Robust Real Time Pattern Matching using Bayesian Sequential Hypothesis Testing

Ofir Pele and Michael Werman

***Abstract*— This paper describes a method for robust real time pattern matching. We first introduce a family of image distance measures, the "*Image Hamming Distance Family*". Members of this family are robust to occlusion, small geometrical transforms, light changes and non-rigid deformations. We then present a novel Bayesian framework for sequential hypothesis testing on finite populations. Based on this framework, we design an optimal rejection/acceptance sampling algorithm. This algorithm quickly determines whether two images are similar with respect to a member of the *Image Hamming Distance Family*. We also present a fast framework that designs a near-optimal sampling algorithm. Extensive experimental results show that the sequential sampling algorithm performance is excellent. Implemented on a Pentium 4 3GHz processor, detection of a pattern with 2197 pixels, in 640x480 pixel frames, where in each frame the pattern rotated and was highly occluded, proceeds at only 0.022 seconds per frame.**

***Index Terms*— Pattern matching, template matching, pattern detection, image similarity measures, Hamming distance, real time, sequential hypothesis testing, composite hypothesis, image statistics, Bayesian statistics, finite populations**

I. Introduction

MANY applications in image processing and computer vision require finding a particular pattern in an image, *pattern matching*. To be useful in practice, pattern matching methods must be automatic, generic, fast and robust.

Pattern matching is typically performed by scanning the entire image, and evaluating a distance measure between the pattern and a local rectangular window. The method proposed in this paper is applicable to any pattern shape, even a non-contiguous one. We use the notion of "window" to cover all possible shapes.

First, we introduce a family of image distance measures called the "*Image Hamming Distance Family*". A distance measure in this family is the number of non-similar corresponding features between two images. Members of this family are robust to occlusion, small geometrical transforms, light changes and non-rigid deformations.

Second, we show how to quickly decide whether a window is similar to the pattern with respect to a member of the "Image Hamming Distance Family". The trivial, but time consuming solution is to compute the exact distance between the pattern and the window by going over all the corresponding features (the simplest feature is a pixel). We present an algorithm that samples corresponding features and accumulates the number of non-similar features. The speed of this algorithm is based on the fact that the distance between two non-similar images is usually very large whereas the distance between two similar images is usually very small (see Fig. 2). Therefore, for non-similar windows the sum will grow extremely fast and we will be able to quickly decide that they are non-similar. As the event of similarity in pattern matching is so rare (see Fig. 2), we can afford to pay the price of going over all the corresponding features in similar windows. Note that the algorithm does not attempt to estimate the distances for non-similar windows. The algorithm only decides that these windows, with a very high probability (for example, 99.9%), are non-similar. The reduction in running time is due to the fact that this unnecessary information is not computed.

The idea of sequential sampling [1] or sequential sampling a distance is not new [2]. The major contribution in our work is a novel efficient Bayesian framework for hypothesis testing on finite populations. Given allowable bounds on the probability of error (false negatives and false positives) the framework designs a sampling algorithm that has the minimum expected running time. This is done in an offline phase for each pattern size. An online phase uses the sampling algorithm to quickly find patterns. In order to reduce offline running time we

O. Pele and M. Werman are with the The Hebrew University of Jerusalem e-mail: {ofirpele,werman}@cs.huji.ac.il

Manuscript received ; revised

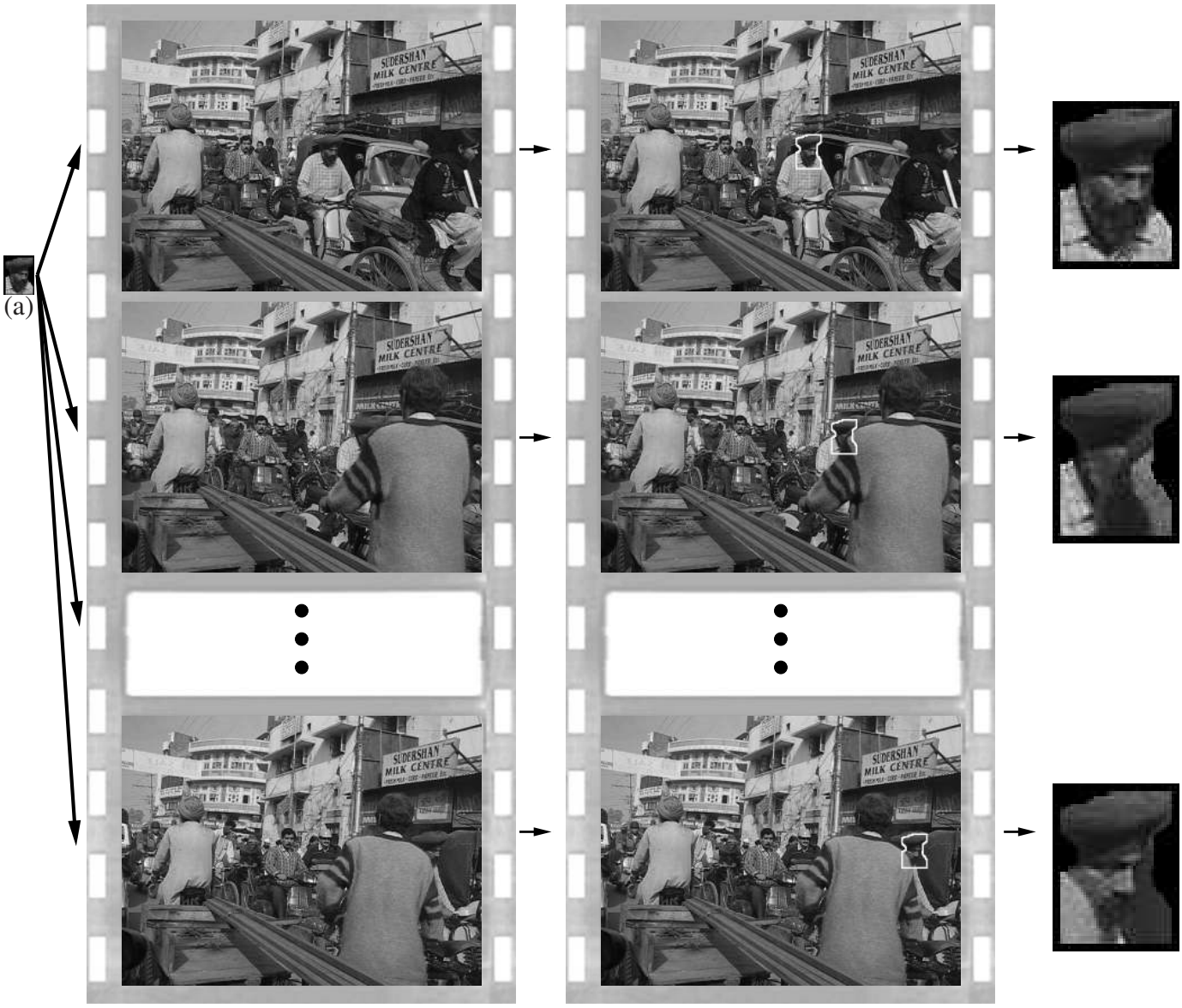


Fig. 1. Real time detection of a rotating and highly occluded pattern.
(a) A non-rectangular pattern of 2197 pixels. Pixels not belonging to the mask are in black. (b) Three 640x480 pixel frames out of fourteen in which the pattern was sought. (c) The result. Most similar masked windows are marked in white. (d) Zoom in of the occurrences of the pattern in the frames. Pixels not belonging to the mask are in black.
The SEQUENTIAL algorithm proceeds at only 0.022 seconds per frame. Offline running time - time spent on the parameterization of the SEQUENTIAL algorithm (with P-SPRT, see Section IV-D) was 0.067 seconds. Note that the distance is robust to out of plane rotations and occlusion. Using other distances such as CC, NCC, l_2, l_1 yielded poor results. In particular they all failed to find the pattern in the last frame. We emphasize that no motion consideration was taken into account in computation. The algorithm ran on all windows. Full size images are available at: `http://www.cs.huji.ac.il/~ofirpele/hs/all_images.zip`

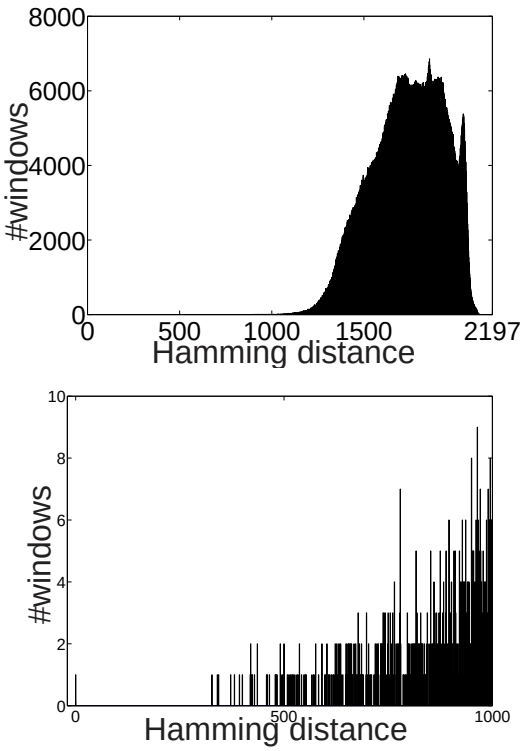


Fig. 2. The distance of the pattern to most windows in pattern matching is very high. A distance measure from the Image Hamming Distance Family was computed between the pattern and a sliding masked window in the video frames of Fig. 1. Above we see the resulting histogram. The left part of the histogram is zoomed in. We can see that most of the windows in the video were very far from the pattern. This is a typical histogram.

also present a fast framework that designs a near-optimal sampling algorithm. For comparison, we also present a framework that designs an optimal fixed size sampling algorithm. Theoretical and experimental results shows that sequential sampling needs significantly fewer samples than fixed size sampling.

Sampling is frequently used in computer vision, to reduce time complexity that is caused by the size of the image data. Our work (like work by Matas et al. [3], [4]) shows that designing an optimal or a near-optimal *sequential* sampling scheme (by contrast to the frequently used *fixed size* sampling scheme) is important and can improve speed and accuracy significantly.

A typical pattern matching task is shown in Fig. 1. A non-rectangular pattern of 2197 pixels was sought in a sequence of 14, 640x480 pixel frames. Using the sampling algorithm the pattern was found in 9 out of 11 frames in which it was present, with an average of only 19.70 pixels examined per window instead of 2197 needed for the exact distance computation. On a Pentium 4 3GHz processor, it proceeds at only 0.022 seconds per frame. Other distances such as cross correlation (CC), normalized cross correlation (NCC), l_1, l_2, yielded poor results even though they were computed exactly (the computation took much longer).

TABLE I
NOTATION TABLE

A, $\|A\|$	Set that contains spatial coordinates of features (for example, spatial coordinates of pixels). $\|A\|$ is the size of the set.
D	Random variable of the Hamming distance.
t	Image similarity threshold, *i.e.* if the Hamming distance of two images is smaller or equal to t, then the images are considered similar. Otherwise, the images are considered non-similar.
p	Pixel similarity threshold. Used in several members of the Image Hamming Distance Family.

This paper is organized as follows. Section II is an overview of previous work on fast pattern matching, Hamming distance in computer vision and sequential hypothesis testing. Section III introduces the Image Hamming Distance Family. Section IV describes the Bayesian framework. Section V discusses the issue of the prior. Section VI presents extensive experimental results. Finally, conclusions are drawn in Section VII. A notation table for the rest of the paper is given in Table I.

II. Previous Work

A. Fast Pattern Matching

The distances most widely used for fast pattern matching are cross correlation and normalized cross correlation. Both can be computed relatively quickly in the Fourier domain [5], [6].

The main drawback of correlation, which is based on the Euclidean distance, is that it is specific to Gaussian noise. The difference between images of the same object often results from occlusion, geometrical transforms, light changes and non-rigid deformations. None of these can be modeled well

with a Gaussian distribution. For a further discussion on Euclidean distance as a similarity measure see [7]–[10]. Note that although the Hamming distance is not specific to Gaussian noise as the l_2 norm, it is robust to Gaussian noise (see Fig. 3). Normalized cross correlation is invariant to additive and multiplicative gray level changes. However, natural light changes include different effects, such as shading, spectral reflectance, etc. In addition, when a correlation is computed in the transform domain, it can only be used with rectangular patterns and usually the images are padded so that their height and width are dyadic.

Lucas and Kanade [11] employed the spatial intensity gradients of images to find a good match using a Newton-Raphson type of iteration. The method is based on Euclidean distance and it assumes that the two images are already in approximate registration.

Local descriptors have been used recently for object recognition [12]–[17]. The matching is done by first extracting the descriptors and then matching them. Although fast, our approach is faster. In addition, there are cases where the local descriptors approach is not successful (see Fig. 7). If one knows that the object view does not change drastically, the invariance of the local descriptors can affect performance and robustness [17]. In this work we decided to concentrate on pixel values or simple relation of pixels as features. Combining the sequential sampling algorithm approach with the local descriptors approach is an interesting extension of this work.

Recently there have been advances in the field of fast object detection using a cascade of rejectors [18]–[21]. Viola and Jones [20] demonstrated the advantages of such an approach. They achieved real time frontal face detection using a boosted cascade of simple features. Avidan and Butman [21] showed that instead of looking at all the pixels in the image, one can choose several representative pixels for fast rejection of non-face images. In this work we do not deal with classification problems but rather with a pattern matching approach. Our approach does not include a learning phase. The learning phase makes classification techniques impractical when many different patterns are sought or when the sought pattern is given online, *e.g.* in the case of patch-based texture synthesis [22], pattern matching in motion estimation, etc.

Hel-Or and Hel-Or [23] used a rejection scheme for fast pattern matching with projection kernels. Their method is applicable to any norm distance, and was demonstrated on the Euclidean distance. They compute the Walsh-Hadamard basis projections in a certain order. For the method to work fast the first Walsh-Hadamard basis projections (according to the specific order) need to contain enough information to discriminate most images. Ben-Artzi et al. [24] proposed a faster projection scheme called "Gray-Code Kernels". Ben-Yehuda et al. [25] extended the Hel-Or pattern matching method to handle non-rectangular patterns by decomposition of the pattern into several dyadic components.

