0}(X)$ as an upper bound for search cost 18

4.6 Illustration of the above results 18

5 Dynamic Stochastic Search Algorithm **19**

5.1 The need to extend FLNN 19

5.2 Object labels as dependent random variables 19

5.3 Dynamic search framework 20
5.4 Minimum mean square error linear estimation 20
5.5 The VSLE algorithm 21
5.6 Combining prior information 22

6 Experiments **22**

6.1 FLNN and Minimum-Cover-Size 23
6.2 VSLE and Covariance Characteristics 24
6.3 VSLE applied to MIT+CMU database 29
6.4 Combining VSLE and VJA 29

7 Discussion **31**

References **34**

A Appendix - Implementation details **38**

1 Introduction

Visual searching means either a human or a machine viewing a scene with the goal of finding one or more familiar entities. The highly effective attention mechanisms in the human visual system (HVS) were extensively studied, in the context of visual search tasks, from psychophysics and physiology points of view. The most dominant model of human-visual-search is Treisman and Gelade's *feature integration theory* [32]. This model suggests (following Neisser [20]) that the search process is divided into a pre-attentive stage during which basic features are extracted, and a focal attention stage during which attention is drawn to specific regions one at a time. Search tasks were divided to easier *pop-out* tasks, associated with salient targets having unique properties (relative to the rest of the image), and harder serial search tasks. Another model, proposed by Duncan and Humphreys [7], suggests that attention is not drawn to locations but rather to image objects, and that search task efficiency depends on similarities between scene objects and possible target models and between objects within the scene.

It is widely believed that search task difficulty depends on all three characteristics: saliency (bottom-up information), similarity to target models (top-down information), and similarities between the scene objects. For more details on related work, see Section 2.

Several models for attention mechanisms and visual search have been implemented. They usually follow the HVS models (e.g. [11]), but sometimes use other sources of information such as context [25, 38, 18]. Several successful search mechanisms relying (mostly) on top-down information were recently suggested as detection algorithms (e.g., [16, 36, 1, 31]). Complex local descriptors (e.g. SIFT [16]), for example, can accurately characterize specific object parts and can serve as efficient indices for finding them in an image. Another option for accelerating the search is to use rejection based classifiers, which spend an adaptive amount of computational effort on different candidate sub-images, and can reject many non-target candidates very fast [1, 36]. In particular, the rejection cascade of AdaBoost classifiers combined with efficient Haar based components, lead to an extremely fast procedure for finding objects [36]. It seems, however, that the computational advantage is possible only when the visual appearance of the sought for object is not too complex or variable. Otherwise, a simple classifier, enabling fast rejection, is not accurate. Moreover, when there is interest in several

different objects, much of the procedure has to be repeated for each target type, slowing down the search.

This study aims to develop a general framework for computer visual search. Specifically, our objective is to find the target(s) as fast as possible, in minimal expected time. We chose to focus on inner-scene similarity as the source of information for directing the search process, because other information sources have already been intensively studied. We show that inner-scene similarity by itself can yield a fast and principled search, and also suggest ways to augment it with the other sources of information.

Our basic assumption is: more similar candidates tend to have more similar identities. Given a scene image, we propose to extract a number of candidate sub-images. The algorithms we consider begin with an initial priority map, indicating the initial likelihood of each candidate to be a target. Iteratively, the candidate with the highest priority receives the attention. The relevant sub-image is examined by a high-level object recognition system. Based on the recognizer's responses and the dependencies among candidates, the priority map is dynamically updated. We have two major goals in our research: The first is to provide an algorithm that will improve the trivial performance of an arbitrary order based search. The second is to provide measures for ease of search. In other words, given an input image (or a set of candidates), provide a grade describing how hard it is to locate the targets in it.

To illustrate the above principles as expressed in our suggested visual search algorithms, let us look at the following simple example: seeking people in a natural scene that includes people and trees. A segmentation process will provide us with a mixed set of candidates that are either trees or people. Say we start by querying the recognition system about one of the tree candidates and get a 'no' answer. We will give higher priority to the candidates that are most different from that tree – the people candidates. Hence, we will examine all the people before considering another tree.

We take both deterministic and stochastic approaches. Under the deterministic approach, we analytically suggest bounds for the performance of all search algorithms and suggest a simple algorithm (denoted FLNN - Farthest Labeled Nearest Neighbor) that meets these bounds. The main bound characterizes the search tasks using a measure similar to Kolomgorov's ϵ-

covering [14]. Under the stochastic approach, we model the identity of the candidates as a set of correlated random variables taking target/non-target values and characterize the task using its second order statistics. Experimentally, we show that the correlation between the candidate identities is commonly a monotone descending function of their feature-space-distances (used as a measure for similarity). We propose a linear estimation based search algorithm, denoted VSLE (Visual Search by Linear Estimation), which uses similarity information, and can also be extended to use bottom-up and top-down information, when available.

In the computer vision field visual search is usually considered under the wider topic of *Active Vision* [30]. Our algorithm can be used in a camera controlled active system in which the camera first acquires a low resolution image of the full scene. This image is used for candidate selection, feature extraction and attention decisions. Each time the algorithm decides on the next candidate to be examined, the camera is directed to the candidate and zooms in, acquiring a high resolution image of it, and possibly sends this enhanced information to a recognition module. The importance of a good dynamic candidate ordering, aiming at reducing such superfluous actions, is obvious; see [6, 18] for related work.

We start with a survey of related work (Section 2). The context for visual search and some basic intuitive assumptions are described in Section 3. Section 4 takes a deterministic approach to the quantitative analysis of search tasks, and presents bounds and algorithms that achieve them. Section 5 describes the stochastic approach, and contains the underlying probabilistic model and the resulting VSLE algorithm. In the experiments section we first show he relation between the algorithms' performance and the tasks' difficulty, and continue by demonstrating the stochastic algorithm's effectiveness, both by itself and in combination with Viola and Jones algorithm (denoted VJA) [36].

2 Related Work

Visual attention mechanisms have been widely studied from the psychophysics and physiology points of view, in an effort to try and understand the way nature deals with the visual search task. Yarbus [41] studied human perception of stationary images by tracking eye movements. He described several types of eye movements and found that the eyes rest much longer on the elements of an image that seem to carry essential information, while other elements may

receive little or no attention; e.g., when looking at a human face, an observer usually pays most attention to the eyes, lips, and noise.

There is much evidence that, in addition to the eye movements (referred to as the *overt* process), the attention mechanism focuses on different image parts without moving the eyes (a *covert* process). Posner et al. [23] suggested that this covert attentional mechanism acts as a "spotlight" moving about the scene. The spotlight can be focused on a small area or diffusely spread over a large area but the total attention capacity is limited.

Neisser [20] suggested that visual processing is divided into pre-attentive and attentive stages. The first consists of parallel processes that simultaneously operate on large portions of the visual field, and form the units to which attention may then be directed. The second stage consists of limited-capacity processes that focus on a smaller portion of the visual field.

Treisman and Gelade's *feature integration theory* [32] suggests that some visual properties (such as color, orientation, spatial frequency, movement) are extracted early, automatically and in parallel across the visual field. A saliency measure, telling how much a location is different from its surrounding is calculated for every feature, and the saliency maps are added, creating a *master map*. An object, which is unique relative to its surroundings in some feature, becomes salient and essentially *pops-out* without the need for focal attention. If there is no feature that makes the target unique in its surroundings, e.g. when the target is unique only in a conjunction of such features, the distinction cannot be detected pre-attentively. This forces subjects to search serially through the image, until the attention *spotlight* focuses on the target, which enables the usage of integrated features (conjunctions) for recognition.

While several aspects of the feature integration theory were criticized, the theory was dominant in visual search research and much work was carried out to understand which features are pre-attentive (e.g., [12],[33],[19]), which features are preferred by the HVS (e.g., [3]), how top-down information may be added to a master map and how this master map is used for directing attention (e.g., [13],[39], [35]).

Duncan and Humphreys [7] rejected the dichotomy of parallel vs. serial search, central to feature integration theory, and proposed an alternative hypothesis based on similarity. According to them, two types of stimulus similarity are involved in a visual search task. One,

denoted the *inter-alternative similarity*, is the similarity between objects in the scene and prior knowledge about the possible targets. The other similarity is between the objects in the scene. They suggested that the search procedures start with a hierarchial segmentation, defining *structural units*, which are the candidates competing for the attention resource. Units that are similar to the potential target get higher weights, which are used as priorities. The search is facilitated if several structural units are similar, because these are linked and when the priority is changed in one unit, the change propagates to the linked units as well. In this way suppression may be spread among the units, without needing to treat each one individually. Thus, if all non-targets are homogeneous, they may be rejected together, resulting in a fast (pop-out-like) detection process; if they are more heterogeneous, the search is slower.

It is widely believed that search task difficulty depends on all three characteristics: saliency (bottom-up information [32]), similarity to target models (top-down information [39, 7]), and similarities between the scene objects [7]. It is not clear whether attention is directed to locations or to full objects and there is evidence for both models [27].

Several models of attention mechanisms and visual search have been implemented, either as a part of studying the perceptual mechanism or for computer vision applications. They are based either on the theories described above (e.g. [13], [10], [37], [39], [35], [11]) or on different data sources. The indirect search approach [38] suggested looking first for large, easy-to-find *intermediate* objects, and then limiting the search to image regions inferred from available spatial relations knowledge; see [25] for another work based on spatial relations.

While several search mechanisms were proposed, relatively little was done on the analysis of search tasks. Tsotsos [34] analyzed the upper bounds of efficiency of visual search. A search may be bounded or unbounded. It is bounded when the model of the target is known. Then, each candidate is a set of connected pixels that can be compared to the model. In an unbounded search the model is unknown. Each subset of pixels many possibly form a target. He proved that the upper-bound for unbounded search is exponential in the image/s size, and that bounded search's complexity is linear in the image/s size. He also discussed the conditions in which a sequence of images facilitates the search.

