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

Tamar Avraham and Michael Lindenbaum
Computer Science Department, Technion
Haifa, Israel 32000
tammya, mic@cs.technion.ac.il

May 22, 2003

Abstract

Visual search is required when applying a recognition process on a scene containing multiple objects. In such a case, 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), our basic hypothesis 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 were conducted where the statistical characteristics, the search algorithm, and the bound are evaluated and verified.
Contents

1. Introduction ........................................... 4

2. Related Work ........................................ 6

3. Framework ........................................... 11
   3.1. The context - A visual search task involving candidate selection and classification ................. 11
   3.2. Sources of information for directing the search ...................................................... 12
   3.3. Models of inner-scene visual similarity ................................................................. 13
   3.4. Dynamic vs. Static Search ............................................................... 14
   3.5. Algorithms ........................................ 14
   3.6. Measures of performance ......................................... 14

4. Deterministic Analysis of Visual Search Performance ................. 15
   4.1. Introduction ............................................ 15
   4.2. Notations .......................................... 16
   4.3. A measure for search difficulty combining targets isolation and candidates scattering ............... 17
   4.4. Bounds on search algorithms .................................................. 19
       4.4.1 No knowledge ...................................... 19
       4.4.2 A minimal knowledge: A lower bound \( d_0 \) on \( d_T \) (\( d_T > d_0 \)) .................... 19
       4.4.3 Bounded metric space ........................................ 20
       4.4.4 knowledge: \( d_T > d_0 \) and \( c_{d_0}(X) \) ........................................ 20
   4.5. Simple search algorithm and performance ............................................. 21
       4.5.1 Simple search algorithms ....................................... 21
       4.5.2 Bounded metric space ........................................ 22
       4.5.3 Sparseness in metric space ........................................ 23
       4.5.4 \( c_{d_0}(X) \) as an upper bound for search performance ........................................ 23
   4.6. Illustration of above results: searching for ladybugs in an insects surroundings .................... 24

5. Dynamic stochastic search algorithm ....................... 26
   5.1. The need to extend FLNN ...................................... 26
   5.2. Object labels as dependent random variables ...................................................... 26
   5.3. Dynamic search framework .................................... 27
   5.4. Minimum mean square error linear estimation ....................................................... 27
   5.5. The algorithm ...................................... 28
5.6. Combining prior information ........................................ 30

6. Experiments ............................................................. 30
   6.1. FLNN and minimum cover size .................................. 30
   6.2. Linear estimation experiments ................................... 32
   6.3. Using Top Down information .................................... 37
   6.4. Covariance vs. distance experiments ............................ 37

7. Discussion ............................................................. 38

References ............................................................... 41
1. Introduction

Visual search is required in situations where a person or a machine views a scene with the goal of finding one or more familiar entities. The highly effective Visual-search (or more generally, attention) mechanisms in the human visual system were extensively studied from Psychophysics and Physiology points of view. The most dominant study of human-visual-attention is Treisman and Gelade’s feature integration theory [TG80], dividing the attention process to a pre-attentive stage in which basic features are extracted and a focal attention stage in which attention is drawn to specific regions at a time. Search tasks are divided to pop-out tasks and serial search tasks. The HVS study that is most related to the work described in this paper was introduced by Duncan and Humphreys [DH89]. They propose an alternative hypothesis for search task efficiency based on similarity between scene objects and possible targets and on similarity between objects within the scene. Several models and computer implementations for attention mechanisms and visual search were implemented. Many of them focus on how and which pre-attentive features to extract and how to combine them to one saliency map that specifies the attention order. e.g. Koch and Ullman [KU85], Tsotsos et al. [TCW95], Itti, Koch, and Niebur [IKN98], Wolfe [Wol94]. In particular, a connectionist model SERR demonstrating the similarity based theory was suggested by Humphreys and Muller [HM93].

Some computer vision attention systems use other sources of knowledge to direct visual search: If we know for example that the sought for object is likely to be close to another, easier to find, object, then we may try to find the second object first and then look for the target only in a limited image region (Rimey and Brown [RB94], Wixon and Ballard [WB94]). Another method uses matching between image and object parts (Tagare, Toyama, and Wang [TTW01]). Minut and Mahadevan [MM01] improve the search performance using on-line training in a limited environment (room).

For more details on related work see section 2.

This study aims to develop a general framework for computer vision attention. Specifically, we set the goal as finding the target(s) as fast as possible, in minimal expected time. We focus on inner-scene similarity as available information for directing the search. 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 from a static 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 responses and the dependencies between candidates, the priority map is dynamically updated. We define the cost of an algorithm, as the number of queries to the recognizer module, until all targets are detected. 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. Meaning, given an input image (or a set of candidate), give a grade describing how hard (or easy) it is to locate the targets in it. To illustrate above principles as it is expressed in our suggested visual search algorithms, let us look at the following simple example: Seeking people in a natural scene including 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 people before considering another tree.

We take both deterministic and stochastic approaches. Under the deterministic approach, we analytically suggest bounds for performance of all search algorithms and suggest a simple algorithm that meets these bounds. The main suggested bound uses a methodology similar to Kolmogorov’s $\epsilon$-covering [KT61].

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 and feature-space-distance (used as a measure for similarity) is commonly a monotone descending function. We propose a linear estimation based search algorithm which can handle both similarity information and top-down information, when available.

In the Computer Vision field, attention and Visual Search is usually considered under the wider topic Active Vision [SS93]. Our algorithm can be described as a camera controlled active system in the following way: The camera first acquires a low resolution full scene image. This image is used for candidates selection, feature extraction and attention decisions. Each time the attention mechanism decides on the next candidate to be examined, the camera is directed to the center of this candidates and zooms in, acquiring a high resolution image of the candidate alone. This much wider information is used as input to the recognition module, providing the ability for a more precise and accurate identification. It is expected that such a system
will spend rather expensive time for each attended candidate. Therefore it illustrates the importance of a good dynamic candidate ordering, aiming to reduce such superfluous actions.

We start with a survey of previous related work (section 2). The context for visual search and some basic intuitive assumptions are described in section 3. The following section includes analytic analysis of search algorithms performance bounds. In section 4.4 we show some lower bounds for all search algorithms. In section 4.5 we describe two rather simple algorithms, show some upper bounds on their performance, and compare these bounds to the bounds we show in subsection 4.4. Section 5 describe the underlying probabilistic model and the resulting linear estimation based algorithm. In the experiments section we start with demonstrating the analytic deterministic results (section 6.1), and continue with demonstrating the stochastic algorithm’s effectiveness (sections 6.2 and 6.3) and a validation of the model (section 6.4).

2. Related Work

Visual-search and Attention mechanisms are widely studied from the Psychophysics and Physiology point of view, trying to understand the way nature deals with this task. Yarbus [Yar67] studied human perception of stationary images by tracking eye-movements. He describes three types of eye-movements that are combined during attention fixation: small involuntary saccades (very rapid simultaneous rotation of both eyes), a slow irregular movements of the optical axes, and an oscillatory movement of the axes of the eyes of high frequency but low amplitude. Yarbus suggests that all those small movements prevent the formation of an empty field during fixation on a stationary object. He shows that the eye fixations follow a path that goes through the salient features of the visual field. The eyes rest much longer on some elements of an image, while other elements may receive little or no attention. Yarbus’s experiments show that the number of details contained in an element of a picture does not determine the degree of attention attracted to it. Human eyes voluntary and involuntary fixate on those elements of an object which may carry essential information (for instance - when looking at a human face, an observer usually pays most attention to the eyes, the lips, and the noise). In addition, he shows that the purpose of the observer effects the fixation pattern.