Cha [26] uses functions that are lower bounds to the sum of absolute differences, and are fast to compute. They are designed to eliminate non-similar images fast. The first function he suggests is the *h-distance*: $\sum_{k=0}^{r-1} |\sum_{l=0}^{k}(G_l(Im_1) - G_l(Im_2))|$, where $G_l(Im)$ is the number of pixels with gray level l in the intensity histogram of the image, Im, and r is the number of gray levels, usually 256. The time complexity is O(r). The second function he suggests is the absolute value of difference between sums of pixels: $|\sum Im_1(x,y) - \sum Im_2(x,y)|$. The method is restricted to the l_1 norm and assumes that these functions can reject most of the images fast.

One of the first rejection schemes was proposed by Barnea and Silverman [2]. They suggested the Sequential Similarity Detection Algorithms - SSDA. The method accumulates the sum of absolute differences of the intensity values in both images and applies a threshold criterion - if the accumulated sum exceeds a threshold, which can increase with the number of pixels, they stop and return *non-similar*. The order of the pixels is chosen randomly. After n iterations, the algorithm stops and returns *similar*. They suggested three heuristics for finding the thresholds for the l_1 norm. This method is very efficient but has one main drawback. None of the heuristics for choosing the thresholds guarantees a bound on the error rate. As a result the SSDA was said to be inaccurate [27]. Our work is a variation of the SSDA. We use a member of the Image Hamming Distance Family instead of the l_1 norm. We also design a sampling scheme with proven error bounds and optimal running time. As the SSDA uses the l_1 norm, in each figure where the l_1 norm yields poor results (see Figs. 1, 4, 5 and 6) the SSDA also yields poor results.

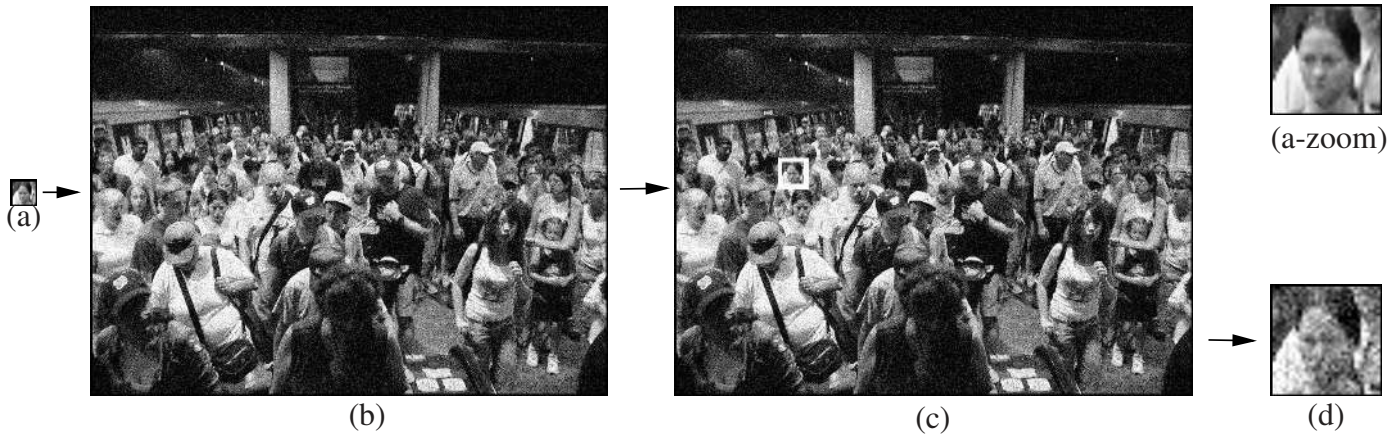


Fig. 3. Real time detection of a specific face in a noisy image of a crowd.
(a) A rectangular pattern of 1089 pixels. (b) A noisy version of the original 640x480 pixel image. The pattern that was taken from the original image was sought in this image. The noise is Gaussian with a mean of zero and a standard deviation of 25.5. (c) The result image. The single similar masked window is marked in white. (d) The occurrence of the pattern in the zoomed in image. The SEQUENTIAL algorithm proceeds at only 0.019 seconds per frame. Offline running time - time spent on the parameterization of the SEQUENTIAL algorithm (with P-SPRT, see Section IV-D) was 0.018 seconds. Note that although the Hamming distance is not specific to Gaussian noise as the l_2 norm, it is robust to Gaussian noise. The image is copyright by Ben Schumin and was downloaded from: `http://en.wikipedia.org/wiki/Image:July_4_crowd_at_Vienna_Metro_station.jpg`. Full size images are available at: `http://www.cs.huji.ac.il/~ofirpele/hs/all_images.zip`

Mascarenhas et al. [28], [29] used Wald's Sequential Probability Ratio Test (SPRT) [1] as the sampling scheme. Two models were suggested for the random variable of the distance of the sample k. The first converts the images to binary and then $P(k)$ is binomially distributed. The second assumes that the images are Gaussian distributed; hence $P(k)$ is also Gaussian distributed. The likelihood ratio is defined as: $\lambda(k) = \frac{P(k|\text{images are non-similar})}{P(k|\text{images are similar})}$. The SPRT samples both images as long as: $\mathcal{A} < \lambda(k) < \mathcal{B}$. When $\lambda(k) \leq \mathcal{A}$ the SPRT stops and returns *similar*. When $\lambda(k) \geq \mathcal{B}$ the SPRT stops and returns *non-similar*. Let the bounds on the allowable error rates be $P(\text{false positive}) = \beta$, $P(\text{false negative}) = \alpha$. Wald's [1] approximation for $\mathcal{A}$ and $\mathcal{B}$ is $\mathcal{A} = \frac{\beta}{1-\alpha}$, $\mathcal{B} = \frac{1-\beta}{\alpha}$.

There are several problems with their method. Converting images to binary results in a loss of information. In addition, gray levels are far from being Gaussian distributed [30]–[33]. Our method does not assume any prior on the images. Mascarenhas et al. assume that all similar images have exactly the same pairwise small distance, whereas any two non-similar images have exactly the same large distance, an assumption that is faulty. Our framework gets a prior on the distribution of image distances as input. The classical SPRT can go on infinitely. There are ways to truncate it [34], but they are not optimal. By contrast, we designed an optimal rejection/acceptance sampling scheme with a restricted number of samples.

B. Hamming Distance in computer vision

Hamming Distance in computer vision [35]–[41] has usually been applied to a binary image, ordinarily a binary transform of a gray level image. Ionescu and Ralescu's crisp version of the "fuzzy Hamming distance" [39] is an exception, where a threshold function is applied to decide whether two colors are similar.

A comprehensive review of local binary features of images and their usage for 2D Object detection and recognition can be found in Amit's book [35]. Amit suggests using the Hough transform [42] to find arrangements of the local binary features. In Appendix II we show how the Hough transform can be used to compute the Hamming distance of a pattern with all windows of an image. We also show that the expected time complexity for each window is $O(|A| - E[D])$, where $|A|$ is the number of pixels in the pattern's set of pixels and D is the random variable of the Hamming distance between a random window and a random pattern. For the pattern matching in Fig. 1, $E[D] = 1736.64$, $|A| = 2197$. Thus, the average work for each window using the Hough transform is 460.36, much higher than the 19.70 needed using our approach, but much less than comparing all the corresponding pixels.

Bookstein et al. [40] proposed the "Generalized Hamming Distance" for object recognition on binary images. The distance extends the Hamming concept to give partial credit for near misses. They suggest a dynamic programming algorithm to compute it. The time complexity is $O(|A| + \sum Im_1 \sum Im_2)$, where $|A|$ is the number of pixels and $\sum Im$ is the number of ones in the binary image, Im. Our method is sub-linear. Another disadvantage is that their method only copes with near misses in the horizontal direction. We suggest using the *Local Deformations* method (see Section III) to handle near misses in all directions.

C. Sequential Hypothesis Testing

Sequential tests are hypothesis tests in which the number of samples is not fixed but rather is a random variable. This area has been an active field of research in statistics since its initial development by Wald [1]. A mathematical review can be found in Siegmund [34].

There have been many applications of sampling in computer vision to reduce time complexity that is caused by the size of the image data. However, most have been applied with a sample of fixed size. Exceptions are [2]–[4], [28], [29], [43]–[45]. The sampling schemes that were used are Wald's SPRT [1] for simple hypotheses, or a truncated version of the SPRT (which is not optimal) or estimation of the thresholds. The Matas and Chum method [3] for reducing the running time of RANSAC [46] is an excellent example of the importance of optimal design of sampling algorithms.

There are several differences between the above methods and the one presented here. The first is that in the pattern matching problem, the hypotheses are composite and not simple. Let D be the random variable of the Hamming distance between a random window and a random pattern. Instead of testing the simple hypothesis $D = d_1$ against the simple hypothesis $D = d_2$, we need to test the composite hypothesis $D \leq t$ against the composite hypothesis $D > t$. This problem is solved by using a prior on the distribution of the Hamming distance and developing a framework that designs an optimal sampling algorithm with respect to the prior. The second difference is that the efficiency of the design of the optimal sampling algorithm is also taken into consideration. In addition, we present a fast algorithm that designs a near-optimal sampling scheme. Finally, as a by-product, our approach returns the expected running time and the expected error rate.

III. Image Hamming Distance Family

A distance measure from the Image Hamming Distance Family is the number of non-similar corresponding features between two images, where the definition of a feature and similarity vary between members of the family. Below is a formal definition and several examples.

A. Formal Definition

$sim(Im_1, Im_2, (x,y)^m) \rightarrow \{0,1\}$ is the similarity function, 1 is for non-similar, 0 is for similar, where Im_1, Im_2 are images and $(x,y)^m$ are the spatial coordinates of a feature. In all our examples m is 1 or 2. If $m = 1$ we are testing for similarity between pixels. If $m = 2$ we are testing for similarity between pairs of pixels (see "*Monotonic Relations*" in Section. III-B for an example). We usually omit m for simplicity.

$HammingDistance_A(Im_1, Im_2) = \sum_{(x,y)^m \in A} sim(Im_1, Im_2, (x,y)^m)$ is the Hamming distance between the set of spatial coordinates A applied to the images Im_1, Im_2. Note that the spatial coordinates in A do not need to form a rectangular window in the image. In fact they do not need to form a connected region.

B. Examples

In all following examples the δ function returns 1 for true and 0 for false.

"*Thresholded Absolute Difference*"
$sim(Im_1, Im_2, (x,y)) =$
$\delta(|(Im_1(x,y)) - (Im_2(x,y))| > p)$

The distance is similar to Gharavi and Mills's PDC distance [47].

"*Thresholded l_2 norm in L*a*b color space*"
$sim(Im_1, Im_2, (x,y)) =$
$\delta(||\text{L*a*b*}(Im_1(x,y)) - \text{L*a*b*}(Im_2(x,y))||_2 > p)$

The L*a*b* color space was shown to be approximately perceptually uniform [48]. This means that colors which appear similar to an observer are located close to each other in the L*a*b* coordinate system. *i.e.* by thresholding the Euclidean distance between the two $\langle L*, a*, b* \rangle$ vectors, the function

tests whether two color pixels are perceptually similar. Note that if the color is more important, we can multiply the $L*$ channel with a coefficient smaller than one.

"*Monotonic Relations*"

The features used in this distance are pairs of pixels. The pair $[Im_1(x_1, y_1), Im_1(x_2, y_2)]$ is considered similar to $[Im_2(x_1, y_1), Im_2(x_2, y_2)]$ if the same relation holds between them. For example, assuming WLOG that $Im_1(x_1, y_1) > Im_1(x_2, y_2)$ for all pairs of coordinates $[(x_1, y_1), (x_2, y_2)]$ in A, the similarity function can be:

$sim(Im_1, Im_2, [(x_1, y_1), (x_2, y_2)]) = \delta(Im_2(x_1, y_1) \leq Im_2(x_2, y_2))$

This distance is invariant to noises that preserve monotonic relations. Thus it is robust to light changes (see Figs. 4 and 5). The distance is equivalent to the Hamming distance on the Zabih and Woodfill *census* transform [36]. We suggest that for a specific pattern, a reasonable choice for $A = \{[(x_1, y_1), (x_2, y_2)]\}$ are pairs of indices that correspond to edges; *i.e.* points that are spatially proximal with large intensity difference. Such pairs are discriminative because of image smoothness.

"*Local Deformations*"

Local Deformations is an extension to distance measures of the Image Hamming Distance Family which makes them invariant to local deformations, *e.g.* non-rigid deformations (see Fig. 6). Let $sim(Im_1, Im_2, (x, y)^m)$ be the similarity function of the original Hamming distance measure. Let $\varepsilon = (\varepsilon_x, \varepsilon_y)$ be a shift. Let $(Im)_\varepsilon(x, y) = Im(x+\varepsilon_x, y+\varepsilon_y)$. We denote by Γ the set of allowable shifts. The *Local Deformations* variant similarity function of this Hamming distance measure is:

$\widehat{sim}(Im_1, Im_2, (x, y)^m) = \min_{\varepsilon \in \Gamma} sim(Im_1, (Im_2)_\varepsilon, (x, y)^m)$

Brunelli and Poggio [49] used a similar technique to make CC more robust.

C. Advantages

Members of the Image Hamming Distance Family can be invariant to light changes, small deformations, etc. Invariance is achieved by "plugging in" the appropriate similarity function.

Members of the Image Hamming Distance Family have an inherent robustness to outlier noise, for example, out of plane rotation, shading, spectral reflectance, occlusion, etc. Using the Hamming distance, outliers up to the image similarity threshold t are disregarded. Norms such as the Euclidean add irrelevant information; namely, the difference between the intensity values of such pixels and the image.

The Euclidean norm is most suited to deal with Gaussian noise. The difference between images of the same object often results from occlusion, geometrical transforms, light changes and non-rigid deformations. None of these can be modeled well with a Gaussian distribution.

Although it might seem that members of the Image Hamming Distance Family are not robust because the similarity function of a feature sim is a threshold function, it is in fact robust because it is a sum of such functions.

Finally, the simplicity of the Image Hamming Distance Family allows us to develop a tractable Bayesian framework that is used to design an optimal rejection/acceptance sampling algorithm. After we design the sampling algorithm offline, it can quickly determine whether two images are similar.