Finally, we would like to observe that visual search, as considered in this paper, is related

to Content Based Image Retrieval (see, e.g. [5], [40], [4]). Both tasks look for familiar objects in a large collection of candidates and try to avoid the sequential testing of all candidates, by using similarities and interactions with a recognizer (computerized or human).

3 Framework

3.1 The context – visual search involving candidate selection and classification

We follow a common approach, dividing the search task into two sub-tasks. One is to *select* sub-images, which should be considered as possible *candidates*. The other, the *object recognition* task, is to decide whether or not a candidate is a sought for object. The input to the *candidate selection* task is the whole-scene image, and its output is a set of *candidate* sub-images. The candidate selection task can be performed by a segmentation process or even by a simple division of the image into small rectangles. It can be based on top-down or bottom-up processes, meaning that it uses prior knowledge about the target's properties or just image-based knowledge. The candidates may be of different size, bounded or unbounded [34], and can also overlap.

The input to the *object recognition* task is a sub-image (a candidate). The recognizer decides whether the given candidate is the required object. The decision can be based on statistical modeling, part decomposition, functional description or any other method. This stage is usually considered a high level vision task and is commonly computationally expensive, as the possible appearances of the object may vary due to changes in shape, color, pose, illumination and imaging conditions. The object recognizer may need to recognize a category of objects (and not a specific model), which usually makes it an even more complex task.

The object recognition process gets the candidates, one by one, after some ordering. This ordering is the attentional mechanism on which we focus here. Many orderings are possible but some are better than others. One goal of this work is to find a method for specifying good ordering so that the number of calls to the recognizer is minimized.

3.2 Sources of information for directing the search

The knowledge for directing the search may be different in different contexts:

1. Bottom-up saliency. The human attention mechanism is attracted to candidates that differ from the other candidates in some property, such as dominant orientation, size, and color

[32][13][11]. Nevertheless, this criterion can sometimes be misleading and is not applicable when, say, the target's uniqueness is defined by a combination of features, or when there are few similar looking targets in the scene.

2. Similarity to the sought for object (top-down approach) [39]. Both the candidate selection stage and the attention stage may obviously benefit from available prior knowledge on the targets. Indeed, the rejection based approach and efficient implementation yield an extremely fast search mechanism (see, e.g. [36]). If we look for a collection of objects or, say, for an object with highly various appearance, then either accuracy or speed are expected to decrease.

3. Mutual similarity of the candidates [7]. Usually a higher inner-scene visual similarity implies a higher likelihood for similar (or equal) identity. Under this assumption, after the identity of one (or few) candidates is already known, it can effect the likelihood of the remaining candidates having the same/different identity.

This last source of information, mutual inner-scene similarities, has not yet been considered in the field of computer vision. This work focuses on this property and studies its effectiveness.

3.3 Models of inner-scene visual similarity

We formalize the basic belief of mutual similarity in two alternative ways:

Deterministic: We assume that a measure of dissimilarity between two candidates $d(c_1, c_2)$ is higher than some threshold d_0 if the candidates do not share the same identity.

Stochastic: We assume that dissimilarity vs. identity correlation is a monotone descending function.

These assumptions are supported by validation experiments examining the behavior of dissimilarity vs. identity correlations; see Section 6.2.

3.4 Dynamic vs. Static Search

Usually, systems based on bottom-up or top-down approaches calculate a saliency map before the search starts, pre-specifying the preference and the implied scan order. In this case, the attentional process is called *static*. Still, for the pre-specified order the search mechanism may be implemented using a dynamically changed saliency as part of the search in order to, for instance, inhibit the return to the already attended locations (e.g., [13][11]). The attention

mechanism proposed here is *dynamic* in the sense that it allows the priorities to be changed based on the results of the object recognition process. As it uses more knowledge, it is expected to need fewer number of calls to the recognition process to find the target(s).

3.5 Algorithms

For each model of inner-scene similarity, deterministic and stochastic, we suggest a dynamic attention ordering algorithm, each of which is driven by the above assumptions, respectively.

All the algorithms we discuss are derived from one generic algorithm. They begin from a *static* priority map, indicating the initial likelihood of each candidate being a target. Iteratively, the candidate with the highest priority receives the attention. The relevant sub-image is investigated by a high-level object recognition module that we refer to as the *recognition oracle*. Based on the *oracle*'s response and the previous priority map, a new priority map is calculated, updating the priorities of the remaining candidates.

The knowledge that enables this generic algorithm to make such decisions is the availability of a similarity measure between each two candidates. For the implementation, the distance between extracted feature vectors provides the dissimilarity measure. The derived algorithms differ in the way the priorities are estimated in each iteration.

3.6 Measures of performance

A natural cost of a search process is either the time or the number of calls to the recognition oracle required before finding either the first target, or say, half of the targets, or even all of them. Predicting this cost can help stop the search process before all candidates are examined.

In the next section we provide predictions on the number of calls required before finding the first target. Evaluating the number of queries is justified when the costs associated with calling the oracle are dominant relative to the costs associated with the ordering process. This assumption is justified for computationally expensive recognition oracles. However, for our algorithm, we found that the preparation phase remained relatively inexpensive even compared to the demands of the extremely efficient AdaBoost based recognition oracle; see Section 6.4.

Choosing this cost also implicitly assumes that all calls to the recognizer 'waste' a constant processing time. This is a uniform, oracle independent, approach and good approximation for many recognition mechanism, but is not true for rejection based approaches, for example.

Other costs associated with finding more targets or with nonuniform computational costs, are left for future work.

4 Deterministic Analysis of Visual Search Performance

4.1 Introduction

A search task is easier if the mutual similarities are more informative. The simplest search situation is when all targets are very similar and all non-target candidates are mutually-similar but dramatically different from the target (one book among some pens and pencils). The hardest task is when all mutual similarities of pairs of candidates are alike and thus uninformative. Such situations happen when all candidates (both targets and non-targets) are very similar (some books all the same size and color) or when all candidates are different from each other (a bunch of office equipment). A task where the candidates are divided into a few groups, where members of the group are similar to each other and dissimilar to all candidates of other groups, and where one group contains all (and only) the targets, is of intermediate difficulty.

We consider the situation where some information on the search task is provided before it is executed (from, say, experience with search tasks in a similar context). This information is used to develop a meaningful quantifier of the search task difficulty. As we will see, the more we are pre-informed, the more accurately we can characterize the task's difficulty. The proposed maximal knowledge is not always available, however. Therefore, we consider other characteristics, which rely on weaker knowledge.

We define the cost of a search algorithm to be the number of recognition attempts required until a target is located. For each suggested characterization, we show that any search algorithm's cost, in the worst case, is not below a certain lower-bound (see Section 4.4), and suggest a specific algorithm with cost that is never above a certain upper-bound (see Section 4.5). We show that these lower and upper bounds coincide, implying that in the worst case sense, the bounds are tight and the algorithm is optimal.

4.2 Notations

We consider an abstract description of a search task as a pair (X, l), where $X = \{x_1, x_2, \ldots, x_n\}$ is a set of partial descriptions associated with the set of candidates, and $l = (l_1, l_2, \ldots, l_n)$ are

the binary identity labels of the candidates. $l_i = 1$ if candidate i is a target, and $l_i = 0$ if it is a non-target. The partial description can be, for example, feature vectors in Euclidian space. A search algorithm is provided with the set X, but not with the labels l. We define the cost of a search algorithm A, $\text{cost}(A, X, l)$, as the number of queries to the recognizer oracle until a target is located.

We refer to the set of partial descriptions $x_1, x_2, \ldots, x_n$ as points in a metric space (S, d). $d : S \times S \rightarrow \mathbb{R}^+$ being the metric distance function satisfies the conditions of reflexivity ($\forall s \in S, d(s, s) = 0$), positivity ($\forall s_i \neq s_j \in S, d(s_i, s_j) > 0$), symmetry ($\forall s_i, s_j \in S, d(s_i, s_j) = d(s_j, s_i)$) and triangle inequality ($\forall s_i, s_j, s_k \in S, d(s_i, s_j) + d(s_j, s_k) \geq d(s_i, s_k)$).

4.3 A measure for search difficulty combining target isolation and candidate scattering

The proposed measures for task difficulty combine two main factors:

1. The distance between target and non-target candidates.
2. The distribution of the candidates in the feature space.

Intuitively, the more the targets are different from non-targets, the easier the search is. However, if the non-targets are also different from each other, the search becomes hard again [7].

A useful quantification for expressing a distribution of points in a metric space (and thus expressing how scattered the candidates are) uses the notation of metric cover.

Definition 1 *Let $X \subseteq S$ be a set of points in a metric space (S, d). Let 2^S be a set of all possible subsets of S. $\mathcal{C} \subset 2^S$ is 'a cover' of X if $\forall x \in X \exists C \in \mathcal{C}$ s.t. $x \cup C \neq \emptyset$.*

Definition 2 *$\mathcal{C} \subset 2^S$ is a 'd_0-cover' of a set X if $\mathcal{C}$ is a cover of X and if $\forall C \in \mathcal{C}$ diameter$(C) \leq d_0$, where diameter(C) is $\max_{c_1, c_2 \in C} d(c1, c2)$.*

Definition 3 *A 'minimum-d_0-cover' is a d_0-cover with a minimal number of elements. We denote some particular minimum-d_0-cover by $\mathcal{C}_{d_0}(X)$ and its size by $c_{d_0}(X)$.*

When X is a set of feature vectors in an m-dimensional Euclidian space, for example, $c_{d_0}(X)$ will be the minimum number of m-dimensional spheres with diameter d_0 required to cover all candidates in X. The above definitions follow Kolomgorov's ϵ-covering [14].

Definition 4 *Given a search task (X, l), let the 'max-min-target-distance', denoted d_T, be the greatest distance of a target to its nearest non-target neighbor.*

When a search task contains one target, d_T is the distance of this target to the closest non-target. When there are several targets in the scene, $T_1, \ldots, T_m$, $d_T = \max_{i=1}^{m}(d_{T_i})$.