Posner et al [PSD80] suggested an attentional mechanism able to move about the scene in a manner similar to a "spotlight". They dissociate the spotlight from foveal vision suggesting the spotlight mechanism is indepen-
dent of eye movements - it is possible to attend to an object while maintaining gaze elsewhere. The spotlight can be focused on a small area or diffusely spread over a large area but the total attention capacity is limited.

Neisser [Nei67] suggested that the 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.

Triesman and Gelade’s feature integration theory [TG80] formulate an hypothesis about how the human visual system performs pre-attentive processing. They hypothesized that some visual properties (such as color, orientation, spatial frequency, movement) are extracted early, automatically and in parallel across the visual field. Therefore, if a target is unique from its surround in such a feature, it pops-out without the need of focal attention. If there is no feature in which the target is unique from its surround, e.g. when the target is unique only in a conjunction of such features, the distinction cannot be detected pre-attentively, forcing 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. They describe how the visual scene is coded by feature-maps. Those maps are linked and form one master map of locations. The attention stage acts through this master map. Triesman [Tre85] also reveals asymmetry in search performance when the target and the background items are switched. When the target has an additional part or property, it acts as a pop-out. When the target lacks a property that the non-targets have, the search includes a scrutiny sequential scan.

While several aspects of the Feature Integration theory were criticized, the theory was dominant in Visual search research and much work was carried on to understand which features are pre-attentive (e.g. [Jul84],[TG88],[NS86]), which features are preferred by the HVS (e.g [Ca188]), how feature integration occurs, etc. (e.g. [KU85],[Wo94],[TCW+95]).

The HVS study that is most related to the work described in this paper was introduced by Duncan and Humphreys [DH89]. They rejected the dichotomy to 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 the prior knowledge about the possible targets. The other similarity is between the objects in the scene. They suggest the search procedures starts with a hierarchical segmentation, defining structural units,
which are the candidates competing for the attention resource. Units which are similar to the hypothesized 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 a weight is changed in one unit, the change propagates to the linked units as well. This way suppression may be spread among the units without treating every one of them individually. Thus, if all nontarget are homogeneous, they may be rejected together resulting in a fast (pop-out like) detection process, while if they are more heterogeneous the search is slower.

While the behavior in a case of a pop-out target is clear and not so interesting, Yokasawa and Lindenbaum [YL92] studied the damaging effect of a pop-out distractor on visual search efficiency. Concluding their psychophysical experiments, they suggest that after the pop-out was examined and found not to be the target, it is 'blocked' for a while, but then the temporary 'inhibition of return' ends and the distractor attracts the attention again. This procedure is repeated until the target is found.

Many models and computer implementations for attention mechanisms and visual search were implemented ([KU85], [HM93], [Wol94], [TCW+95], [IKN98], [Wan99], [AO91], [MS96], [vdPLHG97], [Sam], [Hor95], [Fuk87], [CR98]). Some of them are described below:

Koch and Ullman [KU85], using the Winner-Take-All neural network (term first used by Feldman [Fel82]) and Inhibition-Of-Return principle, suggest to construct one saliency map that 'grades' each location in the scene by the amount of overall conspicuousness. The Winner-Take-All network promises that only one location is active at a time. Inhibiting the selected location causes an automatic shift towards the next most conspicuous location. This saliency map combines the information of the individual (pre-attentive) feature maps. Abstractly, the serial search will 'visit' objects in the scene in the order dictated by their (descending) values in the master saliency map. Ullman and Koch's model also includes the rules of proximity and similarity preferences. Each time the focus shifts from a certain location, the next attended location will preferably be close and similar to the one just attended. (In this paper we claim, that in order to achieve good performance in a visual search task, it is preferable to do the opposite - prefer attention shifts to dissimilar locations until the first target is found). Koch and Ullman not only suggest how to implement a man-made such machine, but also suggest how such a network can exist in the biology system by proposing possible locations for such a saliency map in the visual pathway.

The SERR model proposed by Humphreys and Muller's [HM93] relies
on Duncan and Humphreys’s [DH89] theory. It uses an hierarchy neural network and a Boltzmann machine. Experiments show it matches reaction times measured by psychophysics experiments for character detection tasks, after the network’s weights and thresholds are set accordingly.

Wolfe in his Guided-search model [Wol94], adds and suggests that attention is drawn to peaks in the activation map that represent areas in the image with the largest combination of bottom-up saliency and top-down saliency. An item will activate a bottom-up map if it varies from its close neighborhood. An item will activate a top-down map if it has joint properties with possible targets. The activation map, according to Wolfe, is a signal indicates the likelihood of a target at each location. In his model, attention is deployed in order of decreasing activation, until the target is found or until the search is terminated. A search is terminated when all unvisited locations are with activity below a threshold or if run time exceeds a threshold. The weights given to each of the feature maps in order to combine them into one map is task dependent. Wolfe suggests, that even the same subject changes those weights over time, and by that can improve its efficiency to perform a task. Wolfe builds a computer simulation of this model, describing in details an algorithm for building an activation map for each feature, and for constructing the global activation map.

Tsotsos et al. [TCW+95], suggests a selective tuning model. The paper includes both the theory of the model being a hypothesis for primate’s visual attention, and the description of the computerized-implementation, claiming it can also act as a solution for robot vision implementation. It quite resembles Koch and Ullman’s model, though different in a few principles: The implementation of the winner-take-all principle is different and, according to the authors, fits better the knowledge on primate visual system. Proximity has no effect on choosing the next attended item. Their model includes a separate mechanism for detecting peripheral salient items, suggesting it models eye foveate saccade movements.

Itti, Koch, and Niebur [IKN98] recently implemented various approaches of visual attention and visual search. They provide a detailed mathematical description of the phases for building a saliency map - How to extract features so that saliency will be expressed, and how to normalize each feature map before summing them up. Itti [Itti00] checks and experimentally compares few options for weighting the feature maps. He suggests that one of the approaches is closest to the biology-visual-attention mechanism, but that a different one gives the best results for visual-search tasks.

Wang [Wan99] describes a neural network based on oscillatory correlation modelling attention shifts between objects. The oscillatory correlation
technique adds the property of grouping pixels into objects. The network selects the largest object each time. An inhibitory mechanism causes the objects of a scene to be scanned in an order of their descending size.

Most of the above models suggest that their computer implementation can serve both as a HVS model and a computer application for visual search. Nevertheless, their main goal is not an efficient computer vision machine. Below we describe some Computer Vision suggestions for efficient visual search: Rimoy and Brown [RB94] direct attention based on prior knowledge on relations like ‘part-of’ and ‘next to’ together with the information they gather during the process. A bayes-net is preprogrammed for the domain (domain’s are dinner tables, airport, road traffic etc.) to hold the prior information. Wixson and Ballard [WB94] analyze the contribution of spatial relationship knowledge to the efficiency of the search. They describe the indirect search framework. First an intermediate object, which can be recognized in low resolution, is located. Then only subregions that fulfill a known spatial relationship with the location of the intermediate object are scanned in order to find the desired target. They suggest some factors that effect the efficiency of such a search. Assuming the search task parameters’ probability density functions and the expected cost is known, they suggest a way to choose whether to perform a direct or an indirect search, and to decide which intermediate object based indirect search is most efficient.