IV. Sequential Framework

We first present the SEQUENTIAL algorithm that assesses similarity by a sequential test. Then we evaluate its performance and show how to find the optimal parameters for the SEQUENTIAL algorithm. Finally we illustrate how to quickly find near-optimal parameters for the SEQUENTIAL algorithm.

A. The SEQUENTIAL algorithm

The SEQUENTIAL algorithm, Alg. 1, uses a decision matrix M. $M[k, n]$ is the decision after sampling k non-similar corresponding features out of a total of n sampled corresponding features. The decision can be *NS*=return *non-similar*, *S*=return *similar* or *C*=continue sampling. The last column $|A|$ cannot be *C* as the test has to end there, see the diagram in Fig. 8. We random sample uniformly as we do not want to make any assumptions about the noise. Note that as we sample without replacement, the algorithm always returns *similar* or *non-similar*

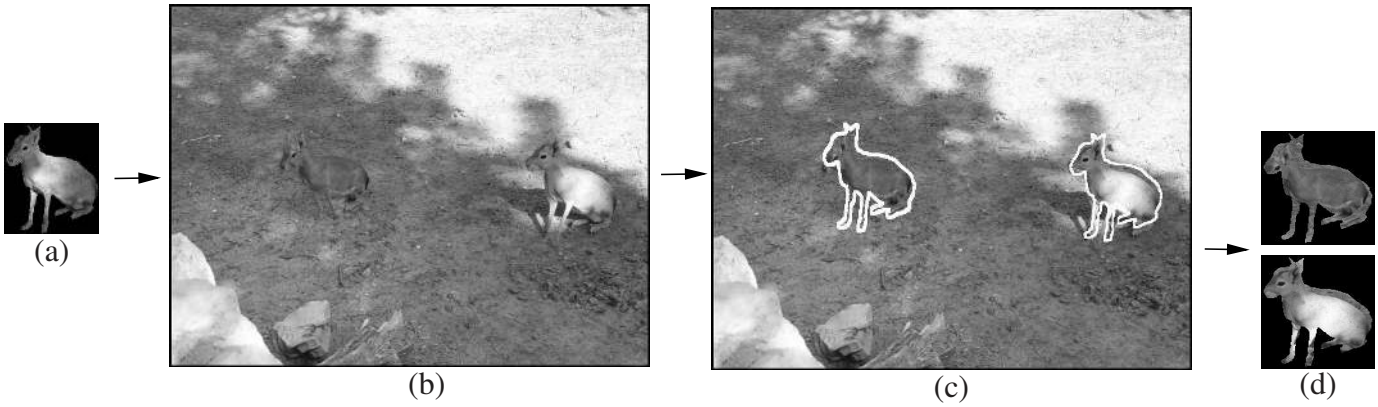


Fig. 4. *Monotonic Relations* Hamming distance is robust to light changes and small out of plane and in plane rotations.
(a) A non-rectangular pattern of 7569 pixels (631 edge pixel pairs). Pixels not belonging to the mask are in black. (b) A 640x480 pixel image in which the pattern was sought. (c) The result image. All similar masked windows are marked in white. (d) The two found occurrences of the pattern in the image. Pixels not belonging to the mask are in black. The SEQUENTIAL algorithm proceeds at only 0.021 seconds. Offline running time - time spent on the parameterization of the SEQUENTIAL algorithm (with P-SPRT, see Section IV-D) and finding the edge pixels was 0.009 seconds. Note the substantial differences in shading between the pattern and its two occurrences in the image. Also note the out of plane (mostly the head) and in plane rotations of the maras (the animals in the picture). Using other distances such as CC, NCC, l_2, l_1 yielded poor results. In particular the closest window using CC, l_2, l_1 was far from the maras. Using NCC the closest window was near the right mara but it found many false positives before finding the left mara. The pairs that were used are pairs of pixels belonging to edges, *i.e.* pixels that have a neighbor pixel, where the absolute intensity value difference is greater than 80. Two pixels, (x_2, y_2), (x_1, y_1) are considered neighbors if their l_∞ distance: $\max(|x_1 - x_2|, |y_1 - y_2|)$ is smaller or equal to 2. There are 631 such pairs in the pattern. Similar windows are windows where at least 25% of their pairs exhibit the same relation as in the pattern. Full size images are available at: `http://www.cs.huji.ac.il/~ofirpele/hs/all_images.zip`

Fig. 5. *Monotonic Relations* Hamming distance is robust to light changes and occlusion.
(a) A non-rectangular pattern of 2270 pixels (9409 edge pixel pairs). Pixels not belonging to the mask are in black. (b) A 640x480 pixel image in which the pattern was sought. (c) The result image. The single similar masked window is marked in white. (d) The occurrences of the pattern in the image zoomed in. Pixels not belonging to the mask are in black. The SEQUENTIAL algorithm proceeds at only 0.037 seconds. Offline running time - time spent on the parameterization of the SEQUENTIAL algorithm (with P-SPRT, see Section IV-D) and finding the edge pixels was 1.219 seconds. Note the considerable differences in the light between the pattern and the occurrences of the pattern in the image, especially the specular reflection in the pattern. Also note the difference in the spotting of the frogs and the difference in the pose of the legs (the top right leg is not visible in the image). Using other distances such as CC, NCC, l_2, l_1 yielded poor results. In particular the closest window using CC, NCC, l_2, l_1 was far from the frog. The pairs that were used are pairs of pixels belonging to edges, *i.e.* pixels that have a neighbor pixel, where the absolute intensity value difference is greater than 80. Two pixels, (x_2, y_2), (x_1, y_1) are considered neighbors if their l_∞ distance: $\max(|x_1 - x_2|, |y_1 - y_2|)$ is smaller or equal to 5. There are 9409 such pairs in the pattern. Similar windows are windows where at least 25% of their pairs exhibit the same relation as in the pattern. Full size images are available at: `http://www.cs.huji.ac.il/~ofirpele/hs/all_images.zip`

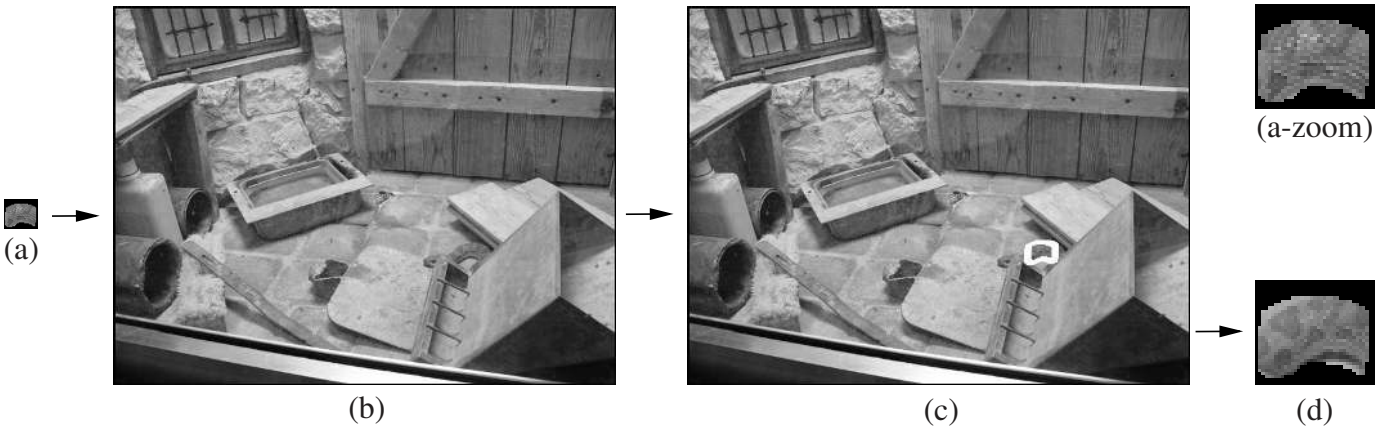


Fig. 6. *Local Deformations* is robust to non-rigid deformations.
(a) A non-rectangular pattern (snake skin) of 714 pixels. Pixels not belonging to the mask are in black. (b) A 640x480 pixel image in which the pattern was sought. (c) The result image. All similar masked (adjacent) windows are marked in white. (d) Most similar occurrence of the pattern in the zoomed-in image. Pixels not belonging to the mask are in black. The SEQUENTIAL algorithm proceeds at only 0.064 seconds. Offline running time - time spent on the parameterization of the SEQUENTIAL algorithm (with P-SPRT, see Section IV-D) was 0.007 seconds. Using other distances such as CC, NCC, l_2, l_1 yielded poor results. In particular the closest window using CC, NCC, l_2, l_1 was far from the snake skin. SIFT descriptor matching [13], also yielded poor results (see Fig. 7). The distance that was used is the *Local Deformations* variant of the *Thresholded Absolute Difference* distance with a threshold of 20. The group of shifts is $\Gamma = \{\pm 1, \pm 1\}$, *i.e.* 8-neighbors. Similar windows are windows where at least 5% of their pixels (or neighbors) have an l_1 distance smaller or equal to 20. Full size images are available at: `http://www.cs.huji.ac.il/~ofirpele/hs/all_images.zip`

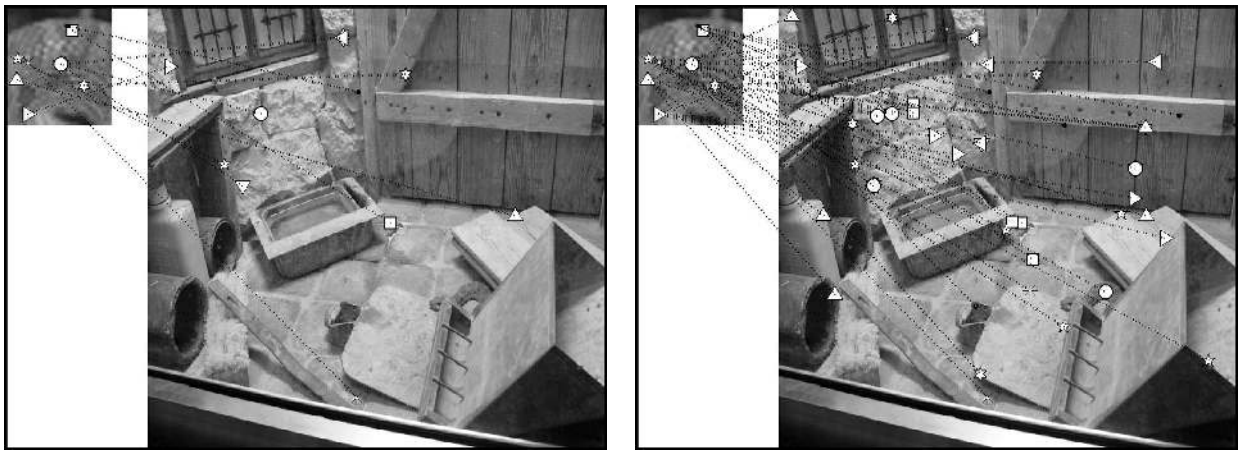


Fig. 7. SIFT descriptor matching [13] on the pattern matching in Fig. 6. The pattern in the left part of each figure is zoomed. (SIFT-1) The correspondences between the eleven SIFT descriptors in the pattern and the most similar SIFT descriptors in the image. Note that all correspondences are false. (SIFT-5) The correspondences between the eleven SIFT descriptors in the pattern and the five most similar SIFT descriptors in the image (each one to five correspondence groups has a different symbol). Note that only one correspondence is true. It is the fifth most similar correspondence of the descriptor and is marked with a circle.

after at most $|A|$ samples. Bear in mind that it is possible to add more kinds of decisions[1].

The framework computes the optimal decision matrix offline. Then, the algorithm can quickly decide whether a pattern and a window are similar, *i.e.* if their Hamming distance is smaller or equal to the image similarity threshold, t. Note that the optimal decision matrix does not have to be computed for each new pattern. It should be computed once for a given prior on the distribution of the distances, desired error bounds and the size of patterns. For example, if the task is finding 30×30 patterns, then it is enough to compute the decision matrix once.

[1] *e.g.* the computation of the exact distance that reduces the running time overhead of the checks on the decision matrix entries (the ifs in the algorithm). However, in practice this did not improve results.

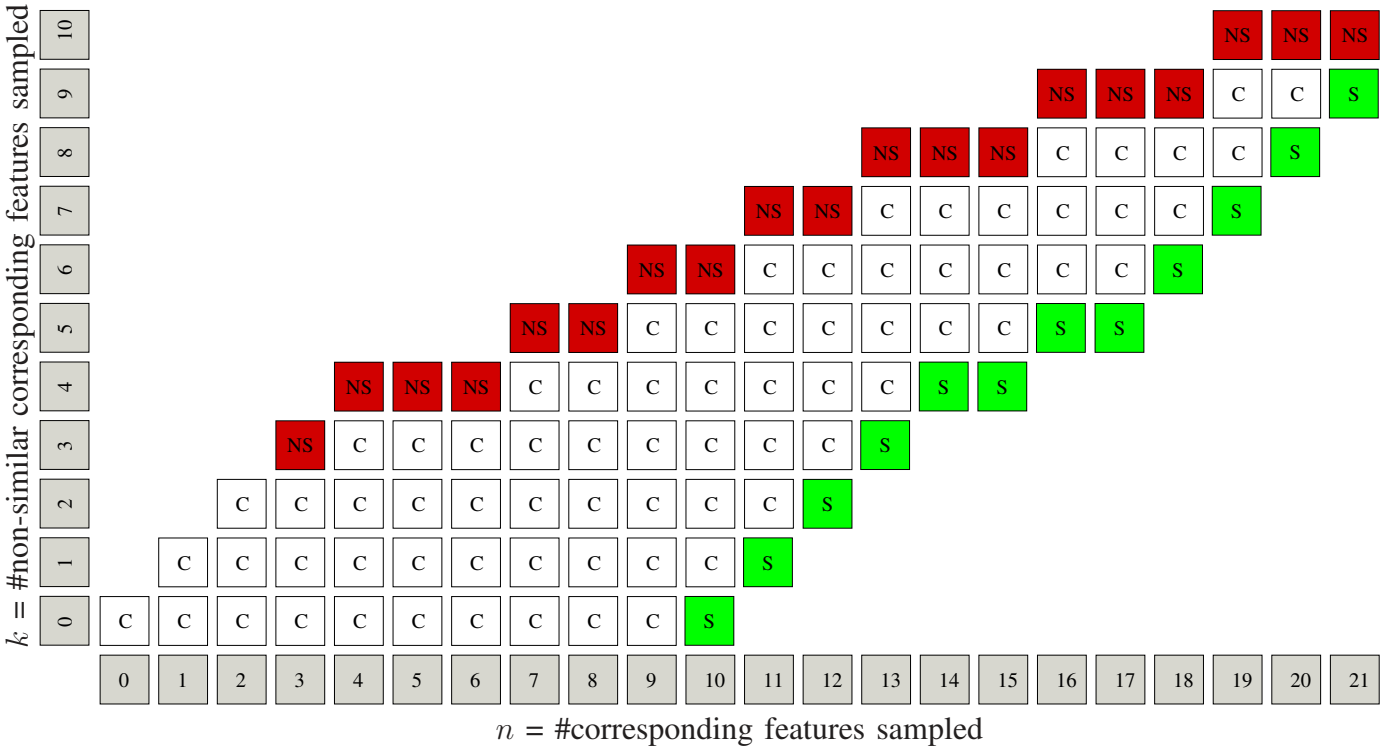


Fig. 8. Graphical representation of the decision matrix M which is used in the SEQUENTIAL algorithm (Alg. 1). In each step the algorithm samples corresponding features and goes right if they are similar or right and up if they are non-similar. If the algorithm touches a red *NS* point, it returns *non-similar*, with the risk of a false negative error. If the algorithm touches a green *S* point, it returns *similar*, with the risk of a false positive error. In this example, the size of the pattern, $|A|$ is 21 and the threshold for image similarity, t is 9. Note that the SEQUENTIAL algorithm parameterized with this decision matrix requires at least three non-similar corresponding features to return *non-similar* and at least ten similar corresponding features to return *similar*.