The following result quantifies the difficulty of the search task using the max-min-target-distance and the cover concept, and is the main result of this section.

Theorem 1 *Given a search task (X, l), so that d_T is its* max-min-target-distance*, the difficulty of finding a target depends on the minimum-d_T-cover size, $c_{d_T}(X)$, in the sense that:*

1. *Any search algorithm may need at least $c_{d_T}(X)$ queries before finding a target, for the worst case.*
2. *There is an algorithm that always finds a target after at most $c_{d_T}(X)$ queries.*

The proof to the two parts of Theorem 1 is given in Sections 4.4 and 4.5.

The minimum-d_T-cover size quantifies the intuition described in Section 4.1. In the case when one target differs dramatically from all non-targets, which are similar, $c_{d_0}(X)$ is 2. (All non-targets can be covered by one covering element, and the target alone is covered by a second element.) In a case where all n candidates are scattered and all mutual distances are $\geq d_T$, $c_{d_T}(X)$ is n. When there are k groups of candidates so that all inner-group distances $< d_T$ and all outer-group distances $\geq d_T$, $c_{d_T}(X)$ is k.

The above examples are trivial extreme cases in which the task difficulty is easily revealed. $c_{d_0}(X)$ also provides a measure for intermediate situations that are harder to analyze. For instance, the candidates may be scattered, but a target can be significantly far away from any of them, providing a relatively small d_T-cover. On the other hand, the candidates may be divided into a small number of groups, but since the targets and non-targets are mixed together in one of these groups (or more), $c_{d_T}(X)$ will be relatively big, characterizing the search difficulty correctly.

4.4 Lower-bounds on search algorithms

In this section, we prove Theorem 1.1 but first provide some other less tight lower bounds on all search algorithms. Consider a set of search tasks associated with some limitation on the partial descriptors $X \in \mathcal{X}$ and some limitation on the possible assignments $l \in \mathcal{L}$. Then, LB is a lower bound on the performance of algorithm A, if there is an input $X \in \mathcal{X}$ and an assignment $l \in \mathcal{L}$ so that $\text{cost}(A, X, l) \geq \text{LB}$.

$\mathcal{X}$ and $\mathcal{L}$ describe the information available about the search task. In this section we gradually reduce the uncertainty, and consequently, tighten the bound.

4.4.1 No knowledge

It is easy to show that if there is no limitation on the labeling, then only a trivial lower bound may be imposed. Note that knowledge about $\mathcal{X}$ does not matter here: for any algorithm and any set of candidates, there is a labeling for which the target is the last candidate examined.

Claim 1 $\forall A, \forall X, \exists l, cost(A, X, l) \geq n$.

Proof: Until a search algorithm A finds the first target, it receives only *no* answers from the oracle O. Therefore, given an input candidate set X, and a specific algorithm A, the sequence of attended candidates may be simulated under the assumption that the oracle returns only *no* answers. Let $\pi = \pi_1, \pi_2, \ldots, \pi_n$ be this resulting ordering of the candidates. Choose the assignment l to be $l_i = 1$ for $i = \pi_n$ and $l_i = 0$, otherwise. ∎

4.4.2 Minimal knowledge: $d_T > d_0$ and the number of isolated points

Suppose we know that at least one target is dissimilar from the non-targets. Quantitatively, suppose that d_T is bounded from below: $d_T > d_0$. Let $\mathcal{L}_{d_0}$ denote the possible labeling satisfying this condition. Furthermore, let d_i denote the feature-space-distance of the i-th candidate to its nearest neighbor. We define an isolated point as a feature-vector x_i for which $d_i > d_0$. Finally, let $\mathcal{X}_{d_0,N_{ip}}$ denote the set of candidate sets for which the number of isolated points is N_{ip}. The claim below suggests that N_{ip} is a lower-bound in this case.

Claim 2 $\forall A \exists X \in \mathcal{X}_{d_0,N_{ip}} \exists l \in \mathcal{L}_{d_0}\ cost(A, X, l) \geq N_{ip}$.

Proof: Under the limitation of $\mathcal{L}_{d_0}$, all candidates for which $d_i > d_0$ can be possible targets. Choose l to assign the label 1 only to the last candidate in π that satisfies $d_i > d_0$. ∎

Note that the target for which the distance to a distractor is maximal (d_T) can be associated with $d_i < d_0$, if its closest neighbor is a target as well. Therefore, the last claim suggest a bound which is not tight.

4.4.3 Bounded metric space

If the feature vectors $x_1, x_2, \ldots x_n$ belong to a bounded metric space $[0, 1]^m$, then the amount of feature vectors x_i that can satisfy $d_i > d_0$ may be limited, leading to a smaller bound.

Claim 3 *Let* $\mathcal{X}_m = \{X \mid x \in X \Rightarrow x \in [0, 1]^m\}$, *and specify* $\mathcal{L}_{d_0}$ *according to the* L_∞ *metric. Then,* $\forall A\ \exists X \in \mathcal{X}_m, l \in \mathcal{L}_{d_0}\ cost(A, X, l) \geq LB_m = \min(\lceil \frac{1}{d_0} \rceil^m, n)$

Proof: Consider a rectangular grid in the $[0,1]^m$ space such that the distance between grid points is more than d_0. There are $\lceil \frac{1}{d_0} \rceil^m$ such points. Choose the n candidates such that their description X are points divided equally between those $\lceil \frac{1}{d_0} \rceil^m$ locations'. Given an algorithm A defining the ordering π, choose the assignment l that assigns 1 to the group of candidates located in the grid point whose first appearance in π is last. ∎

Similar results for other metrics are available.

4.4.4 Knowledge: $d_T > d_0$ and $c_{d_0}(X)$

Now, in addition to being informed that $d_T > d_0$, we also know that the possible input X is limited to having some known minimal-d_0-cover of size c ($X \in \mathcal{X}_{d_0,c}$). In this case, we show that c is a lower bound on all search algorithms' cost. The following claim is equivalent to Theorem 1.1.

Claim 4 $\forall A \; \exists X \in \mathcal{X}_{d_0,c}, l \in \mathcal{L}_{d_0} \; cost(A,X,l) \geq LB_{d_0,c} = c$

Proof: Choose c points in the metric space, so that the entire inter-point distance is more than d_0. Choose the n candidates to de divided equally among these locations. Choose l that assigns 1 to the group of candidates located in the point whose first appearance in π (the ordering dictated by Algorithm A) is last. ∎

The proof clearly resembles that of Claim 3. In fact, the grid built in Claim 3 is a set of points associated with a cover of size $\lceil \frac{1}{d_0} \rceil^m$. Note that no specific metric is considered in the more general theorem, and of course, the metric cover is a much tighter bound, for most input.

4.5 A simple search algorithm and its upper-bounds

In this section we present a simple algorithm and show that its cost is bounded from above by the bounds presented in Section 4.4. UB is an upper bound on the performance of an algorithm A if for all input $X \in \mathcal{X}$ and assignment $l \in \mathcal{L}$, $\text{cost}(A,X,l) \leq UB$.

4.5.1 Simple search algorithms

The following two simple algorithms aim to minimize the cost of finding the first target.

FNN- Farthest Nearest Neighbor: Given the set of candidates' partial description $X = \{x_1, \ldots, x_n\}$, compute the distance d_i of each candidate i to its nearest neighbor. Order the candidates by descending d_i and query the oracle according to this order until finding a target.

FLNN- Farthest Labeled Nearest Neighbor: Given the set of candidates' partial description $X = \{x_1, \ldots, x_n\}$, randomly choose the first candidate, query the oracle and label this candidate. Repeat iteratively until a target is detected: for each unlabeled candidate i, compute the distance dL_i to the nearest labeled neighbor. Choose the candidate i for which dL_i is maximum. Query the oracle to get its label.

FNN is optimal for target assignments $l \in \mathcal{L}_{d_0}$ when there is only one target in the scene. When two targets are present in the scene, they may be similar to each other and hence may be detected last by FNN, even if each of them is far from the nearest non-target. Therefore, we prefer FLNN, which can handle several targets. We now show that, in the worst case, it does not need more queries than those specified in Claims 3 and 4, implying that the latter bounds are tight.

4.5.2 Bounded metric space

As in Section 4.4.3, we assume a lower bound d_0 on d_T and consider candidates that belong to the bounded metric space $[0,1]^m$. We show that FLNN always finds a target after at most LB_m queries.

Claim 5 $\forall X \in \mathcal{X}_m, l \in \mathcal{L}_{d_0}$ $cost(FLNN, X, l) \leq UB_m = min(\lceil \frac{1}{d_0} \rceil^m, n) = LB_m$.

Proof: Let the i-th candidate be a target for which the distance to its nearest distractor is $d_T > d_0$. Consider the hypercube H with sides of length $2d_0$ so that x_i is its center. H includes only targets. The rest of the metric space $[0,1]^m$ may be covered with at most $\lceil \frac{1}{d_0} \rceil^m - 1$ hypercubes of side d_0. By construction, FLNN does not query two candidates in one hypercube (which are at most a distance of d_0 from one another) before it queries at least one candidate in H. Therefore, a target (not necessarily i) will be found after at most $min(n, \lceil \frac{1}{d_0} \rceil^m)$ queries. ∎

Claim 6 *Consider the partition of the* $[0,1]^m$ *into* $\lceil \frac{1}{d_0} \rceil^m$ *hypercubes, and let* $populated(X, d_0)$ *be the number of hypercubes containing at least one candidate. Then,*
$\forall X \in \mathcal{X}_m, l \in \mathcal{L}_{d_0}$ $cost(FLNN, X, l) \leq UB_{populated} = populated(X, d_0)$.