Tsotsos [Tso92] deals with visual search performance in general. He analyzes the upper bounds of efficiency of visual search for passive vs. active search and bounded vs. unbounded search. In the passive case there is one input image. In the active case search is in a series of images, and the system can dynamically effect the choice of the subsequent images. Bounded search is the case in which a model of the target is known. In this case each candidate is a set of connected pixels that can be compared to that model. In unbounded search the model is not known. Each subset of pixels are possible to form a target. He proves that the upper-bound for unbounded search, both passive and active, is exponential in the image/s size, and that upper bounded search, both passive and active, is linear in the image/s size. A target is detected when the comparison grade exceeds a predefined threshold. In an active case the number of possible candidates can be reduced according to previous correspondence. Tsotsos concludes that “a vision system should be able to dynamically decide whether to employ an active or passive strategy, based on a number of decision dimensions, one of them being efficiency”.

Tagare, Toyama, and Wang [TTW01] use an attention based mechanism for directing the search. Their system uses the output of a "pre-attentive
system" that extracts basic features from the searched image. At each stage they choose to match an image region (feature) and model part with the greatest likelihood, and query a "Post-attentive System". They show that their algorithm behaves similar to biology studies theories in cases of pop-out targets vs. cases of camouflage targets.

Minut and Mahadevan [MM01] build an active system that automatically directs a camera to a target. First the camera acquires a low resolution image. Testing this image, the algorithm decides whether it includes candidates. This is done by comparing color histograms of regions in the image to color histograms of a low resolution model of the target. If it took the decision to check candidates in this region, it zooms in and compares the zoomed region to a high resolution model of the target, again using color histograms. If the target isn't found the algorithm chooses where to direct the camera next. Using symmetry properties it selects a point in the image that is most likely to be a center of an object. A learning agent manages states and actions. Each state is a cluster of similar images. Each action adds information to the agent, that will help it perform better decisions in the future. In order to show their agent learns and improves, they perform hundreds of epochs, each starting from a different random point.

Database Image Retrieval and Visual search are two Computer vision fields that are usually not related. We see some connection between the two types of search challenges, and believe that similar techniques can be borrowed from one type to the other. An example is the paper by Cox et al. [CMM+00] which describes the implementation and theory of a system for Image Retrieval based on user interaction. The user is interested in one target image in a big database. In each iteration the system displays few images. The user chooses which of these images are similar to the target using mouse clicks, and then invokes another iteration. Based on the user's response, the probability of each image in the database to be the target is estimated based on the bayes rule, and the next set of images to be displayed is selected. They implement two ways for choosing which images to display next: most-probable and most-informative, and show that in general, a most-informative approach attains better results.

3. Framework

3.1. The context - A visual search task involving candidate selection and classification

The task of looking for object/s of certain identity in a visual scene is often divided into two subtasks: One is to select sub-images which should be
considered as possible candidates. The other, the object recognition task, is to decide whether a candidate is a sought for object or not.

The input to the candidate selection task is the whole-scene image, and its output is a set of sub-images which we call candidates. 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 on the target character or just image based knowledge. The candidates may be of different size, may be bounded or unbounded [Tso92], and can also overlap.

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

For many algorithms, the candidate selection stage may be model dependent and may be based, for example, on invariants. Still, a classification/verification stage is usually needed to get a reasonable performance.

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 recognition stage is minimized.

3.2. Sources of information for directing the search

The knowledge for directing the search may be different in different contexts. In the simplest case no knowledge is available, and the strategy is to scan the candidates in some arbitrary order.

Several sources of information enabling more efficient search are possible:

1. Bottom-up saliency of candidates - In modelling the attention mechanism in the HVS, it is often claimed, for example, that dominant orientation, size, and color are used to assess a saliency measure [TG80][KU85][IKN98]. This measure usually quantifies how one candidate is different from the other candidates in the scene. Often, saliency has a major factor in directing attention to objects in a scene. Nevertheless, it can be sometimes misleading and may also not be applicable when, say, the
target’s uniqueness is defined by a combination of features, or when there are few resembling targets in the scene.

2. Similarity to the object we look for (top-down approach) [Wol94], [Itt00]. When prior knowledge on the targets is available, both the candidate selection stage and the attention stage may benefit from knowing it. The candidate selection stage can filter out irrelevant candidates. The remaining candidates may be ranked by their degree of consistency, or similarity with the target description. In many cases, however, it is hard to characterize the objects of interest visually in a way which is effective for discriminations and inexpensive to evaluate. Then, using top-down information may cost more than actually applying the high level recognition module on each candidate.

3. Mutual similarity of the candidate [DH89] - 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 to have the same/different identity.

In this work we focus only on the last source of information - mutual similarity between candidates, and assume that other information is not given. We leave the goal of integrating it quantitatively with other sources of information to future work.

3.3. Models of inner-scene visual similarity
We shall formalize the basic belief of mutual similarity in two alternative ways:

Deterministic: We shall 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 shall assume that dissimilarity vs. identity correlation is a steep monotone descending function.

As an attempt to make those assumption more than a heuristics, we conduct a set of preliminary experiments examining the behavior of dissimilarity vs. identity correlations.
3.4. Dynamic vs. Static Search

Usually, systems based on bottom-up or top-down approaches suggest to calculate a saliency map before the search starts, pre-specifying the scan order. In this case, the attentional process can be called static. (Usually, the changes in the maps during the search include only inhibition of the attended locations as a part of an algorithmic mechanism, preventing the attention to return to the same place again.)

The attention mechanism proposed here is dynamic in the sense that it allows the saliencies, or priorities, to be changed based on the results of the object recognition process. We show that the dynamic attention mechanism is preferable as it requires a lower 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, that is driven each 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 to be a target. Iteratively, the candidate with the highest priority receives the attention. The relevant sub-image is investigated by a high-level object recognition, which 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 difference between the derived algorithms is the way the priorities are estimated in each iteration.

3.6. Measures of performance

Given an ordering algorithm and a search task, some intuitive questions might come-up: when will the first target be found (after how many queries), how long will it take for half the targets to be found and how long for all. Knowing to predict such answers in advance can practically help stop the process of search before all candidates are examined.

As a preliminary approximation, we assume that only the costs associated with calling the recognition oracle are substantial. That is, using all the
other information, if available, calculating similarity and dissimilarities and calculating the ordering itself, are done in negligible computational effort. (This is justified by the use of features that can be extracted and compared efficiently and even in parallel).

We consider several measures. Under the deterministic assumption, analytically we show upper bounds (or worst case) for the cost our suggested algorithm pays for finding the first target. We prove that any other ordering algorithm can’t promise better results.

Since our cost measures ‘grade’ a combination of algorithm and input scene, it provides: a) Measures for algorithm performance - enabling comparisons between different search algorithms. b) Measures of task complexity - enabling to compare between different search tasks.

4. Deterministic Analysis of Visual Search Performance

4.1. Introduction

As mentioned above, we suggest to direct visual search tasks using inner-scene mutual similarity. Here we suggest measures that depend on the search task and quantify how difficult is such a similarity based search.