Algorithm 1 SEQUENTIAL$_M$(pattern, window, A)

$k \Leftarrow 0$
for $n = 0$ to $|A|$ **do**
  if $M[k, n] = NS$ **then**
    return *non-similar*
  if $M[k, n] = S$ **then**
    return *similar*
  random sample uniformly and without replacement $(x, y)^m$ from A
  \\ add 1 if features are non-similar
  $k \Leftarrow k + sim(\text{pattern}, \text{window}, (x, y)^m)$

B. Evaluating performance of a fixed decision matrix

In order to find the optimal decision matrix for the SEQUENTIAL algorithm, we first evaluate the performance of the algorithm for a fixed decision matrix. The performance of the algorithm is defined by its expected number of samples and its error probabilities:

E_M(#samples) = Expected number of samples (proportional to running time).
P_M(false negative) = Probability of returning *non-similar* on similar windows.
P_M(false positive) = Probability of returning *similar* on non-similar windows.

We denote by $e_{k,n}$ the event of sampling k non-similar corresponding features out of a total of n sampled corresponding features, in any specific order (for example, where the non-similar corresponding features are sampled first). Note that all orders of sampling have the same probability. As we sample without replacement we get:

$$P(e_{k,n}|D=d) = \begin{cases} \left(\prod_{i=0}^{k-1} \frac{d-i}{|A|-i}\right)\left(\prod_{i=0}^{n-k-1} \frac{|A|-d-i}{|A|-k-i}\right) & \text{if } (d \geq k)\& \\ & (|A| - d \geq n-k) \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

The naive computation of $P(e_{k,n}|D=d)$ for each k, n and d runs in $O(|A|^4)$. In order to reduce time complexity to $O(|A|^3)$, we use a dynamic

programming algorithm to compute the intermediate sums, $\Omega_S[k,n]$ and $\Omega_{NS}[k,n]$ (see Eq. 2) for each k and n where $P(D=d)$ is the prior on the distribution of the Hamming distance (see Section V).

$$\begin{aligned}\Omega_S[k,n] &= \sum_{d=0}^{t} P(e_{k,n}|D=d)P(D=d) \\ \Omega_{NS}[k,n] &= \sum_{d=t+1}^{|A|} P(e_{k,n}|D=d)P(D=d)\end{aligned} \tag{2}$$

In each step the SEQUENTIAL algorithm samples spatial coordinates of a feature $(x,y)^m$ and adds $sim(\text{pattern}, \text{window}, (x,y)^m)$ to the sample dissimilarity sum k. Define a specific run of the algorithm as a sequence of random variables: $s_1, s_2, \ldots$ where $s_n \in \{0,1\}$ is the result of $sim(\text{pattern}, \text{window}, (x,y)^m)$ in iteration number n. Let $\Psi_M[k,n]$ be the number of different sequences of $s_1, s_2, \ldots, s_n$ with k ones and $n-k$ zeros which will not cause the SEQUENTIAL algorithm that uses the decision matrix M to stop at an iteration smaller than n. Graphically (see Fig. 8) $\Psi_M[k,n]$ is the number of paths from the point $(0,0)$ to the point (k,n) that do not touch a stopping point (*S*,*NS*). Alg. 2 computes Ψ_M with time complexity of O($|A|^2$).

Algorithm 2 computeΨ_M

```
Ψ[0...|A|, 0...|A|] ⇐ 0
k ⇐ 0   n ⇐ 0
while M[k,n] = C do
   Ψ[k,n] ⇐ 1
   n ⇐ n + 1
Ψ[k,n] ⇐ 1
for n = 1 to |A| do
   for k = n to 1 do
      if M[k,n-1] = C then
         Ψ[k,n] ⇐ Ψ[k,n] + Ψ[k,n-1]
      if M[k-1,n-1] = C then
         Ψ[k,n] ⇐ Ψ[k,n] + Ψ[k-1,n-1]
return Ψ
```

Now we can compute (see full derivation in Appendix III) the error probabilities and expected number of samples explicitly using a prior on the distribution of the Hamming distance , $P(D=d)$ (see Section V):

$$P_M(\text{false negative}) = \frac{\sum_{\substack{(k,n):\\ M(k,n)=NS}} \Psi[k,n]\Omega_S[k,n]}{P(D \le t)} \tag{3}$$

$$P_M(\text{false positive}) = \frac{\sum_{\substack{(k,n):\\ M(k,n)=S}} \Psi[k,n]\Omega_{NS}[k,n]}{P(D > t)} \tag{4}$$

$$E_M(\#\text{samples}) = \tag{5}$$

$$\sum_{\substack{(k,n):\\ M(k,n)\in\{S,NS\}}} \Psi[k,n](\Omega_S[k,n] + \Omega_{NS}[k,n])n \tag{6}$$

C. Finding the optimal decision matrix

Our goal is to find the decision matrix, M that minimizes expected number of samples given allowable bounds on the error probabilities, α, β:

$$\begin{aligned}&\arg\min_M E_M(\#\text{samples}) \\ &\quad s.t: \\ &\quad P_M(\text{false negative}) \le \alpha \\ &\quad P_M(\text{false positive}) \le \beta\end{aligned} \tag{7}$$

Instead of solving Eq. 7 directly we assign two new weights: w_0 for a false negative error event and w_1 for a false positive error event, *i.e.* we now look for the decision matrix, M that solves Eq. 8:

$$\begin{aligned}&\arg\min_M \text{loss}(M, w_0, w_1)\ s.t: \\ &\quad loss(M, w_0, w_1) = E_M(\#\text{samples}) + \\ &\quad P_M(\text{false positive})P(D > t)w_1 + \\ &\quad P_M(\text{false negative})P(D \le t)w_0\end{aligned} \tag{8}$$

Following the solution of Eq. 8 we show how to use it to solve Eq. 7. We solve Eq. 8 using the backward induction technique [50]. The backward induction algorithm, Alg. 3 is based on the principle that the best decision in each step is the one with the smallest expected addition to the loss function. In Appendix IV we show how to explicitly compute the expected additive loss of each decision in each step.

If we find error weights, w_0, w_1 such that the decision matrix, M that solves Eq. 8 has error probabilities, $P_M(\text{false negative}) = \alpha$ and

Algorithm 3 backward(w_0, w_1)

for $k = 0$ to $|A|$ **do**
  $M[k, |A|] \Leftarrow \arg\min_{\text{decision} \in \{NS,S\}}$
  $E[\text{addLoss}(\text{decision})|k, |A|]$
for $n = |A| - 1$ to 0 **do**
  for $k = 0$ to n **do**
    $M[k, n] \Leftarrow \arg\min_{\text{decision} \in \{NS,S,C\}}$
    $E[\text{addLoss}(\text{decision})|k, n]$
return M

$P_M(\text{false positive}) = \beta$, then we have also found the solution to the original minimization problem Eq. 7. See Appendix V, Theorem 1 for the proof.

In order to find the error weights, w_0, w_1 which yield a solution with errors as close as possible to the requested errors (α for false negative and β for false positive) we perform a search (Alg. 4). The search can be done on the 2D rectangle $w_0 \in [0, \frac{|A|}{\alpha P(D \leq t)}]$, $w_1 \in [0, \frac{|A|}{\beta P(D > t)}]$ as it is guaranteed that there is a solution in this rectangle with small enough errors (see Appendix V, Theorem 2). Note that increasing the error weights w_0 and w_1 can only increase the expected number of samples; thus there is no need to search beyond this rectangle.

Alg. 4 returns a decision matrix with minimum expected number of samples compared to all other decision matrices with fewer or equal error rates (see Appendix V, Theorem 1). However, as the search is on two parameters, the search for the requested errors can fail. In practice, the search always returns errors which are very close to the requested errors. In addition, if we restrict one of the errors to be zero, the search is on one parameter, hence a binary search returns a solution with errors as close as possible to the requested errors. If Alg. 4 fails to return a decision matrix with errors close to the requested errors, an exhaustive search of the error weights, w_0, w_1 with high resolution can be performed.

D. Finding a near-optimal decision matrix using P-SPRT

Above, we showed how to find the optimal decision matrix. The search is done offline for each combination of desired error bound, size of pattern and prior and not for each sought pattern. However, this process is time consuming. In this section we describe an algorithm that quickly finds a near-optimal decision matrix.

Algorithm 4 searchOpt(α, β)

$\min_{w_0} \Leftarrow 0 \quad \max_{w_0} \Leftarrow \frac{|A|}{\alpha P(D \leq t)}$
$\min_{w_1} \Leftarrow 0 \quad \max_{w_1} \Leftarrow \frac{|A|}{\beta P(D > t)}$
repeat
  $\text{mid}_{w_0} \Leftarrow \frac{\min_{w_0} + \max_{w_0}}{2}$
  $\text{mid}_{w_1} \Leftarrow \frac{\min_{w_1} + \max_{w_1}}{2}$
  $M \Leftarrow \text{backward}(\text{mid}_{w_0}, \text{mid}_{w_1})$
  computeΨ_M
  $P_M(\text{false negative}) \Leftarrow \frac{1}{P(D \leq t)} \sum_{\substack{(k,n): \\ M(k,n)=NS}} \Psi_M[k, n]\Omega_S[k, n]$
  $P_M(\text{false positive}) \Leftarrow \frac{1}{P(D > t)} \sum_{\substack{(k,n): \\ M(k,n)=S}} \Psi_M[k, n]\Omega_{NS}[k, n]$
  if $P_M(\text{false negative}) > \alpha$ **then**
    $\min_{w_0} \Leftarrow \text{mid}_{w_0}$
  else
    $\max_{w_0} \Leftarrow \text{mid}_{w_0}$
  if $P_M(\text{false positive}) > \beta$ **then**
    $\min_{w_1} \Leftarrow \text{mid}_{w_1}$
  else
    $\max_{w_1} \Leftarrow \text{mid}_{w_1}$
until $|P_M(\text{false negative}) - \alpha| + |P_M(\text{false positive}) - \beta| < \varepsilon$
return M

Our goal is again to find the decision matrix that minimizes the expected running time, given bounds on the error probabilities (see Eq. 7). We present a near-optimal solution based on Wald's Sequential Probability Ratio Test (SPRT) [1]. We call this test the "Prior based Sequential Probability Ratio Test", P-SPRT.

The classical SPRT [1] is a test between two simple hypotheses, *i.e.* hypotheses that specify the population distribution completely. For example, let D be the random variable of the Hamming distance between a random window and a random pattern. A test of simple hypotheses is $D = d_1$ against $D = d_2$. However, we need to test the composite hypothesis $D \leq t$ against the composite hypothesis $D > t$. This problem is solved by using a prior on the distribution of the Hamming distance, D.

We now define the likelihood ratio. We denote by $e_{k,n}$ the event of sampling k non-similar corresponding features out of a total of n sampled corresponding features, in any specific order (for example,

where the non-similar corresponding features are sampled first). Note that all orders of sampling have the same probability. The likelihood ratio, $\lambda(e_{k,n})$ is (see full derivation in Appendix VI):

$$\lambda(e_{k,n}) = \left(\frac{P(D \leq t)}{P(D > t)}\right)\left(\frac{\sum_{d=t+1}^{|A|} P(e_{k,n}, D = d)}{\sum_{d=0}^{t} P(e_{k,n}, D = d)}\right) \quad (9)$$

The P-SPRT samples both images as long as: $\mathcal{A} < \lambda(e_{k,n}) < \mathcal{B}$. When $\lambda(e_{k,n}) \leq \mathcal{A}$ the P-SPRT stops and returns *similar*. When $\lambda(e_{k,n}) \geq \mathcal{B}$ the P-SPRT stops and returns *non-similar*. Let the bounds on the allowable error rates be $P(\text{false positive}) = \beta$, $P(\text{false negative}) = \alpha$. Wald's [1] approximation for $\mathcal{A}$ and $\mathcal{B}$ is $\mathcal{A} = \frac{\beta}{1-\alpha}$ and $\mathcal{B} = \frac{1-\beta}{\alpha}$.

The near-optimal character of the SPRT was first proved by Wald and Wolfowitz [51]. For an accessible proof see Lehmann [52]. The proof is for simple hypotheses. However, replacing the likelihood ratio in the Lehmann proof with the prior based likelihood ratio (see Eq. 9) shows that the P-SPRT is a near-optimal solution to Eq. 7.

The SPRT and P-SPRT are near-optimal and not optimal, because of the "overshoot" effect, *i.e.* because the sampling is of discrete quantities, and finding a P-SPRT with the desired error rates may not be possible. In our experiments Wald's approximations gave slightly lower error rates and a slightly larger expected sample size. An improvement can be made by searching $\mathcal{A}$ and $\mathcal{B}$ for an error closer to the desired error bound. This can be done with $O(|A|^2)$ time complexity and $O(|A|)$ memory complexity for each step of the search. However, we have no bound on the number of steps that needs to be made in the search. In practice, Wald approximations give good results.