The proof is similar to that of Claim 5. While this result is simple, it provides a crude version of our main result, which uses the more complex cover concept, it gives a much tighter bound than UB_m and is much easier to compute than the cover size.

4.5.3 $c_{d_0}(X)$ as an upper bound for search cost

The tightest upper bound, proposed in Theorem 1, is now proved:

Claim 7 $\forall X \in \mathcal{X}_{d_0,c}, l \in \mathcal{L}_{d_0}$ $cost(FLNN, X, l) \leq UB_{d_0,c} = c = LB_{d_0,c}$.

Proof: Take an arbitrary minimum-d_0-cover of X, $\mathcal{C}_{d_0}(X)$. Let the i's candidate be a target so that $d(x_i, x_j) > d_0$ for every distractor j. Let C be a covering element ($C \in \mathcal{C}_{d_0}(X)$) so that $x_i \in C$. Note that all candidates in C are targets. Excluding C, there are $(c-1)$ other covering elements in $\mathcal{C}_{d_0}(X)$ with diameter $\leq d_0$. Since C contains a candidate whose distance from all distractors $> d_0$, FLNN does not query two distractor-candidates in one covering element (whose distance $\leq d_0$), before it queries at least one candidate in C. Therefore, a target will always be located after at most c queries. (It is possible that a target, which is not in C, will be found before. The algorithm then stops even earlier.) ■

The minimum cover size, $c_{d_0}(X)$, can be up to 2^m times smaller than populated(X, d_0) [1], and always not above it. Therefore, Claim 7 is an improvement over Claim 6. Nevertheless, while populated(X, d_0) can be calculated easily, computing $c_{d_0}(X)$ can be hard for a large set of candidates. The problem of finding the minimum cover is NP-hard. Gonzalez [9] proposed a 2-approximation algorithm for the problem of clustering a data set minimizing the maximum inner-cluster distance, and proved that it is the best approximation possible if $P \neq NP$.

4.6 Illustration of the above results

We demonstrate the latest result using a toy example illustrated in Figure 1. (More realistic examples are given in Section 6.) The image contains seven different insects, which are roughly segmented. For simplicity of demonstration, the search uses only the objects' area as a partial description (feature); see the distribution of the objects in one-dimensional feature space in Figure 1c.

Taking the ladybug as the target, the *minimum-d_T-cover* is of size 3; see Figure 1d. This means that using the FLNN algorithm, the ladybug is detected after examining at most two other objects.

[1] Consider a corner of one of the covering hypercubes discussed in the proof to Claim 5. This corner is also a corner of $2^m - 1$ other neighboring hypercubes. Imagine there is one candidate in each of these 2^m cubes, very close to that corner. In such a case populated(X, d_0) is 2^m while $c_{d_0}(X)$ is 1.

Note that a. the cover size and the implied search difficulty depend on the partial description (features), and b. neither the cover size nor the search time changes if more distractors (butterflies, ants) or targets (ladybugs), similar to the existing ones, are added.

5 Dynamic Stochastic Search Algorithm

In this section we suggest a somewhat different and stochastic framework for dynamic visual search, as well as a specific algorithm (VSLE) based on linear estimation.

5.1 The need to extend FLNN

Our search strategy is based on the assumption that a target is likely to be different from all distractors. In Section 4 we took a deterministic approach and proposed the FLNN algorithm that indeed prefers the candidate that is maximally different from the nearest recognized distractor. FLNN stops after finding the first target. Extending it so it will continue and find more targets after the first is located, introduces some problems. According to our assumptions, the consequent targets should be close (in feature-space distance) to the already found targets, and far from the attended distractors. It is not self-evident how to quantify these distances and how to combine them into one value indicating the likelihood of a candidate to be a target. A second weakness of the FLNN algorithm is that it lacks robustness; a single "error" in the form of an attended distractor close to an undetected target will reduce the priority of this target and slow down the search.

5.2 Object labels as dependent random variables

Taking a stochastic approach, we model the object identities as binary random variables with possible values 0 (for non-target) or 1 (for target). These values are initially unknown, and are revealed one by one by the recognition oracle. Estimates of these variables may be available and will help in directing the search.

Intuitively, we know that objects associated with a similar identity (or category) tend to be visually more similar than objects that have different identities. Quantifying this intuition in a probabilistic way, we suggest that the random variables associated with two candidates are more dependent if the visually similarity between these candidates is higher. Specifically, we characterize the dependency behavior as follows:

Basic probabilistic assumption: Similarity dependent stochastic dependency The covariance between two labels is a monotonic ascending function of their visual similarity and a descending function of the feature-space-distance between them,

$$cov(l_i, l_j) = \gamma(d(x_i, x_j)),$$

where l_i and l_j are the labels of candidates i and j, $d(x_i, x_j)$ is the feature space distance between the two candidates, and γ is a monotone descending function.

In Section 6.2 we experimentally challenge this assumption and find that, indeed, such a monotonic dependency is found in most cases. From the experiments we found that an exponentially descending γ is a good approximation for the actual dependency in many cases.

This assumption is one way to quantify the intuition presented above. The advantage in knowing the second order statistics is that it allows us to infer unknown labels from known ones, using common least squares estimation techniques.

5.3 Dynamic search framework

We propose the following greedy approach to dynamic search. At each iteration, estimate the probability of each unlabeled candidate to be a target using all the knowledge available. Choose the candidate for which the estimated probability is the highest and apply the object recognition oracle to the corresponding sub-image.

Formally, after the $m-$th iteration, m candidates with partial descriptions $x_1, x_2, \ldots, x_m$, have already been attended and m labels, $l_1, l_2, \ldots, l_m$ are known. We use them to estimate the conditional probability of the label l_k of each unlabeled candidate k to be 1:

$$p_k = p(l_k = 1 \mid l_1, \ldots, l_m).$$

5.4 Minimum mean square error linear estimation

Now, note that the random variable l_k is binary and, therefore, its expected value is equal to its probability of taking the value 1. Estimating the expected value, conditioned on the known data, is generally a complex problem and requires knowledge about the labels' joint distribution. We chose to use a linear estimator minimizing the mean square error criterion,

which needs only second order statistics. Given the measured random variables $l_1, l_2, \ldots, l_m$, we seek a linear estimate $\hat{l_k}$ of the unknown random variable l_k,

$$\hat{l_k} = a_0 + \sum_{i=1}^{m} a_i l_i,$$

which minimizes the minimum mean square error $e = E((l_k - \hat{l_k})^2)$. This is a standard task with a known solution [22]:

$$\hat{l_k} = E[l_k] + \vec{a}^t(\vec{l} - E[\vec{l}]),$$

where $\vec{l} = (l_1, l_2, \ldots, l_m)$ and $\vec{a} = R^{-1} \cdot \vec{r}$, where r_i, $i = 1, \ldots, m$, R_{ij}, $i, j = 1, \ldots, m$ are given by $R_{ij} = \mathrm{cov}(l_i, l_j)$ and $r_i = \mathrm{cov}(l_k, l_i)$.

The estimated label $\hat{l_k}$ is the conditional mean of a label l_k of an unclassified candidate k, and, hence, may be interpreted as the probability of l_k to be 1:

$$p_k = p(l_k = T \mid l_1, \ldots, l_m) \sim \hat{l_k}.$$

5.5 The VSLE algorithm

- *Given a scene image, choose n sub-images as candidates.*
- *Extract the set of feature vectors* $X = \{x_1, x_2, \ldots x_n\}$.
- *Calculate pairwise feature space distances and the implied covariance for each pair.*
- *Select the first candidate/s randomly (or based on some prior knowledge).*
- *In iteration* $m + 1$*:*
  - *For each candidate* k *out of the* $n - m$ *remaining candidates, estimate* $\hat{l_k} \in [0, 1]$ *based on on the known labels* $l_1, \ldots, l_m$.
  - *Query the oracle about the candidate* k *for which* $\hat{l_k}$ *is maximum.*
  - *If enough targets are found - abort.*

Notes:

1. The extraction of the candidates, and the choice of the feature vector and the feature space distance may depend on the application (see Section 6).

2. The first candidate may be chosen based on different considerations. Choosing, for example, the first candidate from a large subset of similar candidates has the potential to reduce the number of potential candidates significantly, and therefore, may be the best choice, if the prior probability of all candidates to be a target is uniform. This, however, seems to be an unjustified assumption as targets usually do not appear in large groups. (If they did, then saliency would never work.)
3. The estimation of the label associated with any unknown candidate is accelerated only if its nearest (classified) neighbors are used. As the covariance decreases with distance, ignoring far neighbors may be justified. (For example, in the experiments described in Sections 6.3 and 6.4 we use only the $k = 3$ nearest classified neighbors for each estimation.)
4. If the non-targets are similar, and the target differs significantly from them, we get a pop-out like behavior, and the target is found after one or two iterations.
5. If the non-targets consist of k clusters in the feature space, then at most k candidates are tested before the target is attended. If the clustering is not strong, performance smoothly degrades and more attempts are required before the target is found.
6. If the targets themselves are one cluster, then after the first is detected, the rest follow immediately.

5.6 Combining prior information

Bottom-up and top-down information may be naturally integrated by specifying the prior probabilities (or the prior means) according to either the saliency or the similarity to known models. Moreover, if the top-down information is available as k model images (one or more), we can simply add them as virtual candidates that were already examined by the oracle and got a 'yes' answer. Continuing the search from this point is likely to be faster. As this is not the focus of this paper, we do not elaborate. See, however, some relevant experiments in Section 6.2.

6 Experiments

In order to test the proposed search mechanism, we conducted many experiments using images of different types, different methods for candidate selection, and different partial descriptions.

First we demonstrate the relation between the algorithms' performance and the tasks' difficulty (Sections 6.1 and 6.2). Then, we describe some more thorough testing of the VSLE algorithm (Section 6.3) and compare the performance of a combination of VSLE and Viola and Jones' (VJA) algorithms, to that of VJA alone (Section 6.4).