The simplest search situation is one in which there is one target in the scene, all non-target candidates are mutually-similar and are dramatically different from the target (one book on a table with some pens and pencils). The hardest case can be a situation in which all mutual similarities of pairs of candidates are alike (there are no two candidates that are significant similar or significantly different). In such a case the mutual inner-scene similarity is un-informative. Such situations happen when all candidates are very similar (some books on a table all in same size and color, the target being a specific book) or when all candidates are different from each other (a table containing a bunch of office equipment). Intermediate difficulty search task is, for example, when the candidates can be divided into few groups, each group members having strong mutual similarity (a table with few books, few pencils, few staplers and few cups). One of the groups holds all the targets, but we don’t know which. Using mutual similarity information, the uncertainty of the problem is reduced from number of candidates to the number of groups.

Our main goal is to quantify the difficulty of a search task. We consider several characteristics which can be evaluated for every task and show that indeed they determine the difficulty in a quantitative way: We show that the search performance is bounded by a value which changes significantly
with this characteristic. We also provide a search algorithm and show that its performance meets those bounds, implying that the bounds are tight.

We consider the situation where information on the search task is provided before it is executed (from pre-examining the specific task, or from prior information on typical search tasks in a specific context). As we will see, the more we are pre-informed, the more accurate we can characterize the task’s difficulty. Practically, the proposed maximally characterizing knowledge is not always available. Therefore, we consider other characteristics, which rely on weaker knowledge.

The rest of this section is as follows: After some notations, we present the tightest bound of the search difficulty. This bound relies on more information than the others and uses a characteristic denoted min-df-cover (subsection 4.3). Then we start with the case where no information is available, and gradually extend the available information on the task. For each characteristic, we provide bounds on search difficulty and prove that all search algorithms can’t promise (worst case) performance that exceeds those bounds (section 4.4). In section 4.5 we describe a simple algorithm and prove its performance meets the same bounds.

4.2. Notations

A search task is described by 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 \(l : X \rightarrow \{T, D\}\) is a function assigning identity labels to the candidates. \(l(x_i) = T\) if the candidate \(x_i\) is a target, and \(l(x_i) = D\) if \(x_i\) is a non-target (or a distractor).

An attention, or search algorithm, is provided with the set \(X\), but not with the labels \(l\).

The cost of a search algorithm \(A\), \(\text{cost}_q(A, X, l)\) is defined as the number of queries to the recognizer oracle, until the goal of the search \(q\) is achieved. The goal of the search can be different depending on the application: in many search tasks finding one target is the goal (a customer looking for a certain product on a supermarket shelf, or a guy in a cafeteria looking for a date for Saturday), sometimes a certain number or certain percentage of the targets can be the goal (A host looking for cups in his kitchen cabinet in order to prepare coffee for his three guests), and sometimes one will not be satisfied unless he knows all the targets are detected (a mother looking for her five children). In the following sections we concentrate on \(\text{cost}_1(A, X, l)\) - the cost of finding the first target.

We refer to the set of partial descriptions \(X = \{x_1, x_2, \ldots, x_n\}\) as points
in a metric space \((S,d)\), \(d : S \times S \to \mathbb{R}^+\) being the metric distance function. The metric function \(d\) 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 (\(\forall s_i, s_j, s_k \in S, d(s_i,s_j) + d(s_j,s_k) \geq d(s_i,s_k)\)).

The partial description can be, for example, feature vectors in a Euclidian space.

4.3. A measure for search difficulty combining targets isolation and candidates scattering

The proposed measures for task difficulty combine two main factors:

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

Intuitively, the more the targets are different from non-targets, the search is easier. But, if the non-targets are also different between themselves, the search becomes hard again. [DH89]

A useful quantization 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 4.1** 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 \(S\). \(^1\) \(C \subset 2^S\) is a ‘cover’ of \(X\) if \(\forall x \in X \exists C \in C\) s.t. \(x \cup C \neq \emptyset\).

**Definition 4.2** \(C \subset 2^S\) is a ‘\(d_0\)-cover’ of a set \(X\) if \(C\) is a cover of \(X\) and if \(\forall C \in C\) \(\text{diameter}(C) \leq d_0\), where \(\text{diameter}(C) = \max_{c_1,c_2 \in C} d(c_1,c_2)\).

**Definition 4.3** A ‘minimum-\(d_0\)-cover’ is a \(d_0\)-cover with a minimal number of elements. We shall use the notation \(C_{d_0}(X)\) for some particular minimum-\(d_0\)-cover, and denote its size by \(c_{d_0}(X)\).

For \(X\) being a set of feature vectors in a Euclidian space, for example, \(c_{d_0}(X)\) will be the minimum number of \(m\) spheres with diameter \(d_0\) required to cover all candidates in \(X\).

The above definitions follow Kolomgorov’s \(\epsilon\)-covering [KT61]. Similar cover definitions are used in learning theory for learning with respect to

\(^1\) \(2^S\) is a common notation for the set of all possible subsets of \(S\). \(2^S\) is commonly called ‘the power set of \(S\)’.

17
fixed distributions [BI91] and for a learning by distances model [BDIK95] in which the learning process hypotheses are points in a metric space. The teacher responses include information on the metric-space-distance of each hypothesis to the target.

**Definition 4.4** Given a search task \((X,l)\), let the ‘max-min-target-distance’, denoted \(d_T\), be the largest 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 few targets in the scene \(T_1, \ldots, T_m\), \(d_T = \max_{i=1}^{m}(d_{T_i})\).

**Theorem 4.1** Given a search task \((X,l)\), so that \(d_T\) is its max-min-target-distance, the difficulty of finding the first target depends on the minimum \(d_T\)-cover size, \(c_{d_0}(X)\) in the sense that:

1. Any search may need to query the oracle for at least \(c_{d_0}(X)\) candidates before finding a target.
2. There is an algorithm that needs no more than \(c_{d_0}(X)\) queries for finding the first target.

The proof to the two sections of the theorem will be given in the following two subsections.

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. (Let \(d_T\) be the distance of the target from the nearest non-target, and let us assume that the distances between each two non-targets < \(d_T\). all non-targets can be covered together by one covering set, and the target alone is covered by a second covering set). In a case where all \(n\) candidates are scattered and all mutual distances are \(\geq d_T\), \(c_{d_0}(X)\) is \(n\). When there are \(k\) groups of candidates so that all inner-group distances < \(d_T\) and all outer-group distances \(\geq d_T\), \(c_{d_0}(X)\) is \(k\).

The above examples are trivial extreme cases in which the difficulty is revealed easily. \(c_{d_0}(X)\) also provides a measure for intermediate situations which 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 those groups (or more), the cover size, \(c_{d_0}(X)\), will be relatively big, characterizing the search difficulty correctly.
In the next two subsections, we prove the above theorem, and also provide some less tight measures that rely on weaker information.

4.4. Bounds on search algorithms

This section presents lower bounds on the performance ability of all search algorithms. LB is a lower bound for search performance if for any algorithm \( A \), there is an input \( X \in \mathcal{X} \) and an assignment \( l \in \mathcal{L} \) so that \( \text{cost}_1(A, X, l) \geq \text{LB} \), where \( \mathcal{X} \) is some relevant distributions of candidates and \( \mathcal{L} \) a limitation of the possible assignments.

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

4.4.1 No knowledge

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 \( \pi \) be this resulting ordering, or permutation, of \( X \).

**Claim 4.2**

\[ \forall A, \forall X, \exists l, \text{cost}_1(A, X, l) \geq n \]

where \( \text{cost}_1(A, X, l) \) is the number of queries before the first target is found, and \( n = |X| \).

or in words: for any algorithm and any set of candidates, there is a labelling for which the target is the last candidate examined.