The search for the optimal decision matrix is equivalent to a search for two monotonic increasing lines. First is the line of acceptance (see Fig. 8 green *S* line); *i.e.* if the SEQUENTIAL algorithm touches this line it returns *similar*. Second is the line of rejection (see Fig. 8 red *NS* line); *i.e.* if the SEQUENTIAL algorithm touches this line it returns *non-similar*. Note that unlike the optimal solution, the P-SPRT solution cannot contain more than two kinds of complementary decisions (in our case - returning *similar* or returning *non-similar*).

We now describe an algorithm (Alg. 5) that computes the line of rejection in $O(|A|^2)$ time complexity and $O(|A|)$ memory complexity. The computation of the line of acceptance is similar. For each number of samples, n, we test whether the height of the point of rejection can stay the same as it was for the last stage, or whether it should increase by one. For this purpose we need to compare the likelihood ratio with the threshold $\mathcal{B}$. In order to compute the likelihood ratio fast, we keep a cache of the probability of being in the next rejection point and that the true distance is equal to d. The cache is stored in the array $P(e_{k,n}, D = d)$ for each distance d. Thus its size is $|A| + 1$. In Appendix VI we describe the explicit derivation of the cache initialization and update rules. For numerical stability, the cache in Alg. 5 can be normalized. In our implementation we store $P(D = d|e_{k,n})$ instead of $P(D = d, e_{k,n})$.

Algorithm 5 computeRejectionLine($|A|, \alpha, \beta, P[D]$)

$\mathcal{B} \Leftarrow \frac{1-\beta}{\alpha}$
\\ Never reject after 0 samples
rejectionLine[0] $\Leftarrow 1$
\\ Try (1,1) as first rejection point
$k \Leftarrow 1$
for $d = 0$ to $|A|$ **do**
  $P(e_{k,n}, D = d) \Leftarrow \frac{d}{|A|} P(D = d)$
for $n = 1$ to $|A|$ **do**
  likelihoodRatio $\Leftarrow$
    $\left(\frac{P(D \leq t)}{P(D > t)}\right)\left(\frac{\sum_{d=t+1}^{|A|} P(e_{k,n}, D=d)}{\sum_{d=0}^{t} P(e_{k,n}, D=d)}\right)$
  if likelihoodRatio $> \mathcal{B}$ **then**
    for $d = 0$ to $|A|$ **do**
      $P(e_{k,n}, D = d) \Leftarrow$
        $P(e_{k,n}, D = d) P(\text{next } 0 | e_{k,n}, D = d)$
  else
    for $d = 0$ to $|A|$ **do**
      $P(e_{k,n}, D = d) \Leftarrow$
        $P(e_{k,n}, D = d) P(\text{next } 1 | e_{k,n}, D = d)$
    $k \Leftarrow k + 1$
  rejectionLine[n] $\Leftarrow k$
return rejectionLine

E. Implementation note

The fastest version of our algorithm is a version of the SEQUENTIAL algorithm that does not check its position in a decision matrix. Instead, it only checks whether the number of non-similar features sampled so far is equal to the minimum row number

in the appropriate column in the decision matrix that is equal to *NS*. In other words, we simply check whether we have only touch the upper rejection line (see Fig. 8 red line of *NS*). If we finish sampling all the corresponding features and we have not touched the upper line, the window is unquestionably similar. In fact, the exact Hamming distance is automatically obtained in such cases. There is a negligible increase in the average number of samples, as we do not stop on similar windows as soon as they are definitely similar. However, the event of similarity is so rare that the reduction in the running time of processing each sample, reduces the total running time.

V. Prior

The proposed frameworks are Bayesian, *i.e.* they use a prior on the distribution of the distances between two natural images, $P(D = d)$. The prior can be estimated, offline, by computing the exact distance between various patterns and windows. Another option is to use a non-informative prior, *i.e.* a uniform prior in which the probability for each possible distance is equal. Fig. 9 and Fig. 10 show that the true distribution of distances is not uniform. Nevertheless, Fig. 17 shows that even though we use an incorrect (uniform) prior to parameterize the algorithm, we obtain good results. It should be stressed that other fast methods assume certain characteristics of images. For example, Hel-Or and Hel-Or [23] assume that the first Walsh-Hadamard basis projections (according to their specific order) contain enough information to discriminate most images. Mascarenhas et al. [28], [29] assume that images are binary or Gaussian distributed. In addition, they assume that all similar images have exactly the same pairwise small distance, while all two non-similar images have exactly the same large distance. By explicitly using a prior our method is more general.

For each distance measure and pattern size, we estimated the prior using a database of 480 natural images. First, outlier noise was added to each image. To simulate such noise we chose a different image at random from the test database and replaced between 0% to 50% (the value was chosen uniformly), with replacement, of the original image pixels with pixels from the different image in the same relative position.

For each image we computed the set of distances between two patterns (each a not too smooth randomly chosen 2D window from the image before the addition of the noise) and a sliding window over the noisy image. The prior that was used is a mixture model of this histogram and a uniform prior (with a very small probability for uniformity). We used a mixture model as we had almost no observations of small distances.

Fig. 9 shows that priors of the same Hamming distance for different pattern sizes are similar. Fig. 10 shows that as the distance measure becomes more invariant, the distances are smaller.

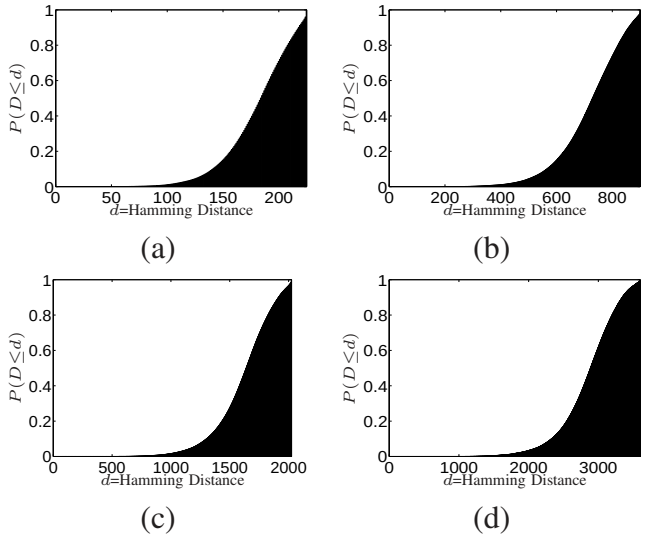


Fig. 9. Estimated cumulative PDFs of priors of *Thresholded Absolute Difference* Hamming distance with pixel similarity threshold equal 20 (pixels with intensity difference greater than 20 are considered non-similar) for patterns size: (a) 15×15 (b) 30×30 (c) 45×45 (d) 60×60. Note that the shapes of the priors are similar.

VI. Experimental Results

The proposed frameworks were tested on real images and patterns. The results show that the SEQUENTIAL algorithm is fast and accurate, with or without noise.

Recall that there are two kinds of errors: false negative (the event of returning *non-similar* on a similar window), and false positive (the event of returning *similar* on a non-similar window). A window is defined as similar to the pattern if and only if the Hamming distance between the window and the pattern is smaller or equal to the image similarity threshold, t. Note that in all the experiments (Figs. 1, 3, 4, 5 and 6) the similar windows are also visually similar to the pattern.

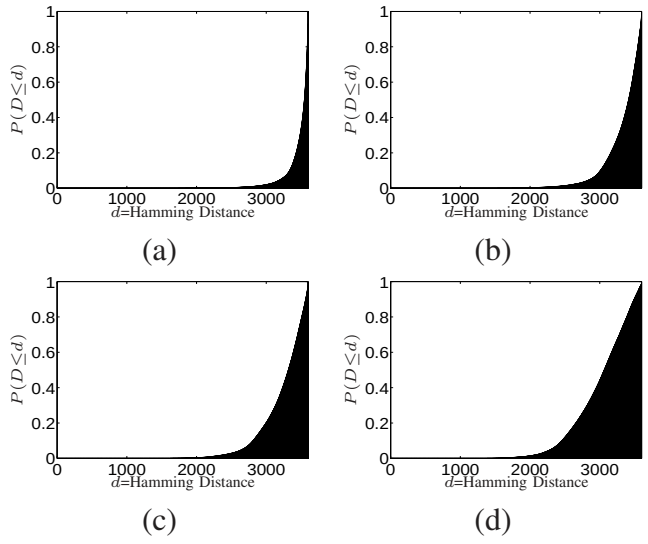


Fig. 10. Estimated cumulative PDFs priors of *Thresholded l_2 norm in L*a*b color space* Hamming distance for 60×60 patterns, with pixel similarity threshold equal: (a) 100 (b) 300 (c) 500 (d) 1000. Note that as the distance measure becomes more invariant (with a higher pixel similarity threshold), the distances are smaller.

We set the false positive error bound to zero in all experiments. Setting it to a higher value decreases the running time mostly for similar windows. As it is assumed that similarity between pattern and image is a rare event, the speedup caused by a higher bound on the false positive is negligible. We set the false negative error bound to 0.1%; *i.e.* out of 1000 similar windows, only one is expected to be classified as non-similar. Note that this small error rate enables the large reduction in the running time.

A typical pattern matching task is shown in Fig. 1. A non-rectangular pattern of 2197 pixels was sought in a sequence of 14, 640x480 pixel frames. We searched for windows with a *Thresholded Absolute Difference* Hamming distance lower or equal to 0.4×2197, *i.e.* less than 40% outlier noise such as out of plane rotation, shading, spectral reflectance, occlusion, etc. Two pixels were considered non-similar if their absolute intensity difference was greater than 20, *i.e.* $p = 20$. The SEQUENTIAL algorithm was parameterized with P-SPRT (see Section IV-D), a uniform prior and false negative error bound of 0.1%. Using the parameterized SEQUENTIAL algorithm, the pattern is found in 9 out of 11 frames in which it was present, with an average of only 19.70 pixels examined per window instead of 2197 needed for the exact distance computation. On a Pentium 4 3GHz processor, detection of the pattern proceeds at 0.022 seconds per frame. The false positive error rate was 0%. The false negative error rate was 0.28%. Note that due to image smoothness, there are several similar windows in each frame near the sought object. The errors were mostly due to missing one of these windows. Although we use an incorrect (uniform) prior to parameterize the algorithm, we obtain excellent results. Other distances such as cross correlation (CC), normalized cross correlation (NCC), l_1, l_2, yielded poor results even though they were computed exactly (the computation took much longer).

More results are given in Figs. 3, 4, 5 and 6. All of these results are on 640x480 pixel images and use the SEQUENTIAL algorithm that was parameterized with P-SPRT (see Section IV-D), a uniform prior and false negative error bound of 0.1%. These results are also summarized in Table II. Comparison of the results using the estimated prior and the uniform prior is given in Fig. 11.

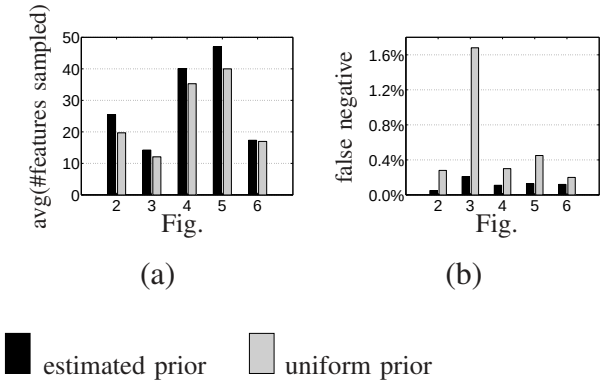


Fig. 11. Comparing optimal parameterization of the SEQUENTIAL algorithm with the estimated prior against optimal parameterization with a uniform prior in all figure experiments. In (a) the average number of features sampled per window was slightly smaller with the uniform prior. However, in (b) the error rate was higher with the uniform prior. Although higher, the error rate was still usually small. Thus, the performance using an incorrect (uniform) prior is still quite good.

Note that the parameters (pixel similarity threshold, p and relative image similarity threshold, $\frac{t}{|A|}$) are the same for each kind of distance. These parameters were chosen as they yield good performance for images experimentally. They do not necessarily give the best results. For example, on Fig. 3, using *Thresholded Absolute Difference* Hamming distance with pixel similarity threshold, p equal 100 and the image similarity threshold, t equal 0, the SEQUENTIAL algorithm ran only 0.013 seconds. The average number of pixels examined per window was only 2.85 instead of 1089 needed for the exact distance computation. The false negative error rate was 0%.

Another parameter that can be tuned is which pairs of pixels should set A contain when we use the *Monotonic Relations* Hamming distance. In all the experiments that use this distance, the pairs that were used are pairs of pixels belonging to edges, *i.e.* pixels that have a neighbor pixel, where the absolute intensity value difference is greater than 80. In all the experiments (except the experiment in Fig. 5) two pixels are considered neighbors if they are in the same 5×5 neighborhood. In the experiment in Fig. 5, two pixels are considered neighbors if they are in the same 11×11 neighborhood because pairs in the 5×5 neighborhood did not describe the pattern well. Thus, all parameters can be tuned for a specific pattern matching task. However, our work shows that for each of the proposed members of the Image Hamming Distance Family there is a standard set of parameters that usually yield good performance.

To illustrate the performance of Bayesian sequential sampling, we also conducted extensive random tests. The random tests were conducted mainly to illustrate the characteristics of the SEQUENTIAL algorithm and to compare its parameterization methods.

A test database (different from the training database that was used to estimate priors) of 480 natural images was used. We consider similar windows as windows with a Hamming distance smaller or equal to 50% of their size; *e.g.* a 60×60 window is considered similar to a 60×60 pattern if the Hamming distance between them is smaller/equal to 1800.

For comparison we also developed an optimal fixed size sampling algorithm, FIXED_SIZE (see Appendix I). Each test of the FIXED_SIZE algorithm or the SEQUENTIAL algorithm in Figs. 15, 16 and 17 was conducted using a different combination of members of the Image Hamming Distance Family and different sizes of patterns. For each such combination a prior was estimated (see Section V). In order to parameterize the FIXED_SIZE and the SEQUENTIAL algorithms, we used either the estimated prior or a uniform prior.