6.1 FLNN and Minimum-Cover-Size

The first set of experiments considers several search tasks and focuses on their characterization using the proposed metric cover. Because calculating the minimal cover size is computationally hard, we suggest several ways to bound it from above and below, and show that combining these methods yields a very good approximation. In this context we also test the FLNN algorithm and demonstrate its guaranteed performance. Finally, we provide the intuition explaining why indeed harder search tasks are characterized by larger covers.

The first three search tasks are built around the 100 images corresponding to the 100 objects in the COIL-100 database [21] in a single pose; see Figure 2a. We think of these images as candidates extracted from some larger image. The extracted features are first, second, and third Gaussian derivatives in five scales [24] resulting in feature vectors of length 45; see Figure 2c. A Euclidian metric is used as the feature space distance. The tasks differ in the choices of the target that were cups (10 targets), toy cars (10 targets) and toy animals (7 targets) in the three search tasks.

The minimal cover size for every task is bounded as follows: First the minimal target-distractor distance, d_T, is calculated. We developed a greedy heuristic algorithm that prefers sparse regions and provides a possibly non-tight but always valid d_T- cover. For all tasks, this algorithm provided smaller (tighter) covers than those obtained with a 2-approximation algorithm [9]. Both algorithms provide upper bounds on the size of the minimal cover; see Table 1 for cover sizes. Being a rigorous 2-approximation, half of the latter upper bound value (42/2=21 for the cups) is also a rigorous lower bound on the minimal cover size. Another lower bound may be found by running the FLNN algorithm itself, which, by Theorem 1, needs no more than $c_{d_T}(X)$ queries to the oracle. By running the algorithm 100 times, starting from a different candidate each run and taking the largest number of queries required, we get the tightest lower bound; see Table 1 where the average number of queries required by the FLNN

is given as well.

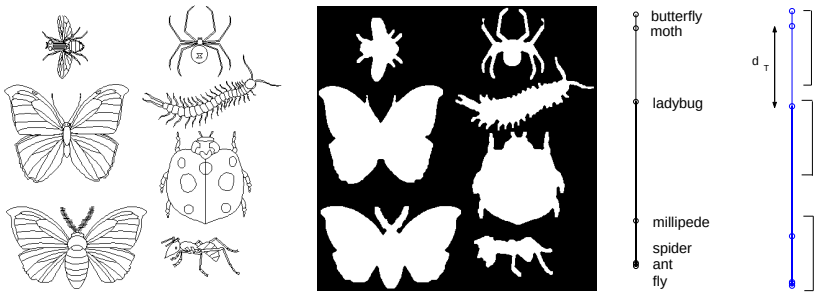
Note that the search for cars was the hardest. While the car targets are very similar to each other (which should ease the search), finding the first car is hard due to the presence of similar distractors (d_T is small). The cups are also similar to each other, but are dissimilar to the distractors, implying an easier search. On the other hand, the different *toy animals* are dissimilar, but as one of them is very dissimilar from all candidates, the task is easier as well. Note that the minimal cover size captures the variety of reasons characterizing search difficulty in a single scalar measure.

We also experimented with images from the Berkeley hand segmented database [17] and used the segments as candidates; see Figure 3. Small segments are ignored, leaving us with 24 candidates in the *elephants* image and 30 candidates in the *parasols* image. The targets are the segments containing elephants and parasols, respectively. For these color images we use color histograms as feature vectors. In each segment (candidate), we extract the values of $\frac{b}{r+g+b}$ and $\frac{r}{r+g+b}$ from each pixel, where r, g, and b are values from the RGB representation. Each of these two dimensions is divided into 8 bins, resulting in a feature vector of length 64. Again, we use a Euclidean metric for distance measure. (Using other histogram comparison methods, such as the ones suggested in [29], the results were similar.) See the results in Table 1 and Figure 4. Although the mean results are usually not better than the mean results of a random search, the worst results are much better.

6.2 VSLE and Covariance Characteristics

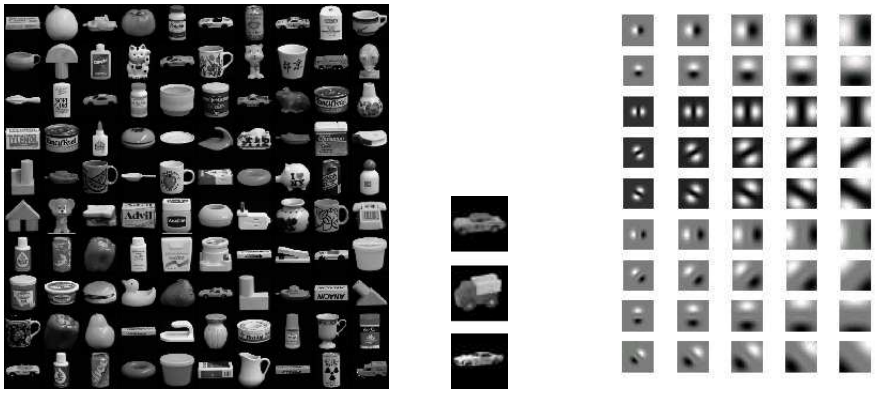
The VSLE algorithm described in Section 5 was implemented and applied to the same five visual search tasks described in Section 6.1. We model the dependencies of the covariance on the feature-space-distance using $cov(l_1, l_2) = \gamma(d(x_1, x_2)) = \mu(1-\mu)e^{-d(x_1,x_2)}$, where l_i is the label of candidate i, $d(x_i, x_j)$ is x_i and x_j feature-space distance, and $\mu = E(l_i)$. The results are described in Figures 5a to 5e. The solid lines in the graphs describe one typical run. Other runs, starting each time from a different candidate, are described by the size of the gray spots as a distribution in the (time, number of target found) space.

Unlike FLNN, which deals only with finding the first target, VSLE continues and aims also



(a) (b) (c) (d)

Figure 1: An illustration of the cover upper bound. (a) Input image. (b) Segmentation results. (c) The representation of candidates in 1-dimensional feature space, with the objects' area serving as the feature. (d) A d_T-cover associated with the ladybug being a target.



(a) (b) (c)

Figure 2: (a) The objects from the COIL-100 database used as candidates. (b) The three model images used for demonstrating the use in top-down information. (c) The Gaussian derivatives filters used for extracting a partial description of the candidates. First two rows: first derivatives in 0° and 90° in different scales; three successive rows: second derivatives in 0°, 60° and 120°; last four rows: third derivatives in 0°, 45°, 90°, and 135°.

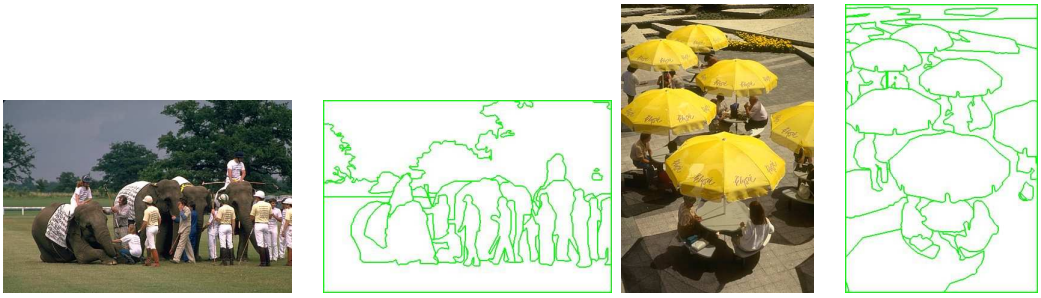


Figure 3: The *elephants* and *parasols* images taken from the Berkeley hand segmented database and the segmentations used in our experiments (color images).

to find the other targets. Moreover, in almost all the experiments we performed, VSLE was faster in finding the first target (both in the worst and the mean results). See the rightmost column in Table 1.

In all above experiments the candidates of each task can be divided into a few weakly homogeneous groups (or clusters), which seems to be the situation in many natural scenes as well. To compare between the results of the experiments qualitatively, we focus on the two following difficulty factors: d(T,NT)- relative feature-space-distance between targets and non-targets, and d(T,T)- relative feature-space-distance between targets and targets. This qualitative description is summarized in Table 2.

VSLE relies on the covariance between candidates' labels. We use a covariance function that depends only on feature-space-distance, and argue that for many search tasks this function is monotonically descending in this distance. To verify this assumption we estimate the covariance of labels vs. feature-space-distance of search tasks. Given a search task, i.e., the set of candidates and their labels, we compute the distances between each pair of candidates. The distances are sorted and divided into h intervals. (Non-uniform intervals containing an equal number of target-target pairs resulting less erratic estimates.) The labels covariances $\gamma_i \;\; i = 1, \ldots, h$ are calculated separately for every interval I_i:

$$\begin{aligned} \gamma_i &= cov_i(l_1, l_2) = E(l_1 l_2) - E(l_1)E(l_2) \\ &\simeq \frac{n_{TT} \cdot 1 \cdot 1 + n_{TN} \cdot 1 \cdot 0 + n_{NN} \cdot 0 \cdot 0}{n_{TT} + n_{TN} + n_{NN}} - \hat{\mu}^2 = \frac{n_{TT}}{n_{TT} + n_{TN} + n_{NN}} - \hat{\mu}^2 \end{aligned}$$

where n_{TT}, n_{TN}, and n_{NN} are the number of target-target, target–non-target, and non-target–non-target pairs in interval I_i, respectively. $\hat{\mu} = E(x)$ is the number of total targets divided to the number of total candidates. The estimated covariances for the five search tasks described in Section 6.1 are plotted as a function of distance in Figure 6. Except for the task of finding toy animals (in which there is almost no resemblance among targets), all other results show a monotonically descending behavior.

Using the method suggested in Section 5.6, we incorporate top-down information and demonstrate it on the *toy cars* case: three toy cars that do not belong to the COIL-100 database are used as model targets; see figure 2b. The search time was significantly reduced as expected; see Figure 5f.