**Proof:** Given an algorithm \( A \) and an input \( X \), a permutation \( \pi = x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n} \) may be specified. Choose the assignment \( l \) to be

\[ l(x) = \begin{cases} T & x = x_{\pi_n} \\ D & \text{otherwise} \end{cases} \]

4.4.2 A minimal knowledge: A lower bound \( d_0 \) on \( d_T \) (\( d_T > d_0 \))

Suppose we know that at least one target is dissimilar from the non-targets. Quantitatively, we consider the case where a lower bound \( d_0 \) satisfies \( d_T > d_0 \). Let \( \mathcal{L}_{d_0} \) denote the set of such possible assignments.
Claim 4.3 Let $d_i$ denote the distance of candidate $x_i$ to its nearest neighbor 

$$\forall A \forall X \exists l \in \mathcal{L}_{d_0} \ cost_1(A, X, l) \geq |\{x_i \in X \mid d_i > d_0\}|$$

Proof: Under the limitation of $\mathcal{L}_{d_0}$, all candidates for which $d_i > d_0$ can be possible targets. Given an algorithm $A$ and an input $X$, a permutation $\pi = x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n}$ is defined. Choose $l$ to assign $T$ only to the latest candidate in $\pi$ that satisfies $d_i > d_0$. 

Note that in the worst case, this bound does not give any improvement over claim 4.2. There may be a case in which all candidates satisfy $d_i > d_0$ and the bound will be $n$ again.

4.4.3 Bounded metric space

The worst situation described above cannot happen if we limit the candidates to be taken from a bounded $m$ dimensional feature space where, say, the feature values are in $[0,1]$. For simplicity of calculations, let us consider the $L_\infty$ metric $^2$. Similar results can be computed for other metrics.

Let $\mathcal{X}_m = \{X \mid x \in X \Rightarrow x \in [0,1]^m\}$.

Claim 4.4

$$\forall A \exists X \in \mathcal{X}_m, l \in \mathcal{L}_{d_0} \ cost_1(A, X, l) \geq LB_m = \min(\left[\frac{1}{d_0}\right]^m, n)$$

Proof: Choose the set of candidates, $X$ on a grid such that the distance between grid points is more than $d_0$ (see figure 1). There are $\left[\frac{1}{d_0}\right]^m$ such points. Divide the $n$ candidates equally between the grid points. Given an algorithm $A$ defining the permutation $\pi$, choose the assignment $l$ that assigns $T$ to the group of candidates located in the grid point that its first appearance in $\pi$ is last.

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

In this section we use information about the distribution of the candidates (by the minimum-$d_T$-cover size, $c_{d_0}(X)$) in addition to the known lower bound $d_0$ on $d_T$, and provide the proof to the first part of theorem 4.1.

$^2$In $L_\infty$ metric, the distance between two vectors is their maximal difference in a single component.
Let $\mathcal{X}_{d_0,c} = \{X \mid c_{d_0}(X) = c\}$ describe the possible sets of candidates that can be covered by $c$ subsets so that each subset diameter doesn’t exceed $d_0$.

Claim 4.5

$$\forall A \exists X \in \mathcal{X}_{d_0,c}, l \in \mathcal{L}_{d_0} \quad \text{cost}_1(A, X, l) \geq LB_{d_0,c} = c$$

Proof: Choose $c$ points in the metric space, so that all the inter-point distance is more than $d_0$. Choose the $n$ candidates to be divided equally between these locations. Choose $l$ that assigns $T$ to the group of candidates located in the point that its first appearance in $\pi$ (the ordering dictated by algorithm $A$) is last.

The proof is clearly similar to that of claim 4.4. In fact the grid built in claim 4.4 is a set of points associated with a cover of size $\lceil \frac{1}{d_0} \rceil^m$. Note however that no specific metric is considered in the more general theorem.

4.5. Simple search algorithm and performance

In this section we present a simple algorithm and show that its performance never exceeds 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 a set of candidates $X = \{x_1, \ldots, x_n\}$, compute the distance $d_i$ of each candidate $x_i$ to its nearest neighbor. Order the candidates by descending $d_i$ and query the oracle by this order until the first target is found.

FLNN- Farthest Labelled Nearest Neighbor: Given a set of candidates $X = \{x_1, \ldots, x_n\}$, choose randomly the first candidate, query the oracle and label this candidate. Repeat iteratively, until a target is detected: for each unlabelled candidate $x_i$, compute the distance $dL_i$ to the nearest labelled neighbor. Choose the candidate $x_i$ for which $dL_i$ is maximum. Query the oracle to get its label.
It is easy to see that FNN is optimal for the criteria of label assignments $l \in L_{d_0}$ we used in section 4.4 when there is always only one target in the scene.

When there is more than one target in a scene, targets are likely to be similar. In such a situation FNN will perform badly. Let us take, for instance, a simple and realistic case in which there are two targets, similar to each other more than all distractors are between themselves. FNN will detect the targets last.

Therefore, we consider FLNN as a more general solution. In the following sections we show that FLNN performs below the mentioned bounds for any number of targets in the scene.

### 4.5.2 Bounded metric space

As in section 4.4.3, we consider candidates that belong to the bounded metric space $[0, 1]^m$, and use the $L_\infty$ metric. We also assume a lower bound $d_0$ on $d_T$.

**Claim 4.6**

$$\forall X \in X_m, l \in L_{d_0} \ cost_1(FLNN, X, l) \leq UB_m = \min\left(\frac{1}{d_0^m}, n\right) = LB_m$$

or in words: if there is a target that the closest distractor is distant more than $d_0$ from it, FLNN will find the first target after at most $UB_m$ queries.

**Proof:** Since $l \in L_{d_0}$, there is a target $x_i$ that the distance to its nearest distractor is $d_T > d_0$. Imagine the hypercube $H$ with sides in length $2d_0$ so that $x_i$ is its center (note it is possible that part of $H$ exceeds $[0, 1]^m$). $H$ includes only targets. Now we can cover the rest of the metric space $[0, 1]^m$ with at most $\left[\frac{1}{d_0}\right]^m - 1$ hypercubes with sides of length $d_0$.

According to FLNN’s behavior, it will not query two candidates in one hypercube (that are distanced at most $d_0$ one from another) before it queries at least one candidate in $H$. Therefore, a target (not necessarily $x_i$) will be located after at most $\left[\frac{1}{d_0}\right]^m$ queries.

Of course, if the total number of candidates, $n$, is even smaller, $n$ queries always suffice.

\[\square\]
4.5.3 Sparseness in metric space

In the above proof, we count all hypercubes dividing \([0,1]^m\). Given \(X\), we can lower the bound on FLNN’s performance by counting only the hypercubes that contain candidates.

**Claim 4.7** Let \(NEH(X,d_0)\) be the number of hypercubes out of the \([0,1]^m\) that divide the feature space \([0,1]^m\), that contain candidates. (NEH stands for non-empty-hypercubes).

\[
\forall X \in X_m, \ l \in \mathcal{L}_{d_0} \ cost_1(FLNN, X, l) \leq UB_{NEH} = \min(NEH(X,d_0), n)
\]

**Proof:** Use exactly the same arguments as in the proof to claim 4.6, only don’t count the empty hypercubes.

4.5.4 \(c_{d_0}(X)\) as an upper bound for search performance

A generalization of claim 4.7 to a general metric space completes the proof of theorem 4.1:

**Claim 4.8**

\[
\forall X \in X_{d_0,c}, \ l \in \mathcal{L}_{d_0} \ cost_1(FLNN, X, l) \leq UB_{d_0,c} = c = LB_{d_0,c}
\]

or in words: if there is a target which is more than \(d_0\) far from all the distractors, and all the candidates can be covered by \(c\) sets with diameter at most \(d_0\), FLNN will find the first target after at most \(c\) queries.