Each test of the parameterized algorithms was conducted by performing 9600 iterations (20 times for each image) as follows:

- A random not too smooth 2D window pattern was chosen from one of the images, Im, from the test database.
- Outlier noise was added to the image, Im. To simulate such noise we chose a different image at random from the test database and replaced between 0% to 50% (the value was chosen uniformly), with replacement, of the original image (*i.e.* Im) pixels with pixels from the different image in the same relative position.
- The pattern was sought for in the noisy image, using the parameterized SEQUENTIAL algorithm or the parameterized FIXED_SIZE algorithm.

In each test the false negative error rate and the average number of pixels examined per window were calculated. Overall, the results can be summarized as follows:

1) Even with very noisy images the SEQUENTIAL algorithm is very fast and accurate. For example, the average number of pixels sampled for pattern matching on 60×60 patterns with additive noise of up to 20 (each pixel gray value change can range from -20 to +20) and outlier noise of up to 50% was only 92.9, instead of 3600. The false negative error rate was only 0.09% (as mentioned above, the false positive error rate bound was always 0%).
2) The SEQUENTIAL algorithm is much faster than the FIXED_SIZE algorithm, with the same error rates. In addition, usually the SEQUENTIAL algorithm is less sensitive to incorrect priors (see Fig. 15).
3) The performance of the near-optimal solution, P-SPRT, is good (see Fig. 16).
4) The average number of features examined per window is slightly smaller with the uniform prior. However, the error rate is higher (although still small). Thus, there is not a substantial difference in performance when using an incorrect (uniform) prior (see Figs. 11,17).

To further illustrate the robustness of the method we conducted another kind of experiment. Five image transformations were evaluated: small rotation; small scale change; image blur; JPEG compression; and illumination. The names of the datasets used are *rotation*; *scale*; *blur*; *jpeg*; and *light* respectively. The *blur*, *jpeg* and *light* datasets were from the Mikolajczyk and Schmid paper [14]. Our method is robust to small but not large geometrical transforms.

TABLE II

SUMMARY OF FIGURE RESULTS

Fig.	(a) Distance type	(b) $\lvert A\rvert =$ Set Size	(c) Max Diff (%)	(d) False Negative (%)	(e) Average Features Sampled	(f) Offline Time (seconds)	**(g) Online Time (seconds)**
1	20TAD$^{(1)}$	2197	40	0.28	19.70	0.067	**0.022**
3	20TAD$^{(1)}$	1089	40	1.68	12.07	0.018	**0.019**
4	MR$^{(2)}$	631	25	0.30	35.28	0.009	**0.021**
5	MR$^{(2)}$	9409	25	0.45	39.98	1.219	**0.037**
6	LD-20TAD$^{(3)}$	714	5	0.20	16.98	0.007	**0.064**

(a) Distance types:
1) 20TAD - *Thresholded Absolute Difference*, with threshold(p) of 20.
2) MR - *Monotonic Relations*.
3) LD-20TAD - *Local Deformations* variant of *Thresholded Absolute Difference*, with threshold(p) of 20.

(b) Size of the set of spatial coordinates of features, *i.e.* number of pixels in *Thresholded Absolute Difference* distances, or number of pairs of pixels in *Monotonic Relations* Hamming distance.

(c) Maximum percentage of pixels, or pairs of pixels, that can be different in similar windows. For example, in Fig. 1, similar windows Hamming distance is less than $(\frac{40}{100})2197 = 878$.

(d) The false negative error rate (percentage of similar windows that the algorithm returned as non-similar). For example, in Fig. 1, on average out of 10000 similar windows, 28 were missed. Note that due to image smoothness, there were several similar windows in each image near each sought object. The errors were mostly due to missing one of these windows.

(e) Average number of pixels sampled in *Thresholded Absolute Difference* distances, or average number of pairs of pixels sampled in *Monotonic Relations* Hamming distances.

(f) Running time of the parameterization of the SEQUENTIAL algorithm. In addition, in *Monotonic Relations* distances it also includes the running time of finding the pairs of pixels that belong to edges.

(g) Running time of pattern detection using the SEQUENTIAL algorithm, where each image is 640x480 pixels in size.

Thus, it did not perform well on the geometrical changes datasets from the Mikolajczyk and Schmid paper [14]. We created two datasets with small geometrical transforms: a *scale* dataset that contains 22 images with an artificial scale change from 0.9 to 1.1 in jumps of 0.01; and a *rotation* dataset that contains 22 images with an artificial in-plane rotation from -10° to 10° in jumps of 1° (see for example Fig. 14).

For each collection, ten rectangular patterns were chosen from the image with no transformation. The pairs that were used in the set of each pattern were pairs of pixels belonging to edges, *i.e.* pixels that had a neighbor pixel, where the absolute intensity value difference was greater than 80. Two pixels, (x_2, y_2), (x_1, y_1) are considered neighbors if their l_∞ distance: $\max(|x_1 - x_2|, |y_1 - y_2|)$ is smaller or equal to 2. We searched for windows with a *Monotonic Relations* Hamming distance lower or equal to $0.25 \times |A|$. In each image we considered only the window with the minimum distance as similar, because we knew that the pattern occurred only once in the image. The SEQUENTIAL algorithm was parameterized using P-SPRT (see Section IV-D) with input of a uniform prior and a false negative error bound of 0.1%. We repeated each search of a pattern in an image 1000 times.

We defined two new notions of performance: miss detection error rate and false detection error rate. As we know the true homographies between the images, we know where the pattern pixels are in the transformed image. We denote a correct match as one that covers at least 80% of the transformed pattern pixels. A false match is one that covers less than 80% of the transformed pattern pixels. Note that there is also an event of no detection at all if the SEQUENTIAL algorithm does not find any window with a *Monotonic Relations* Hamming distance lower or equal to $0.25 \times |A|$. The miss detection error rate is the percentage of searches

of a pattern in an image that does not yield a correct match. The false detection error rate is the percentage of searches of a pattern in an image that yields a false match. Note that in the random tests that illustrated the performance of the Bayesian sequential sampling, it was not possible to use these error notions. In these tests we used a large number of patterns that were chosen randomly, thus we could not guarantee that the patterns did not occur more than once in these test images.

In the *light* and *jpeg* tests, the performance was perfect; *i.e.* 0% miss detection rate and 0% false detection rate. In the *blur* test, only one pattern was not found correctly in the most blurred image (see Fig. 14). The miss detection rate and false detection rate for this specific case was 99.6%. In all other patterns and images in the *blur* test, the miss detection rate and false detection rate was 0%. In the *scale* test, there was only one pattern with false detection in two images with scale 0.9 and 0.91. In the *rotation* test, there was only one pattern with false detection in images with rotation smaller than -2° or larger than +2°. Miss detection rates in the *scale* and *rotation* tests (see Fig. 12) were dependent on the pattern. If the scale change or rotation was not too big, the pattern was found correctly.

The average number of pair of pixels that the SEQUENTIAL algorithm sampled per window was not larger than 45 in all of the above tests. The average was 29.38 and the standard deviation was 4.22. In general, the number of samples decreased with image smoothness; *e.g.* it decreased with image blur, lack of light and JPEG compression (see for example Fig. 13). Note that the SEQUENTIAL algorithm using the *Monotonic Relations* Hamming distance stops as soon as there are not enough edge pairs of pixels in the same spatial position as in the pattern. Smoothness decreases the number of edge pairs of pixels; thus it decreases the average number of samples that the SEQUENTIAL algorithm samples.

Finally, Table III compares the running time of the two kinds of offline phases. *i.e.* it compares the running time of finding the optimal decision matrix (see Section IV-C) with the running time of finding the P-SPRT (near-optimal) decision matrix (see Section IV-D). Thus finding the P-SPRT decision matrix is an order of magnitude faster. All runs were conducted on a Pentium 4 3GHz processor.

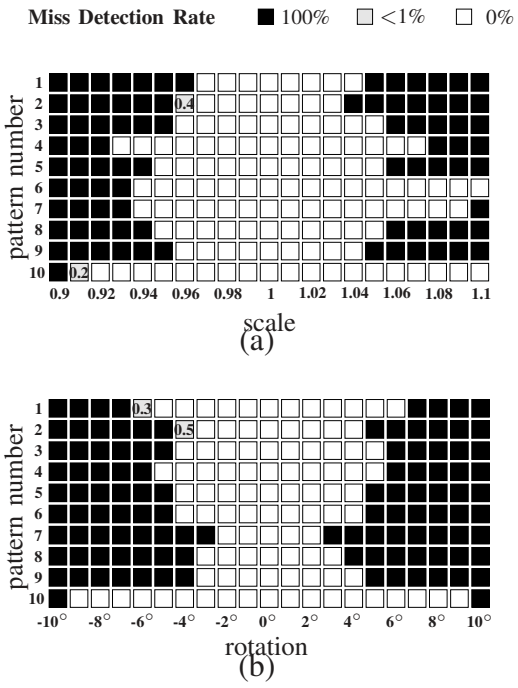


Fig. 12. (a) Miss detection error rates on the *scale* test. (b) Miss detection error rates on the *rotation* test.

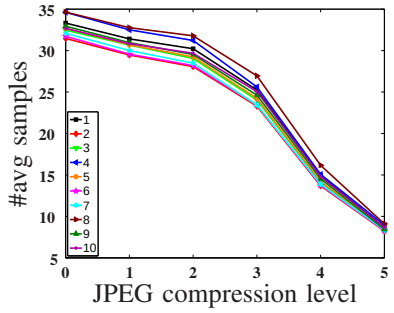


Fig. 13. Average number of pairs of pixels that the SEQUENTIAL algorithm sampled per window in the *jpeg* test.

VII. Conclusions

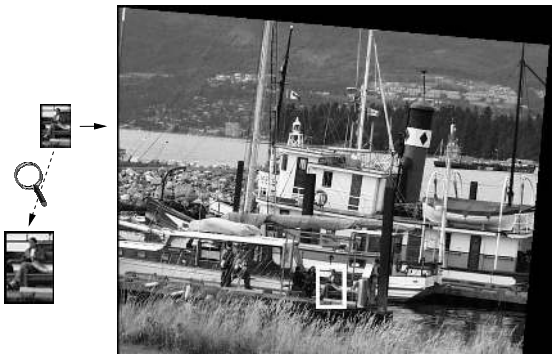
This paper introduced the "Image Hamming Distance Family". We also presented a Bayesian framework for sequential hypothesis testing on finite populations that designs optimal sampling algorithms. Finally, we detailed a framework that quickly designs a near-optimal sampling algorithm. We showed that the combination of an optimal or a near-optimal sampling algorithm and members of the Image Hamming Distance Family gives a robust, real time, pattern matching method.

Extensive random tests show that the SEQUENTIAL algorithm performance is excellent. The SEQUENTIAL algorithm is much faster than the FIXED_SIZE algorithm with the same error rates. In addition, the SEQUENTIAL algorithm is less sensi-

TABLE III
OFFLINE RUNNING TIME COMPARISON

$\|A\|$ - features' coordinates set size	500	1000	1500	2000	2500	3000
Offline P-SPRT (seconds)	0.005	0.018	0.042	0.075	0.14	0.17
Offline optimal (seconds)	7.510	49.220	154.520	653.890	2012.40	3504.97

(a)



(b)

Fig. 14. (a) The single false detection event on the *blur* test. (b) An example of detection on the *rotation* test. The image is 5° artificially in-plane rotated.

tive to incorrect priors. The performance of the near-optimal solution, P-SPRT, is good. It is noteworthy that performance using an incorrect (uniform) prior to parameterize the SEQUENTIAL algorithm is still quite good.

The technique explained in this paper was described in an image pattern matching context. However we emphasize that this is an example application. Sequential hypothesis tests on finite populations are used in quality control (*e.g.* [53]) , sequential mastery testing (*e.g.* [54], [55]) and possibly more fields. Thus the method can be used as is to produce optimal sampling schemes in these fields.

The project homepage is at: `http://www.cs.huji.ac.il/~ofirpele/hs`

ACKNOWLEDGMENT

We thank Professor Ester Samuel-Cahn for an enlightening discussion, Refael Vivanti for optimization and coding assistance and to Liat Pele, Amichai Zisken, Ori Maoz, Amit Gruber, Amnon Aaronsohn, Aviv Hurvitz, Eran Maryuma and Refael Vivanti for proofreading.

REFERENCES

[1] A. Wald, *Sequential Analysis*. Wiley, New York, 1947.
[2] D. I. Barnea and H. F. Silverman, "A class of algorithms for fast digital image registration," *IEEE Trans. Computer*, vol. 21, no. 2, pp. 179–186, Feb. 1972.
[3] J. Matas and O. Chum, "Randomized ransac with sequential probability ratio test," in *Proc. IEEE International Conference on Computer Vision (ICCV)*, vol. 2, October 2005, pp. 1727–1732.
[4] J. Šochman and J. Matas, "Waldboost - learning for time constrained sequential detection," in *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, June 2005, pp. 150–157.
[5] P. E. Anuta, "Spatial registration of multispectral and multitemporal digital imagery using fast fourier transform," *IEEE Trans. Geoscience Electronics*, vol. 8, pp. 353–368, 1970.
[6] J. P. Lewis, "Fast normalized cross-correlation," Sept. 02 1995. [Online]. Available: http://www.idiom.com/~zilla/Work/nvisionInterface/nip.pdf
[7] A. J. Ahumada, "Computational image quality metrics: A review." *Society for Information Display International Symposium*, vol. 24, pp. 305–308, 1998.
[8] B. Girod, *Digital Images and Human Vision, "Whats wrong with the mean-squared error?"*. MIT press, 1993, ch. 15.
[9] A. M. Eskicioglu and P. S. Fisher, "Image quality measures and their performance," *IEEE Trans. Communications*, vol. 43, no. 12, pp. 2959–2965, 1995.
[10] S. Santini and R. C. Jain, "Similarity measures," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, no. 9, pp. 871–883, Sept. 1999.
[11] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Image Understanding Workshop*, 1981, pp. 121–130.
[12] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions," in *Proceedings of the British Machine Vision Conference*, vol. 1, London, UK, September 2002, pp. 384–393.
[13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.
[14] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, 2005.