Search task	# of cand.	# of targets	FLNN worst	FLNN mean	Heuristic cover size	2-Approx. cover size	Real cover size	VSLE worst
cups	100	10	18	8.97	24	42	21-24	15
cars	100	10	73	33.02	79	88	73-79	39
toy animals	100	7	22	9.06	25	42	22-25	13
elephants	24	4	9	5.67	9	11	9	8
parasols	30	6	6	3.17	8	13	7-8	4

Table 1: Experiment results for FLNN and cover size. The real value of the minimal cover size is bounded from below by 'FLNN worst' and half of the '2-Approx. cover size', and bounded from above by the 'Heuristic cover size' and the '2-Approx. cover size'. The rightmost column shows that VSLE improves the results of FLNN for finding the first target.

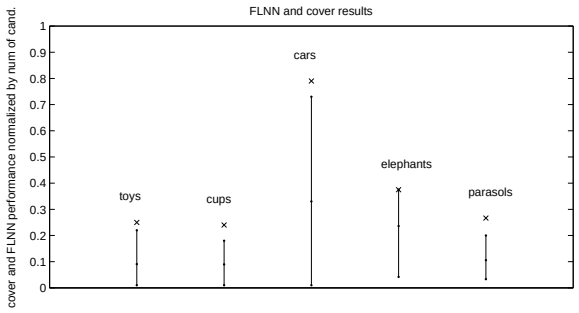
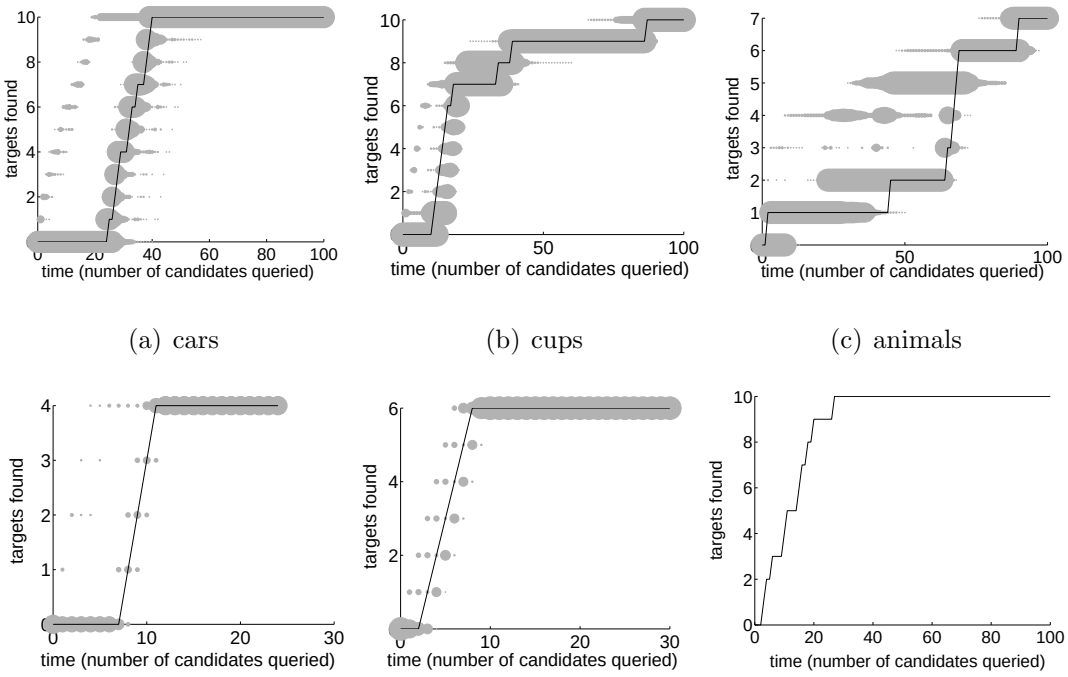


Figure 4: Graphic representation of FLNN and cover size experiment results. The results in Table 1 are normalized by the number of candidates to enable comparison between the difficulty of different search tasks. The vertical lines describe the results of all FLNN runs, the dots being the best, mean and worst results. The x describes the heuristic calculated cover size.

	d(T,T)	d(T,NT)	Finding first target	Finding successive targets
cars	small	small	hard	easy
cups	some small, some big	big	easy	easy to find some. hard to find all
animals	big	big	easy	hard
elephants	small	intermediate	easy	intermediate
parasols	small	big	easy	easy

Table 2: Qualitative search performance dependencies on the target-target similarity and target–non-target similarity.



(a) cars (b) cups (c) animals

(d) elephants (e) parasols (f) cars with top-down

Figure 5: VSLE results. The solid lines describe one typical run. Other runs, starting each time from a different candidate, are described by the size of the gray spots as a distribution in the (time, number of targets found) space. Figures (a) to (c) are results for the candidates from the COIL-100 database. In (a) the cars are the targets; the first car is quite hard to find. Once it is found, the rest are detected quickly. In (b) cups are the targets. It is easy to find the first cup since most of the cups are different from the non-targets. Most of the cups resemble each other and follow pretty fast, but there are two cups (one without a handle and one with a special pattern) that are different from the rest of the cups, and are found rather late. In (c) the toy animals are the targets. It is easy to find the first animal since one animal differs significantly from all non-targets. There is no special resemblance among the different toy animals, therefore, the rest of the search proceeds similarly to a random search procedure. Figures (d) to (e) are results for the Berkeley hand segmented color images. In (d) finding the first elephant is of medium difficulty since the color of the elephants is not too salient. Once the first is found, the rest of the elephants, having a similar color, follow. In (e) all the parasols are detected very fast, since their color is similar and differs from that of all other candidates. In (f) we demonstrate how VSLE can be improved by top-down information for the *toy cars* search task.

6.3 VSLE applied to MIT+CMU database

As a more intensive validation of the VSLE algorithm, we tested it using the MIT+CMU database [26][28], which includes 130 images with 511 labeled frontal faces.

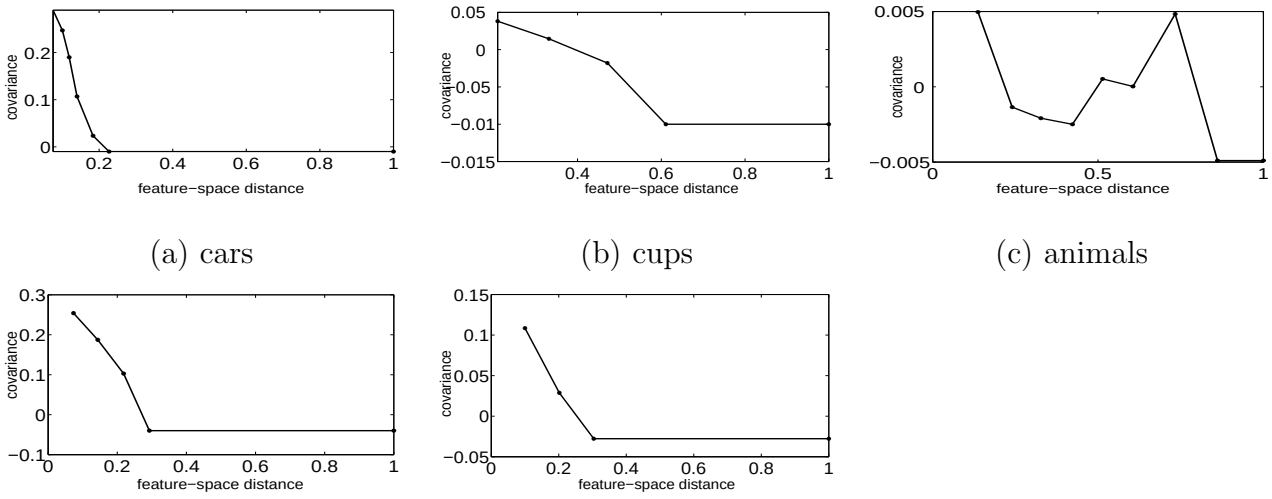
For these images, candidates were dilated bounding-boxes of image segments resulting from a simple image-pyramids based segmentation [2]. Each candidate is assigned with a range of scales, depending on its size, in which recognition attempts are performed. (i.e., when a recognition oracle is applied to a candidate sub-image, it searches only for faces that fit the size of the candidate) The similarities between candidates are estimated by distance between gray level histograms (see implementation details in Appendix A). Using a perfect oracle, we compared the detections times when serially passing through the candidates to the detection times when using VSLE to set the order of queries; see Figure 7a. As expected, the curve of the serial search is close to a linear curve, while VSLE detects targets faster. For instance, if for each image we stop VSLE after checking 30% of the candidates, 72% of the targets are found. If we stop VSLE after checking 55% of the candidates, 90% of the targets are found; see Table 3a and Figure 8. Note that this improvement is achieved using very simple features, and no model-based information. Also note, that using this simple candidate selection process, we are able to recognize 487 out of the 511 faces in the database. This can be improved, of course, if the segmentation process was model-based rather than a process that connects blobs of similar gray level.

In order to check whether VSLE helps in early rejection of non-targets and does not only help in detecting successive targets after other targets are found, we separately checked its behavior on images that include a single face; see Figure 7b. Out of the 130 images, there are 58 such images. On average, the face is detected after checking 35% of the candidates.

6.4 Combining VSLE and VJA

Aiming to test the proposed attention mechanism with a real oracle, we combined the VSLE algorithm with the popular face detector algorithm proposed by Viola and Jones [36]. The latter algorithm is based on a cascade of AdaBoost classifiers, implemented efficiently, resulting in an extremely fast model based search algorithm.