**Proof:** Take an arbitrary minimum-\(d_0\)-cover of \(X\), \(C_{d_0}(X)\). Let \(x_i\) be a target so that \(d(x_i, x_j) > d_0\) for every distractor \(x_j\). Let \(C\) be a covering element (\(C \in C_{d_0}(X)\)) so that \(x_i \in C\). Note that all candidates in \(C\) are targets. Excluding \(C\), there are \((c-1)\) other covering elements in \(C_{d_0}(X)\) with diameter \(\leq d_0\). Since \(C\) contains a candidate that its distance from all distractors > \(d_0\), FLNN will not query two distractor-candidates in one covering element (that their distance \(\leq d_0\)), before it queries at least one candidate in \(C\). Therefore, a target will be located after at most \(c\) queries. (it is possible that a target that is not in \(C\) will be found earlier, and then the algorithm will stop even before).
The minimum cover size, $c_{d_0}(X)$, can be up to $2^m$ times smaller than the number of non-empty hypercubes, $\text{NEH}(X, d_0)$ \footnote{Consider a corner of one of the covering hypercubes discussed in the proof to claim 4.6. This corner is also a corner of $2^m - 1$ other neighboring hypercubes. Imagine there is one candidate at each of these $2^m$ cubes, very close to that corner. In such a case occupied($X, d_0$) is $2^m$ while $c_{d_0}(X)$ is 1.}, and always not greater than occupied($X, d_0$). Therefore claim 4.8 is an improvement over claim 4.7. Nevertheless, while $\text{NEH}(X, d_0)$ can be calculated easily, computing $c_{d_0}(X)$ can be hard for a big set of candidates. The problem of finding the minimum cover is NP-hard. Gonzalez [Gon85] proposes a 2-approximation algorithm for the problem of clustering a data set minimizing the maximum inner-cluster distance, and proves it is the best approximation possible if $P \neq NP$.

4.6. Illustration of above results: searching for ladybugs in an insects surroundings

We demonstrate the latest result using the “insects” example (see figure 2). More realistic examples are described in section 6.

The image contains 7 different insects, which are roughly segmented. For simplicity of the demonstration, the search uses only the objects area as a partial description (feature). See the distribution of the objects in the one-dimensional feature space in figure 2c.

Taking ‘ladybug’ as the target the minimum-$d_0$-cover is of size 3 \footnote{As mentioned above, finding the minimum cover is NP-hard when number of dimensions is $\geq 2$. For the one dimensional case, we can compute the minimum-$d_0$-cover using the following simple algorithm: The first covering section will start from the leftmost candidate and end in distance $d_0$ right to it. The next covering section will start from the location of the leftmost uncovered candidate etc., until all candidates are covered.}. This promises that using FLNN algorithm, the ladybug will be detected after examining at most 2 other objects.

At this stage we would like to add some comments: 1) The features chosen have a crucial effect on the search results. The same effect will be expressed in the value of the minimal cover. Therefore, the upper bounds we suggest actually ‘grade’ the ‘amount’ the features fit the task. 2) If the image had some more butterflies and some more small creatures, like ants, $c_{d_0}(X)$ for the target ‘ladybug’ would not have been changed. Meaning, the search difficulty wouldn’t have grown with additional candidates. 3) If there where some more ladybugs in the image, that would have added points in the feature space very close to the current existing ladybug, using FLNN algorithm we could still promise that one of the ladybugs would have found after the same number of queries promised before. 4) Imagine there was an original
Figure 1: grid points in a 2D bounded metric space. $m = 2, d_0 = \frac{1}{2} - \varepsilon$

Figure 2: 'insects': Image for demonstrating the cover upper bound. (a) the original 'insects' image. (b) the 'insects' image after a rough segmentation. (c) insects in one dimensional feature space. feature = object area
image containing more objects such as big animals, trees, houses etc., and that the insects image is an output of a prior process that filtered out only insects. If we loose the prior process and immediately performed the search, the minimum cover would have increased only by 1. This comes to illustrate that when using a dynamic attention ordering as we suggest we can be less picky in the candidate selection process, and leave more candidates without paying a large price in the search.

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 based on linear estimation minimizing the mean square error [PP02].

5.1. The need to extend FLNN

In section 4 we took a deterministic approach, and based our search strategy on the assumption that a target is likely to be different from all distractors. In the algorithm suggested, FLNN, this assumption is expressed preferring the candidate that is most different from the nearest recognized distractor. FLNN stops after finding the first target. We wish to extend it so it will continue and find more targets after the first is located. According to our assumptions, the consequent targets will be close (in feature-space distance) to the found targets, and far from the attended distractors. The question is how to quantify these distances and how to combine them to one value indicating the likelihood of a candidate to be a target. The second major weakness of the FLNN algorithm is that it lacks robustness. A single “error” in the form of an attended distractor close to an unattended target, will reduce the priority of this target and make the search slow.

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 help in directing the search.

Intuitively we know that objects associated with similar identity (or category) tend to be visually similar more than objects which are of 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.

\[
\text{cov}(l(x_i), l(x_j)) = \gamma(d(x_i, x_j))
\]

where \(l(x_i)\) and \(l(x_j)\) are the labels of candidates \(x_i\) and \(x_j\), \(d(x_i, x_j)\) is the feature space distance between the two candidates, and \(\gamma\) is a monotone descending function.

In section 6.4 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 exponential descending \(\gamma\) is a good approximation to 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 to infer unknown labels from known ones, using common estimation techniques.

### 5.3. Dynamic search framework

We propose the following greedy approach to dynamic search: At each iteration, estimate the probability of each unlabelled 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 on the corresponding sub-image.

After the \(m\)-th iteration, \(m\) candidates, \(x_1, x_2, \ldots, x_m\), were already attended and \(m\) labels, \(l(x_1), l(x_2), \ldots, l(x_m)\) are known. We use them to estimate the conditional probability of the label \(l(x_k)\) of each unlabelled candidate \(x_k\) to be 1.

\[
p_k = p(l(x_k) = 1 \mid l(x_1), \ldots, l(x_m))
\]

### 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 to take 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 \( l(x_1), l(x_2), \ldots, l(x_m) \), we seek a linear estimate \( \hat{l}_k \) of the unknown random variable \( l(x_k) \),

\[
\hat{l}_k = a_0 + \sum_{i=1}^{m} a_i l(x_i)
\]

which minimizes the minimum mean square error \( e = E((l(x_k) - \hat{l}_k)^2) \). This is a standard task with a known solution:

\[
\hat{l}_k = E[l(x_k)] + \tilde{a}^T (\tilde{l} - E[\tilde{l}])
\]

where \( \tilde{l} = (l(x_1), l(x_2), \ldots, l(x_m)) \) and the vector \( \tilde{a} \), composed of the coefficients \( a_i, i = 1, \ldots, m \), specifies by solving the following set of (Yule-Walker) equations [PP02]

\[
\tilde{a} = R^{-1} \tilde{r}
\]

where \( r_i, i = 1, \ldots, m, R_{ij}, i, j = 1, \ldots, m \) are given by

\[
R_{ij} = E[(l(x_i) - E[l(x_i)])(l(x_j) - E[l(x_j)])] = \text{cov}(l(x_i), l(x_j))
\]

\[
r_i = E[(l(x_k) - E[l(x_k)])(l(x_i) - E[l(x_i)])] = \text{cov}(l(x_k), l(x_i))
\]