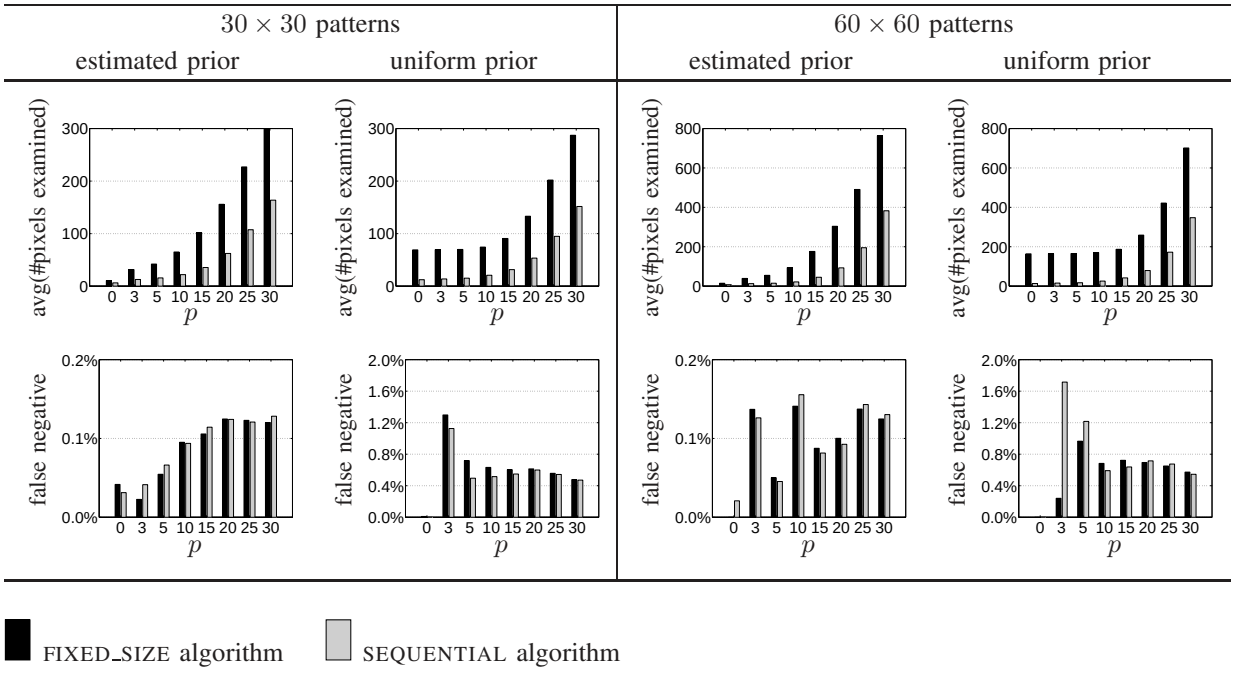


Fig. 15. Comparing the FIXED_SIZE algorithm with the SEQUENTIAL algorithm. Both algorithms were parametrized using the estimated or the uniform prior. The SEQUENTIAL algorithm is much faster than the FIXED_SIZE algorithm, with the same error rates. In addition, the SEQUENTIAL algorithm is less sensitive to incorrect (uniform) priors. In the top row we see that the average number of pixels examined per window was smaller using the SEQUENTIAL algorithm. In the bottom row we see that the error rate was the same in both algorithms. We can also see that the SEQUENTIAL algorithm is less sensitive to incorrect (uniform) priors (note that when the pixel similarity threshold is equal to 0, 3 and 5 the number of samples using the FIXED_SIZE algorithm increases when using the uniform prior). All tests were conducted using the *Thresholded Absolute Difference* Hamming distance with various pixel similarity thresholds p.

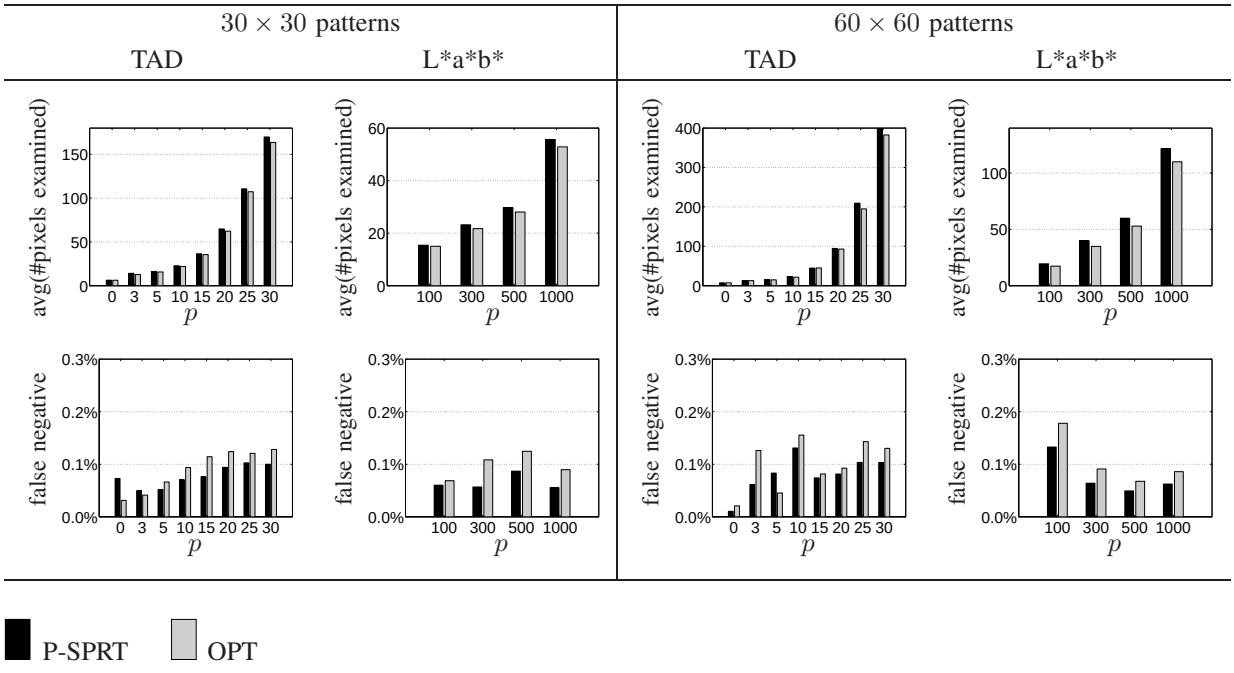


Fig. 16. Comparing the two parameterizations of the SEQUENTIAL algorithm: optimal and P-SPRT. The *Thresholded Absolute Difference* Hamming distance (TAD) or the *Thresholded Ratio* Hamming distance (L*a*b*) are used in the experiments with various pixel similarity thresholds p. All parameterizations were done with the estimated prior. P-SPRT samples a little more, with slightly smaller error rates.

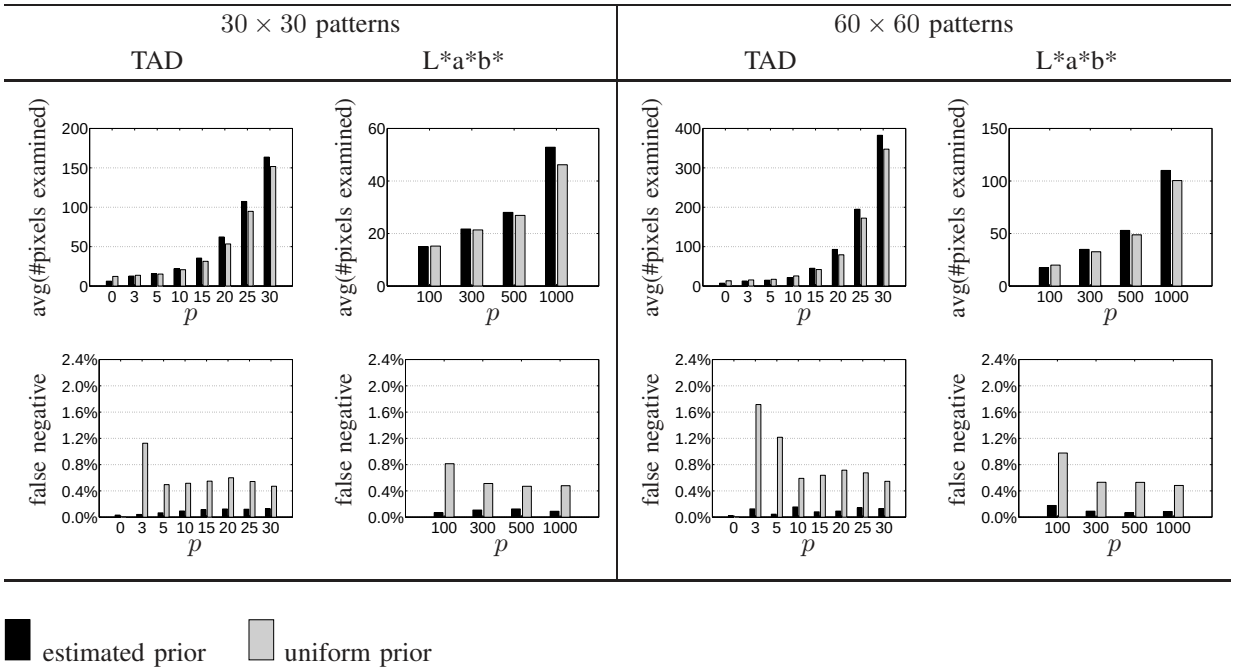


Fig. 17. Comparing optimal parameterization of the SEQUENTIAL algorithm with the estimated prior against optimal parameterization with a uniform prior. The *Thresholded Absolute Difference* Hamming distance (TAD) or the *Thresholded Ratio* Hamming distance (L*a*b*) are used in the experiments with various pixel similarity thresholds p. In the top row we see that the average number of pixels examined per window was slightly smaller with the uniform prior. However, in the bottom row we see that the error rate was higher with the uniform prior. Although higher, the error rate is still small. To conclude, the performance using an incorrect (uniform) prior is still quite good.

[15] H. Bay, T. Tuytelaars, and L. J. V. Gool, "Surf: Speeded up robust features." in *ECCV (1)*, 2006, pp. 404–417.

[16] V. Lepetit and P. Fua, "Keypoint recognition using randomized trees." *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1465–1479, 2006.

[17] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, "Local features and kernels for classification of texture and object categories: A comprehensive study," *Int. J. Comput. Vision*, vol. 73, no. 2, pp. 213–238, 2007.

[18] D. Keren, M. Osadchy, and C. Gotsman, "Antifaces: A novel, fast method for image detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 7, pp. 747–761, 2001.

[19] S. Romdhani, P. H. S. Torr, B. Scholkopf, and A. Blake, "Computationally efficient face detection," in *International Conference on Computer Vision*, 2001, pp. II: 695–700. [Online]. Available: http://dx.doi.org/10.1109/ICCV.2001.937694

[20] P. Viola and M. J. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Computer Vision and Pattern Recognition or CVPR*, 2001, pp. I:511–518.

[21] S. Avidan and M. Butman, "The power of feature clustering: An application to object detection," *Neural Information Processing Systems (NIPS)*, Dec. 2004. [Online]. Available: http://www.nips.cc/

[22] L. Liang, C. Liu, Y.-Q. Xu, B. Guo, and H.-Y. Shum, "Real-time texture synthesis by patch-based sampling," *ACM Trans. Graph.*, vol. 20, no. 3, pp. 127–150, 2001.

[23] Y. Hel-Or and H. Hel-Or, "Real-time pattern matching using projection kernels," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 9, pp. 1430–1445, Sept. 2005. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2005.184

[24] G. Ben-Artzi, H. Hel-Or, and Y. Hel-Or, "The gray-code filter kernels." *IEEE Trans. Pattern Analysis and Machine Intelligence.*, vol. 29, no. 3, pp. 382–393, 2007.

[25] M. Ben-Yehuda, L. Cadany, and H. Hel-Or, "Irregular pattern matching using projections." in *ICIP (2)*, 2005, pp. 834–837.

[26] S.-H. Cha, "Efficient algorithms for image template and dictionary matching," *J. Math. Imaging Vis.*, vol. 12, no. 1, pp. 81–90, 2000.

[27] B. Zitova and J. Flusser, "Image registration methods: a survey," *Image and Vision Computing*, vol. 21, no. 11, pp. 977–1000, Oct. 2003.

[28] J. A. G. Pereira and N. D. A. Mascarenhas, "Digital image registration by sequential analysis," *Computers and Graphics*, vol. 8, pp. 247–253, 1984.

[29] N. D. A. Mascarenhas and G. J. Erthal, "Image registration by sequential tests of hypotheses: The relationship between gaussian and binomial models," *Computers and Graphics*, vol. 16, no. 3, pp. 259–264, 1992.

[30] A. B. Lee, D. Mumford, and J. Huang, "Occlusion models for natural images: A statistical study of a scale-invariant dead leaves model," *International Journal of Computer Vision*, vol. 41, no. 1/2, pp. 35–59, 2001.

[31] E. Simoncelli and B. Olshausen, "Natural image statistics and neural representation," *Annual Review of Neuroscience*, vol. 24, pp. 1193–1216, May 2001.

[32] D. J. Field, "Relations between the statistics of natural images and the response properties of cortical cells," *J.Opt.Soc. Am. A,*, vol. 4, no. 12, pp. 2379–2394, 1987.

[33] J. G. Daugman, "Entropy reduction and decorrelation in visual coding by oriented neural receptive fields," *IEEE Trans. Biomedical Engineering*, vol. 36, no. 1, pp. 107–114, 1989.

[34] D. Siegmund, *Sequential Analysis, Test and Confidence Intervals*. Springer-Verlag, 1985.

[35] Y. Amit, *2D Object Detection and Recognition: Models, Algorithms, and Networks*. MIT Press, 2002.