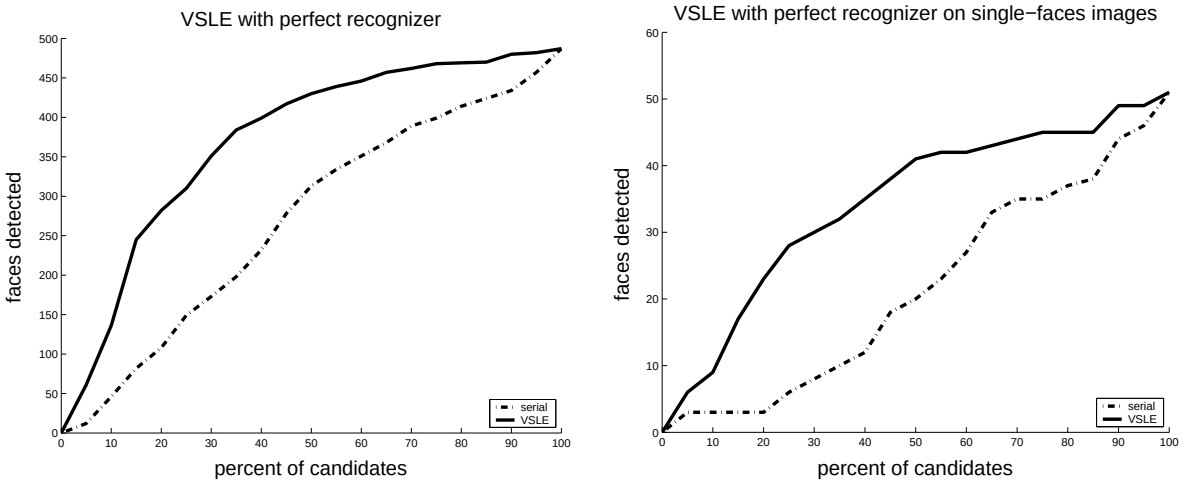
We used the same candidates selection and similarity measures described in Section 6.3,



(a) cars (b) cups (c) animals

(d) parasols (e) elephants

Figure 6: Estimation of labels covariance vs. feature-space-distance.



(a) (b)

Figure 7: VSLE vs. serial search applied to the MIT+CMU database. Using perfect oracle. (a) Applied to all the database. (b) Applied only to single face images.

a) VSLE applied on MIT+CMU database using a perfect face detection oracle

Percent of candidates queried	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100
Percent of targets found	13	28	50	58	64	72	79	82	86	88	90	92	94	95	96	96	97	99	99	100

b) VSLE applied on MIT+CMU database using VJA as the recognition oracle

Percent of candidates queried	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100
percent of total time (in seconds)	21	28	34	39	44	49	54	58	62	66	70	74	78	81	84	87	90	93	96	100
Percent of targets found	17	34	48	57	67	74	81	84	86	88	92	93	93	95	96	97	97	98	99	100

Table 3: VSLE applied to the MIT+CMU database. Demonstrates the percentage of targets found if VSLE is stopped at different times.

and sent the resulting candidates, one by one according to the order specified dynamically by VSLE to the VJA algorithm used as the recognition oracle. We used Intel's OpenCV library [15]; see implementation details in Appendix A. The number of detected targets (faces) and the number of false detections are plotted in Figure 9 (solid lines) as a function of time and compared to the results of applying the VJA over the full images.

Note that in the initial stage of VSLE, some time is spent on the segmentation process (about 12 sec) and in calculating the similarities (about 5 sec). Then it finds targets much faster than the serial VJA, until a certain point, when the remaining targets are dissimilar to those already found, slowing the search.

To compare the combined VSLE and VJA to the original VJA, the graphs in Figure 9 are plotted against time and not against the number of candidates (as was the case in previous experiments). Note that this makes the change in priorities, caused by the VSLE, less visible because the VJA (using its rejection cascade) spends more time in processing targets than on the processing of most non-targets; see Table 3. Still, the curve is much above a linear curve, and at a certain range above the curve of the original VJA as well.

The results suggest that the proposed combination is preferable (over the VJA) if the search time is limited to 75% or less of the VJA run time. Note that the advantage of the combined algorithm is somewhat better: with the already trained VJA we used, the false detections rate is lower with the combined method. For a fair comparison, we could train the independently used VJA so that the false detection rates are similar. This would increase the VJA misses as well, but the change would not be substantial, as apparent from the ROC curve in [36].

7 Discussion

In this paper we considered the usage of inner-scene similarity for visual search. Similarity is useful for search because, intuitively, similar objects are more likely to share the same identity. Therefore, similar non-targets may be 'rejected together' and finding one target helps to find the others.

The intuitive assumption is quantified either by a deterministic or a stochastic approach. Taking a quantitative approach allowed us not only to optimize the search but to also quanti-

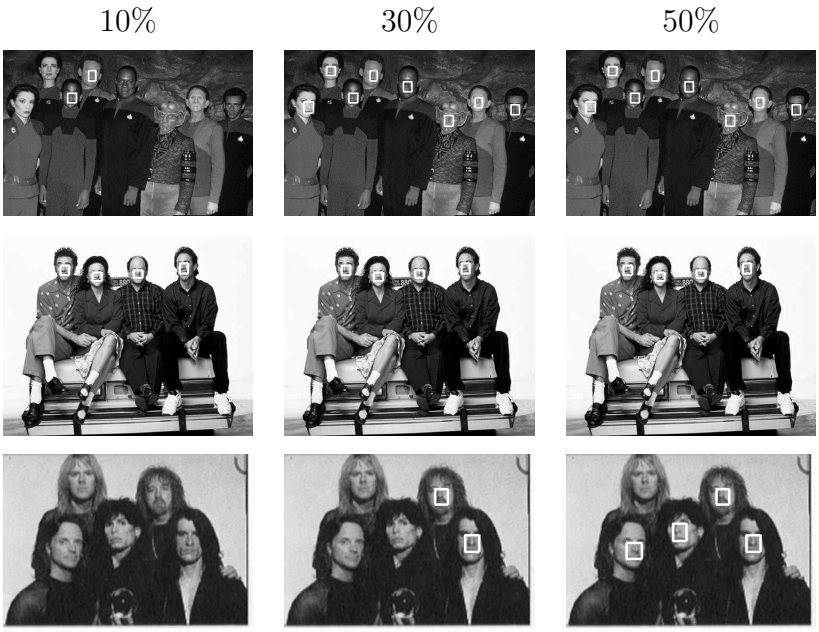


Figure 8: VSLE applied to the MIT+CMU database using a perfect oracle. Some examples of the times faces were detected. The percentages above the images indicate the percentage of candidates investigates before VSLE was stopped.

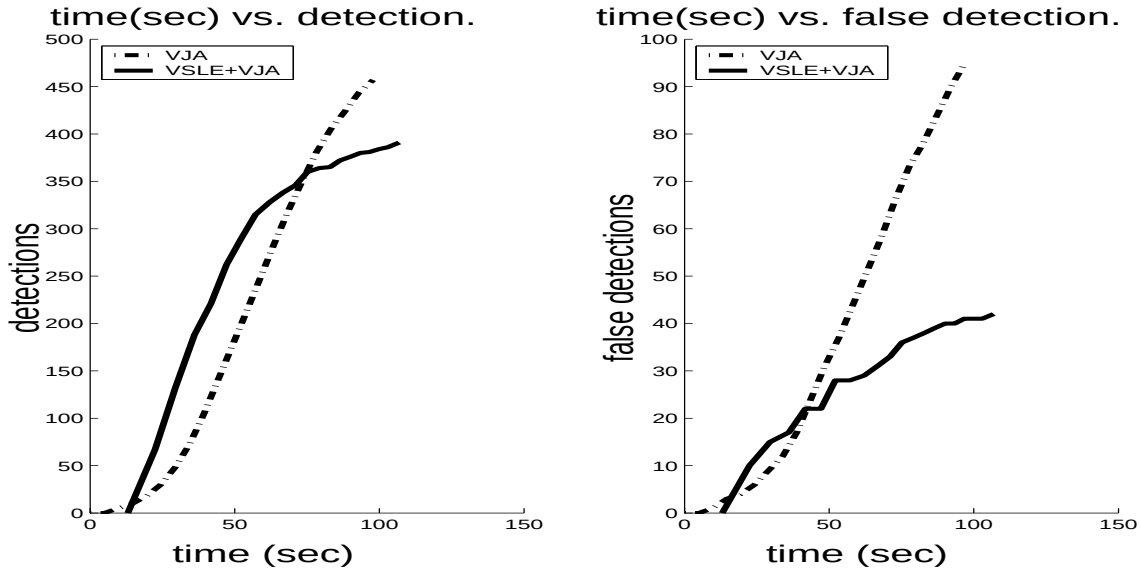


Figure 9: Detection times of the Viola and Jones algorithm (VJA), and of the combined implementation of VJA and VSLE, applied to the MIT+CMU database.

tatively predict its performance. Search performance naturally depends both on the task difficulty and the algorithm effectiveness. The d_T cover size measure of the search task difficulty, proposed here, is an inherent, algorithm independent measure and bounds the performance of every possible algorithm.

Interestingly, while, originally, we did not aim at modeling the HVS attention system, it turns out that the proposed search model is similar to Duncan and Humphreys's HVS attention model [7]. As such, our work can be considered as a quantification of their observations. Not surprisingly, our results also show that there is continuity between the two poles of 'pop-out' and 'sequential' searches.

While many search tasks rely only on top-down or bottom-up knowledge, inner scene similarities always helps, and may even become the dominant source of knowledge. Consider, for example the *parasols* search task example (Figures 3 and 5e). Note that the targets encompass a significant fraction of the image, and therefore, cannot be salient. The parasols are similar to each other and different from the non-targets in their color, but if this color is unknown, they cannot be searched using top-down information.

Notably, objects belonging to the same scene and the same category seems to be more similar than objects belonging to the same category but to different scenes. This happens because the imaging conditions are more uniform and because the variability of objects is smaller in one place (e.g, two randomly chosen trees are more likely to be of the same type if they are taken from the same area). Hence, inner scene similarities are expected to be higher than the similarities of a scene's objects to some prespecified model (or database of models), which is the type of information used for top down search mechanisms.