The estimated label \( \hat{l}_k \) is the conditional mean of a label \( l(x_k) \) of an unclassified candidate \( x_k \), and therefore may be interpreted as the probability of \( l(x_k) \) to be 1

\[
p_k = p(l(x_k) = T \mid l(x_1), \ldots, l(x_m)) \sim \hat{l}_k
\]

5.5. The algorithm

- **Given a scene image, choose \( n \) sub-images to be 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 $x_k$ out of the $n - m$ remaining candidates, estimate $\hat{l}_k \in [0,1]$ based on on the known labels $l(x_1), \ldots, l(x_m)$.
  - Query the oracle on the candidate $x_k$ for which $\hat{l}_k$ is maximum.
  - If enough targets were 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).

2. Our goal is to minimize the expected search time, and the proposed algorithm, being greedy, cannot achieve an optimal solution. It is however optimal with respect to all other greedy methods, as it uses all the information collected in the search to make the decision.

3. 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 if the prior probability of all candidates to be target is uniform. This however seems to be an unjustified assumptions as targets usually do not appear in large groups. (If they would then saliency would never work.)

4. The estimation of the label associated with any unknown candidate is accelerated if only its nearest (classified) neighbors are used. As the covariance decreases with distance, ignoring far neighbors may be justified.

5. In tasks where the target differs significantly from the non-targets, and the non-targets are similar, we get a pop-out like behavior, and the target is found after one or two iterations.

6. If the non-targets consist of several $k$ 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.

7. If the targets themselves are one cluster, then after the first is detected, the rest follow immediately.

In the next section we describe some visual search experiments.
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. See section 6.3.

6. Experiments

6.1. FLNN and minimum cover size

Our first set of experiments use the Columbia Object Image Library COIL-100 [NNM96] containing 100 gray level images of various objects. We can think about these images as candidates extracted from some larger image or to refer to a mosaic of them, denoted \textit{columbia100}, (figure 3) as the image we look at.

![Figure 3: columbia100 - test image constructed from the COIL-100 Database images](image)

The extracted features are gaussian derivatives, based on the filters described in the work of Rao and Ballard 1995 [RB95]. We use two first order gaussian derivatives (at 0 and 90 degrees), three second order gaussian derivatives (at 0, 60 and 120 degrees), and four third order gaussian derivatives (at 0, 45, 90 and 135 degrees). Each of them at 5 scales, giving a total of 45 filters, resulting feature vectors of length 45 per candidate. See figure 4. Euclidean metric is used for measuring distance between two feature vectors.
First we choose *cups* as the targets category (10 out of the 100 candidates are cups). We perform all possible runs of FLNN - since FLNN’s first step is random, there are 100 different possible runs (starting from a different candidate each run). The result of a run is the time (or number of queries) that takes to find the first target. In the worst case it takes 18 queries, in the best case 1 query (when the first randomly chosen candidate is a target), the mean result is about 9.

For this same search task (columbia100 image, candidates are the 100 objects, targets are the 10 cups, gaussian derivation feature vectors extracted, Euclidean distance computed) we compute an approximation of the minimum \(d_T\) cover size. Since computing the optimal result is NP-complete, we use the following heuristic procedure as which we found, gives a close upper bound for the real value: First we compute \(d_T\), as it is defined in definition 4.4. Then we choose a candidate for which the number of \(d_T\)-close candidates is minimal. We check if this set of candidates can be covered by one covering element (in many cases they do). if not, we remove items from this set until its diameter doesn’t exceed \(d_T\). The chosen set is removed from the set of whole candidates, and the procedure is repeated, until all candidates are covered. The number of covering elements used in this process is the result of this heuristic algorithm. (We have also implemented the 2-approximation algorithm suggested by Gonzalez [Gon85] and applied it on this experiment and on the other experiments described below. The results of the algorithm described above gave a better result for all cases).
For the case of the cups search task, the resulting approximated cover is 24. We see that all runs of FLNN don't reach this bound, but some get quite close to it, implying that the bound is tight.

Still using the columbia100 image, we repeat the above set of experiments (all runs of FLNN and calculation of cover size) for the case where the targets category is toy cars, and for the case where targets category is toy animals. The results are summarized in table 1 (first three rows) and in figure 6.

Comparing the three above cases, we can see that for the cars case we get the worst performance, while for the two other cases results are similar. In the cars case, the targets are very similar to each other, which should ease the search. However, finding the first car is hard since there are some distractors which are very similar to the cars. (Here \(d_T\) is small). In the cups case most cups are similar and they are all quite different from distractors, causing the task of finding the first cup to be of reasonable difficulty. For the toy animals, case the results are similar to the cups case, but not due to the same reasons. The different toy animals have no resemblance between themselves, but one of them (the duck) is very different from all candidates and finding it is easy. (Here \(d_T\) is large).

The next two experiments use different input images, a different method for candidates selection, and different type of feature vectors. The two images, the elephants image and the parasols image, are taken from the Berkeley land segmented Database [MFTM01]. See figure 5 for the two images and the hand segmentations we choose to used. Segments under a certain size are ignored, leaving us with 24 candidates in the elephants image and 30 candidates the parasols image. The targets are the segments containing elephants and parasols respectively.

For those colored 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 of a colored image. Each of these two dimensions are divided into 8 bins. This results a 2D histogram with 64 bins, i.e. - a feature vector of length 64. Again, we use Euclidean metric for distance measure. (we tried also other histogram comparison methods, such as the ones suggested by Swain and Ballard in [SB91], but the results where similar). The results for those two search tasks are also added to table 1 and figure 6.

6.2. Linear estimation experiments
The Linear estimation based algorithm described in section 5 was implemented and applied to the same five visual search tasks described in section
Figure 5: The *elephants* and *parasols* images taken from the Berkeley hand segmented database and the segmentations we have chosen to use in our experiments. (colored images)

<table>
<thead>
<tr>
<th>Image</th>
<th>Targets</th>
<th># of candidates</th>
<th># of targets</th>
<th>FLNN best</th>
<th>FLNN worst</th>
<th>FLNN mean</th>
<th>$d_T$</th>
<th>calculated Cover size</th>
</tr>
</thead>
<tbody>
<tr>
<td>columbia100</td>
<td>animals</td>
<td>100</td>
<td>7</td>
<td>1</td>
<td>22</td>
<td>9.06</td>
<td>0.2937</td>
<td>25</td>
</tr>
<tr>
<td>columbia100</td>
<td>cups</td>
<td>100</td>
<td>10</td>
<td>1</td>
<td>18</td>
<td>8.97</td>
<td>0.3030</td>
<td>24</td>
</tr>
<tr>
<td>columbia100</td>
<td>cars</td>
<td>100</td>
<td>10</td>
<td>1</td>
<td>73</td>
<td>33.02</td>
<td>0.1094</td>
<td>79</td>
</tr>
<tr>
<td>Berkeley-elephants</td>
<td>elephants</td>
<td>24</td>
<td>4</td>
<td>1</td>
<td>9</td>
<td>5.67</td>
<td>0.2520</td>
<td>9</td>
</tr>
<tr>
<td>Berkeley-parasols</td>
<td>parasols</td>
<td>30</td>
<td>6</td>
<td>1</td>
<td>6</td>
<td>3.17</td>
<td>0.3136</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 1: Experiments results for FLNN and cover size
6.1. We model the dependencies of the covariance on the feature-space-distance using \( \text{cov}(l(x_1), l(x_2)) = \gamma(\text{d}(x_1, x_2)) = \mu (1 - \mu) e^{-\text{d}(x_1, x_2)} \), where \( l(x_i) \) is the label of candidate \( x_i \), \( d(x_i, x_j) \) is \( x_i \) and \( x_j \) feature-space distance, and \( \mu = E(l(x_i)) \).

Unlike FLNN which deals only with finding the first target, the linear estimation algorithm continues and aims to find also the other targets. It is possible to stop the search at any time (for instance - after a certain number of targets are detected). For the sake of this study, we don’t stop the run, and let the algorithm scan all the candidates. The solid lines in figures 7 and 8 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.

In all above experiments the candidates of each task can be divided into few weakly homogeneous groups (or clusters), which seems to be the situation is in most natural scenes as well. In order to compare between the results of the experiments qualitatively, we focus on the two following difficulty factors: \( d(T,N,T) \)- 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.

Note that in all above experiments the linear estimation based algorithm
Figure 7: Linear estimation algorithm results for columbia100 image. 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 target found) space. In (a) the cars are the targets, the first car is quit hard to find. Once it is found the rest are detected fast. In (b) cups are the targets, it is easy to find the first cup since most cups are different from non-targets. Most cups resemble and follow pretty fast, but there are two cups (one without a handle and one cup 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 between the different toy animals, therefore the rest of the search performs similarly to a random search procedure.
Figure 8: Linear estimation algorithm results for Berkeley hand segmented color images. 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 target found) space. In (a) 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 (b) all the parasols are detected very fast, since their color is similar and differs from that of all other candidates.

<table>
<thead>
<tr>
<th></th>
<th>d(T,T)</th>
<th>d(T,N,T)</th>
<th>Finding first target</th>
<th>Finding successive targets</th>
</tr>
</thead>
<tbody>
<tr>
<td>cars</td>
<td>small</td>
<td>small</td>
<td>hard</td>
<td>easy</td>
</tr>
<tr>
<td>cups</td>
<td>some small, some big</td>
<td>big</td>
<td>easy</td>
<td>easy to find some. hard to find all</td>
</tr>
<tr>
<td>animals</td>
<td>big</td>
<td>big</td>
<td>easy</td>
<td>hard</td>
</tr>
<tr>
<td>elephants</td>
<td>small</td>
<td>intermediate</td>
<td>easy</td>
<td>intermediate</td>
</tr>
<tr>
<td>parasols</td>
<td>small</td>
<td>big</td>
<td>easy</td>
<td>easy</td>
</tr>
</tbody>
</table>

Table 2: Qualitative search performance dependencies on the target-target similarity and target-nontarget similarity.
improved the performance for finding the first target compared to FLNN.

6.3. Using Top Down information

Using the method suggested in section 5.6, we extended the implementation of the Linear estimation based algorithm to use top-down information and demonstrate it on the toy cars search task: Three toy cars taken from various internet sources are used as model targets. Those images were converted from colored images to gray scale images and were scaled to the size of the other candidates. The models and algorithm performance are described in figure 9.

![Figure 9: Results of Linear estimation using top-down information for the toy cars search task. Left: the three model images used (taken from various internet sources). Right: Algorithm results in the (time, number of target found) space.](image)

6.4. Covariance vs. distance experiments

The linear estimation based search relies on a given covariance between every pair of labels of candidates. We use a covariance function that depends only on feature-space-distance, and we claim that for many search tasks this function is monotone descending. In this section we study the behavior of the covariance of labels vs. feature-space-distance of search tasks, and check whether the choice made in section 6.2 is justified.

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. The labels covariance \( \gamma_i, i = 1, \ldots, h \) in calculated separately for every interval \( I_i \):
\[
\gamma_i = \frac{\text{cov}_i(l_1, l_2) = E(l_1l_2) - E(l_1)E(l_2)}{n_{TT}\cdot 1\cdot 1 + n_{TN}\cdot 1\cdot 0 + n_{NN}\cdot 0\cdot 0 - \mu^2}
\]

where \(n_{TT}, n_{TN},\) and \(n_{NN}\) are the number of target-target, target-nontarget, and nontarget-nontarget pairs in interval \(I_i\) respectively. \(\mu = E(x)\) is number of total targets divided to number of total candidates.

The estimated covariances are plotted as a function of distance in figure 10. (Non-uniform intervals containing an equal number of target-target pairs result less erratic estimates.)

This procedure was performed on each of the search tasks described in sections 6.1 and 6.2. The results are displayed in figure 10. Except for the task of finding toy animals in the columbia100 image (in which there is almost no resemblance between targets), all other results show a monotone descending behavior. Note the different range of the the estimated covariances for each case. For instance, compare the results of the cars being targets to that of cups being targets: in both cases the covariance decreases monotonically while with distance, but the covariance for short distances is about 0.3 for the ‘cars’ and about 0.004 for the ‘cups’.

7. Discussion

Objects with the same identity tend to appear more similar when they are in the same scene than in the general case: The acquisition conditions are the same therefore illumination, background noise and camera calibration is common. They tend to be in the same spatial resolution (same size) and to appear in a similar pose. In addition, in natural scenes objects with same identity often appear few times. We suggest to benefit from these two facts by dynamically changing the priorities of objects to be attended to by considering previous recognition responses. This causes that: a) similar non-targets are ‘rejected together’. b) after one target is found, all similar targets follow.

In this work we put aside the common measures used for directing attention during visual search, and focus on the measure of inner-scene similarity. We show that in many cases this sort of information can help speed-up the search. For example - the parasols search task described in section 6.2 is a
Figure 10: Estimation of labels covariance vs. feature-space-distance. For each distance interval $I_i$ for which we estimated the covariance $\gamma_i$, the point $(d_i, \gamma_i)$, where $d_i$ is the center of $I_i$, is plotted.
good example for a case in which we may know that all our targets are similar and different from the non-targets in a certain feature/s, but we don’t know the targets value of this feature, so we do not have a pre-specified preference for a certain feature value. (in this case - we know all parasols have the same color, but we don’t know which color). We can’t use bottom-up approach, giving priority to salient regions, since the targets cover a significant area of the image, and therefore are not salient. The knowledge available in this case is only based on inner-scene relationships and therefore the search strategy suggested in this paper is appropriate.

We have found numeric expression for the difficulty of a search task, that enables to decide when the search can benefit from inner-scene similarity and how much. We have suggested a general framework for inner-scene-similarity based search, as well as a specific stochastic algorithm based on Linear estimation and guided by the assumption that labels correlation descends as candidates feature-space-distance grows. By primary experiments, we have showed that this is the case for common visual search tasks.

Our results shows there is a continuity between the two poles of ‘pop-out’ and ‘sequential’ searches. Two major factors effect the difficulty of a search task: similarity between target and non-targets and similarity between non-targets and non-targets. We have expressed the combination of those two factors by one measure, which we call $dy$-cover-size, and showed it is a bound on all search algorithms.
References