As expected, using inner-scene similarity allows the majority of targets to be found earlier than when using a sequential search. This was experimentally verified to be the case even in comparison to a very efficient detection algorithm. While the performance of the combined algorithm (VSLE+VJA) and that of the VJA by itself seems comparable, we expect that harder search tasks involving, say, objects from several classes and objects associated with a more visually varying appearance would make the advantage of combined algorithms higher. Such tasks require a more complex recognition oracle (e.g., [8]), which is not likely to be

implementable with the extreme efficiency associated with rejection based methods, and in particular, VJA. Then, investing more in similarity evaluation and in ordering would save more time.

Visual search involves a tradeoff between time and accuracy (detection and false alarm rates). At the moment it seems that the proposed approach relying (currently) on non-model-based candidate selection would not be able to compete on the basis of high accuracy, when time does not matter. It is, however, very effective when time is a contraint under which the best accuracy is sought.

We believe that an optimal search mechanism should use all the information available and stay within the time contraints. Therefore, we plan to generalize the principles proposed here so that bottom-up and top-down information are coherently and effectively integrated. While we tested the proposed algorithms using a variety of recognition and segmentation processes, we would like to also extend the analysis so that it takes into account the common imperfections of these processes.

References

[1] S. Baker and S.K. Nayar. Pattern rejection. *In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 544–549, June 1996.

[2] P.J. Burt, T.H. Hong, and A. Rosenfeld. Segmentation and estimation of image region properties through cooperative hierarchical computation. *SMC*, 11(12):802 – 809, December 1981.

[3] T.C. Callaghan. Interference and domination in texture segregation. *Visual Search. Proceedings of the First International Conference on Visual Search*, pages 81–87, 1988.

[4] C. Carson, S. Belongie, H. Greenspan, and J. Malik. Blobworld: Image segmentation using expectation-maximization and its applications to image quering. *IEEE PAMI*, 24(8):1026–1038, 2002.

[5] I.J. Cox, M.L. Miller, T.P. Minka, T.V. Papathomas, and Peter N. Yianilos. The bayesian image retrieval system, pichunter. *IEEE T. on Image Processing*, 9(1):20–37, January 2000.

[6] S. J. Dickinson, H. I. Christensen, J. K. Tsotsos, and G. Olofsson. Active object recognition integrating attention and viewpoint control. *Computer Vision and Image Understanding*, 67(3):239–260, September 1997.

[7] J. Duncan and G.W. Humphreys. Visual search and stimulus similarity. *Psychological Review*, 96:433–458, 1989.

[8] L. Fei-Fei, R. Fergus, and P. Perona. A bayesian approach to unsupervised one-shot learning of object categories. In *ICCV03*, pages 1134–1141, 2003.

[9] T.F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38(2-3):293–306, June 1985.

[10] G.W. Humphreys and H.J. Muller. Search via recursive rejection (serr): A connectionist model of visual search. *Cognitive Psychology*, 25:43–110, 1993.

[11] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *PAMI*, 20(11):1254–1259, November 1998.

[12] B. Julesz. A brief outline of the texon theory of human vision. *Trends in Neuroscience*, 7(2):41–45, 1984.

[13] C. Koch and S. Ullman. Shifts in selective visual attention: towards the underlying neural vircuity. *Human Neurobiology*, 4:219–227, 1985.

[14] A.N. Kolmogorov and V.M. Tikhomirov. *epsilon*-entropy and *epsilon*-capacity of sets in functional spaces. *AMS Translations. Series 2*, 17:277–364, 1961.

[15] R. Lienhart and J. Maydt. An extended set of Haar-like features for rapid object detection. *IEEE ICIP*, 1:900–903, December 2002.

[16] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91 – 110, 2004.

[17] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

[18] S. Minut and S. Mahadevan. A reinforcement learning model of selective visual attention. In *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 457–464, Montreal, Canada, 2001. ACM Press.

[19] K. Nakayama and G.H. Silverman. Serial and parallel processing of visual feature conjunction. *Nature*, 320:264–265, 1986.

[20] U. Neisser. *Cognitive Psychology*. Appleton-Century-Crofts, New York, 1967.

[21] S. Nene, S. Nayar, and H. Murase. Columbia object image library (coil-100). *Technical Report CUCS-006-96, Department of Computer Science, Columbia University*, February 1996.

[22] A. Papoulis and S.U. Pillai. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, New York, NY, USA, fourth edition, 2002.

[23] M.I. Posner, C.R.R. Snyder, and B.J. Davidson. Attention and the detection of signals. *Journal of Experimental Psychology: General*, 109(2):160–174, June 1980.

[24] R.P.N. Rao and D.H. Ballard. An active vision architecture based on iconic representations. *Artificial Intelligence*, 78(1–2):461–505, 1995.

[25] R.D. Rimey and C.M. Brown. Control of selective perception using bayes nets and decision theory. *IJCV*, 12:173–207, 1994.

[26] H. Rowley, S. Baluja, and T. Kanade. Rotation invariant neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1), January 1998.

[27] B.J Scholl. Objects and attention: The state of the art. *Cognition*, 80:1–46, 2001.

[28] K. Sung and T. Poggio. Example-based learning for view-based face detection. *IEEE Pattern Anal. Mach. Intell.*, 20:39 – 51, 1998.

[29] M.J. Swain and D.H. Ballard. Color indexing. *IJCV*, 7:11–32, 1991.

[30] M.J. Swain and M.A. Stricker. Promising directions in active vision. *IJCV*, 11(2):109–126, 1993.

[31] H. Tagare, K. Toyama, and J.G. Wang. A maximum-likelihood strategy for directing attention during visual search. *IEEE PAMI*, 23(5):490–500, 2001.

[32] A. Treisman and G.Gelade. A feature integration theory of attention. *Cognitive Psychology*, 12:97–136, 1980.

[33] A. Treisman and S. Gormican. Feature analysis in early vision: Evidence from search asymetries. *Psychological Review*, 95(1):15–48, 1988.

[34] J.K. Tsotsos. On the relative complexity of active versus passive visual search. *IJCV*, 7(2):127–141, 1992.

[35] J.K. Tsotsos, S.M. Culhane, W.Y.K. Wai, Y. Lai, N. Davis, and F.J. Nuflo. Modeling visual attention via selective tuning. *Artificial intelligence*, 78(1-2):507–545, 1995.

[36] P. Viola and M.J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137 – 154, May 2004.

[37] D.L. Wang. Object selection based on oscillatory correlation. *Neural Networks*, 12(4-5):579–592, 1999.

[38] L.E. Wixson and D.H. Ballard. Using intermediate objects to improve the efficiency of visual-search. *IJCV*, 12(2-3):209–230, April 1994.

[39] J.M. Wolfe. Guided search 2.0: A revised model of visual search. *Psychonomic Bulletin and Review*, 1(2):202–238, 1994.

[40] Rubner Y., Tomasi C., and Guibas L.J. The earth movers distance as a metric for image retrieval. *IJCV*, 40(2):99–121, 2000.

[41] A.L. Yarbus. *Eye Movements and Vision*. Plenum Press, New York, 1967.

A Appendix - Implementation details

This appendix discusses, in detail, the experiments described in Sections 6.3 and 6.4.

The segmentation process is an implementation of [2] in the openCV library. We use the function cvPyrSegmentation with parameters levels = 4, threshold1 = 255, threshold2 = 17.

To create the set of candidates, each segment is first bounded by a rectangle denoted r. Let $s_1(s_2)$ denote the short (long) side of r. Rectangles for which $s_1 < 12$ (too small) or for which $\frac{s_2}{s_1} > 3$ (too narrow) are ignored. Then, r is dilated in each direction by $\frac{1}{2}s_2$. The dilated candidate is denoted R. For each candidate we set a minimal $(s_1 \times s_1)$ and maximal $(1.5s_2 \times 1.5s_2)$ scale for faces to be detected inside this rectangle, depending on its size.

VSLE measures similarity between the candidates simply by comparing gray level histograms with 10 bins using L_2 norm. To get the effective histograms of the candidates, we first compute a Gaussian pyramid of the original image. Then, for each candidate, we compute its histogram from the pixels in the pyramid level in which the candidate's side s_1 is between 24 and 47.

The linear estimates are performed using only the 3-nearest-neighbors version of VSLE (Section 5.5). That is, in each iteration each unlabelled candidate is estimated based on the three labeled candidates that are most similar to it. VSLE starts (arbitrarily) from the first candidate (which corresponds to the image's most upper-left segment).

In Section 6.4 we used the extended implementation of VJA [15] provided by the OpenCV library. We did not train the classifier ourself, but used the default training results supplied by the OpenCV library (haarcascade_frontalface_default.xml). When applying the OpenCV detection function (cvHaarDetectObjects) on an image, it tries to detect faces of size 24×24 to faces of the image's size minus 10 pixels, jumping in scale factors of 1.1. In each scale the detection window is shifted by $s\Delta$ where $\Delta = 2$ or 4, depending on previous windows' results, and s is the scale factor.

When VSLE is incorporated with VJA, the same candidates and similarity measures described above are used. Each time VSLE decides to apply VJA on a candidate, it is applied on the sub-image corresponding to the candidate. The detection function was extended to be applied from the given minimal and maximal scales for detection, which are limited to fit the candidate's size as described above (from $s_1 \times s_1$ to $1.5s_2 \times 1.5s_2$). If VJA recognized at least one target inside this sub-image, VSLE treats the answer as 'yes'.

Before reporting detections and false detections at a given time, the following processing takes place: all the squares recognized as faces by the classification cascade are collected. Overlapping

groups of squares turn into one (average) square. Individual squares that were not recognized as part of a group are thrown out (to lower the false detection rate). This processing is part of the cvHaarDetectObjects function. When we use the VSLE+VJA version, we do not perform this processing on each candidate, but only on the whole image before reporting detection and false alarm rates.