[36] R. Zabih and J. Woodfill, "Non-parametric local transforms for

computing visual correspondence," in *ECCV*, 1994, pp. 151–158.
[37] B. Cyganek, "Comparison of nonparametric transformations and bit vector matching for stereo correlation." in *IWCIA*, 2004, pp. 534–547.
[38] Q. Lv, M. Charikar, and K. Li, "Image similarity search with compact data structures," in *CIKM*. New York, NY, USA: ACM Press, 2004, pp. 208–217.
[39] M. Ionescu and A. Ralescu, "Fuzzy hamming distance in a content-based image retrieval system," in *FUZZ-IEEE*, 2004.
[40] A. Bookstein, V. A. Kulyukin, and T. Raita, "Generalized hamming distance," *Inf. Retr.*, vol. 5, no. 4, pp. 353–375, 2002.
[41] A. Abhyankar, L. A. Hornak, and S. Schuckers, "Biorthogonal-wavelets-based iris recognition," A. K. Jain and N. K. Ratha, Eds., vol. 5779, no. 1. SPIE, 2005, pp. 59–67. [Online]. Available: http://link.aip.org/link/?PSI/5779/59/1
[42] P. Hough, "A method and means for recognizing complex patterens," *U.S. Patent 3,069,654*, 1962.
[43] S. D. Blostein and T. S. Huang, "Detecting small, moving objects in image sequences using sequential hypothesis testing," *IEEE Trans. Signal Processing*, vol. 39, no. 7, pp. 1611–1629, 1991.
[44] D. Shaked, O. Yaron, and N. Kiryati, "Deriving stopping rules for the probabilistic hough transform by sequential analysis," *Comput. Vis. Image Underst.*, vol. 63, no. 3, pp. 512–526, 1996.
[45] A. Amir and M. Lindenbaum, "A generic grouping algorithm and its quantitative analysis." *IEEE Trans. Pattern Analysis and Machine Intelligence.*, vol. 20, no. 2, pp. 168–185, 1998.
[46] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.
[47] H. Gharavi and M. Mills, "Blockmatching motion estimation algorithms-new results," *IEEE Trans. Circuits Syst. Video Techn.*, vol. 37, no. 5, pp. 649–651, 1990.
[48] G. Wyszecki and W. S. Stiles, *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Wiley, 1982.
[49] R. Brunelli and T. Poggio, "Face recognition: Features versus templates," *IEEE Trans. Pattern Analysis and Machine Intelligence.*, vol. 15, no. 10, pp. 1042–1052, 1993.
[50] Z. Govindarajulu, *Sequential Statistical Procedures*. Academic Press, 1975, pp. 534–536.
[51] A. Wald and J. Wolfowitz, "Optimum character of the sequential probability ratio test," *Ann. Math. Stat.*, vol. 19, pp. 326–339, 1948.
[52] E. L. Lehmann, *Testing Statistical Hypotheses*. John Wiley & Sons, 1959, pp. 104–110.
[53] H.-J. Mittag and H. Rinne, *Statistical Methods of Quality Assurance*. Chapman and Hall, 1993.
[54] H. J. Vos, "A bayesian sequential procedure for determining the optimal number of interrogatory examples for concept-learning," *Computers in Human Behavior*, vol. 23, no. 1, pp. 609–627, January 2007.
[55] C. Lewis and K. Sheehan, "Using bayesian decision theory to design a computerized mastery test," *Applied Psychological Measurement*, vol. 14, no. 4, pp. 367–386, 1990.

Ofir Pele received the BSc degree in computer science and life science and the MSc degree in computer science from the Hebrew University of Jerusalem, Israel, in 2003 and 2005, respectively. He is currently a PhD student in computer science at the Hebrew University of Jerusalem. His research interests include computer vision, sequential analysis and Bayesian statistics.

Michael Werman Ph.D. 1986, The Hebrew University, currently professor of computer science at The Hebrew University. His research has is mainly in designing computer algorithms and mathematical tools for analyzing, understanding and synthesizing pictures.

APPENDIX I
FIXED SIZE FRAMEWORK

We first present the FIXED_SIZE algorithm that tests for similarity using a fixed size sample. Then we evaluate its performance. Finally we show how to find the optimal parameters for the FIXED_SIZE algorithm.

A. The FIXED_SIZE algorithm

The FIXED_SIZE algorithm has threshold parameters l, u and a fixed sample size n. The framework computes optimal l, u and n offline. Then, the algorithm can quickly decide whether a pattern and a window are similar; *i.e.* if their Hamming distance is smaller or equal to the image similarity threshold, t.

The algorithm samples n corresponding features from the pattern and the window, computes their Hamming distance and decides according to the result whether to return *similar*, *non-similar* or to compute the exact distance.

Algorithm 6 FIXED_SIZE$_{l,u,n,t}$(pattern,window, A)

$k \Leftarrow 0$
for $i = 1$ to n **do**
  random sample uniformly and without replacement $(x, y)^m$ from A
  $k \Leftarrow k + sim$(pattern, window, $(x, y)^m$)
if $k \leq l$ **then**
  return *similar*
if $k \geq u$ **then**
  return *non-similar*
return $(HammingDistance_A(\text{pattern, window})) \leq t$

B. Evaluating performance for fixed parameters l, u, n

The performance of the algorithm is defined by its expected number of examined features and its error probabilities. The computation is similar to the one in the SEQUENTIAL algorithm (see Section IV):

$$P_{l,u,n}(\text{false negative}) = \frac{\sum_{k=u}^{n} \binom{n}{k} \Omega_S[k,n]}{P(D \leq t)} \quad (10)$$

$$P_{l,u,n}(\text{false positive}) = \frac{\sum_{k=0}^{l} \binom{n}{k} \Omega_{NS}[k,n]}{P(D > t)} \quad (11)$$

$$E_{l,u,n}(\#\text{examined features}) = n + P(\text{compute exact})(|A| - n) = n + \sum_{k=l+1}^{u-1} \binom{n}{k} (\Omega_S[k,n] + \Omega_{NS}[k,n])(|A| - n) \quad (12)$$

C. Finding optimal parameters l, u, n for the algorithm

We first find optimal thresholds l, u for a given sample size, n. Our goal is to minimize the expected number of examined features given bounds on the error probabilities, α, β:

$$\begin{aligned} &\arg\min_{l,u} E_{l,u}(\#\text{examined features}) \\ &s.t: \\ &P_{l,u}(\text{false negative}) \leq \alpha \\ &P_{l,u}(\text{false positive}) \leq \beta \\ &\text{Fixed number of samples } n \end{aligned} \quad (13)$$

$P_{l,u}$(false negative) (Eq. 10) monotonically decreases with the threshold u (the number of non-negative summands decreases). $P_{l,u}$(false positive) (Eq. 11) monotonically increases with the threshold l (the number of non-negative summands increases). $E_{l,u}$(#examined features) (Eq. 12) monotonically decreases with the threshold u and monotonically increases with the threshold l. Thus, we want l to be as large as possible, and u to be as small as possible.

The algorithm that chooses the optimal u, Alg. 7, starts by taking $u = n + 1$ and decreases it until $P_{l,u}$(false negative) is too high. The algorithm that chooses optimal l, Alg. 8 starts by taking $l = -1$ and increases it until $P_{l,u}$(false positive) is too high.

Algorithm 7 opt_u(n, t, α, P)

$err \Leftarrow 0$
$nCk \Leftarrow 1$ {The current n choose k}
for $k = n$ to 0 **do**
  $err \Leftarrow err + \frac{nCk \times \Omega_S[k,n]}{P(D \leq t)}$
  if $err > \alpha$ **then**
    return $k + 1$
  $nCk \Leftarrow nCk \times \left(\frac{k}{n-k+1}\right)$
return k

Algorithm 8 otp_l(n, t, β, P)

$err \Leftarrow 0$
$nCk \Leftarrow 1$ {The current n choose k}
for $k = 0$ to n **do**
  $err \Leftarrow err + \frac{nCk \times \Omega_{NS}[k,n]}{P(D>t)}$
  if $err > \beta$ **then**
    return $k - 1$
  $nCk \Leftarrow nCk \times \left(\frac{n-k}{k+1}\right)$
return k

In order to find the optimal n we compute optimal l, u and the expected number of examined features for each $n = 1 \ldots |A|$. Finally, we choose the n that minimizes the expected number of examined features. Note that the search can be terminated as soon as the current minimum of the expected number of examined features is smaller than n.

The intermediate sums $\Omega_S[k, n]$ and $\Omega_{NS}[k, n]$ (see Eq. 2) are computed for all possible k and n with a dynamic programming algorithm with a time complexity of $O(|A|^3)$. The algorithms that compute optimal l, u for each n (Algs. 8, 7) have a time complexity of O($|A|$). These algorithms run a maximum of $|A|$ times. Thus, finding optimal n, l, u has a time complexity of O($|A|^3$). It should be noted that the search for the optimal parameters is done offline. The user can employ the FIXED_SIZE algorithm parameterized with the optimal l, u, n, to quickly detect patterns in images.

APPENDIX II
HOUGH TRANSFORM COMPUTATION OF HAMMING DISTANCE

For simplicity we show how to use the Hough transform [42] to compute the *Thresholded Absolute Difference* Hamming Distance (see Section III-B) between a pattern and all windows in a 256 gray level image. The generalization to other members of the Image Hamming Distance Family is easy. We also analyze its time complexity.

List of symbols:

A = Set that contains spatial coordinates of pixels. $|A|$ is the size of this set.
R_{Im} = Number of rows in large image.
C_{Im} = Number of columns in large image.
L = A 256 array that contains lists of all pixel coordinates in the pattern that are similar to a specific gray value.
p = Pixel similarity threshold of the *Thresholded Absolute Difference* Hamming Distance.
H = The *Thresholded Absolute Difference* Hamming Distance Map, *i.e.* H$[r, c]$ is the *Thresholded Absolute Difference* Hamming Distance between the pattern and the window of the image Im whose top left pixel is $[r, c]$.

Algorithm 9 HoughTAD(pattern,image,p)

L$[0 \ldots 255] \Leftarrow$ empty list of indices.
for $(x, y) \in A$ **do**
  for $g =$ (pattern$[x, y] - p$) to (pattern$[x, y] + p$) **do**
    L$[g]$.insert($[x, y]$)
H$[0 \ldots R_{Im}, 0 \ldots C_{Im}] \Leftarrow |A|$
for $r = 1$ to R_{Im} **do**
  for $c = 1$ to C_{Im} **do**
    for it = (L[image$[r, c]$].begin) to (L[image$[r, c]$].end) **do**
      H$[r - it.r, c - it.c] \Leftarrow$ H$[r - it.r, c - it.c] - 1$

The first stage of Algorithm 9 which computes the array of lists, L has a time complexity of $O(|A|p)$. The second stage which computes the Hamming distance map, H has an expected time complexity of $O(R_{Im}C_{Im}(|A| - E[D]))$, where D is the random variable of the Hamming distance. Total expected time complexity is $O(|A|p + R_{Im}C_{Im}(|A| - E[D]))$. Average expected time complexity per window is $O(\frac{|A|p}{C_{Im}R_{Im}} + |A| - E[D])$. Since usually $\frac{|A|p}{C_{Im}R_{Im}}$ is negligible the average expected time complexity per window is $O(|A| - E[D])$

APPENDIX III
COMPUTATION OF PROBABILITIES OF THE SEQUENTIAL ALGORITHM

List of symbols:

- M = SEQUENTIAL algorithm decision matrix. $M[k, n]$ is the algorithm decision after sampling k non-similar corresponding features out of a total of n sampled corresponding features. The decision can be *NS*=return *non-similar*, *S*=return *similar* or *C*=continue sampling. See the graphical representation in Fig. 8.
- D = Random variable of the Hamming distance.

- t = Image similarity threshold, *i.e.* if the Hamming distance of two images is smaller or equal to t, then the images are considered similar. Otherwise, the images are considered non-similar.
- $\Psi_M[k,n]$ = Number of paths from the point $(0,0)$ to the point (k,n) that do not touch a stopping point (*S*, *NS*) in Fig. 8 on page 10.
- $e_{k,n}$= The event of sampling k non-similar corresponding features out of a total of n sampled corresponding features, in any specific order (for example, where the non-similar corresponding features are sampled first). Note that all orders of sampling have the same probability. See Eq. 1 on page 10.
- $\Omega_S[k,n]$, $\Omega_{NS}[k,n]$ = Intermediate sums defined in Eq. 2 on page 11.

$$P_M(\text{false negative}) \tag{14}$$
$$= P_M(\text{return } \textit{non-similar}|\text{images are similar}) \tag{15}$$
$$= P_M(\text{return } \textit{non-similar}|D \leq t) \tag{16}$$
$$= \sum_{d=0}^{|A|} P_M(\text{return } \textit{non-similar}, D=d|D \leq t) \tag{17}$$
$$= \frac{1}{P(D \leq t)} \sum_{d=0}^{t} P_M(\text{return } \textit{non-similar}, D=d) \tag{18}$$
$$= \frac{1}{P(D \leq t)} \sum_{d=0}^{t} \sum_{\substack{(k,n):\\ M(k,n)=NS}} \Psi[k,n] P(e_{k,n}, D=d) \tag{19}$$
$$= \frac{1}{P(D \leq t)} \sum_{\substack{(k,n):\\ M(k,n)=NS}} \Psi[k,n] \sum_{d=0}^{t} P(e_{k,n}, D=d) \tag{20}$$
$$= \frac{1}{P(D \leq t)} \sum_{\substack{(k,n):\\ M(k,n)=NS}} \Psi[k,n] \Omega_S[k,n] \tag{21}$$

$$P_M(\text{false positive}) \tag{22}$$
$$= P_M(\text{return } \textit{similar}|\text{images are non-similar}) \tag{23}$$
$$= P_M(\text{return } \textit{similar}|D > t) \tag{24}$$
$$= \sum_{d=0}^{|A|} P_M(\text{return } \textit{similar}, D=d|D > t) \tag{25}$$
$$= \frac{1}{P(D > t)} \sum_{d=t+1}^{|A|} P_M(\text{return } \textit{similar}, D=d) \tag{26}$$
$$= \frac{1}{P(D > t)} \sum_{d=t+1}^{|A|} \sum_{\substack{(k,n):\\ M(k,n)=S}} \Psi[k,n] P(e_{k,n}, D=d) \tag{27}$$
$$= \frac{1}{P(D > t)} \sum_{\substack{(k,n):\\ M(k,n)=S}} \Psi[k,n] \sum_{d=t+1}^{|A|} P(e_{k,n}, D=d) \tag{28}$$
$$= \frac{1}{P(D > t)} \sum_{\substack{(k,n):\\ M(k,n)=S}} \Psi[k,n] \Omega_{NS}[k,n] \tag{29}$$

$$E_M[\#\text{samples}] \tag{30}$$
$$= \sum_{d=0}^{|A|} E_M[\#\text{samples}, D=d] \tag{31}$$
$$= \sum_{d=0}^{|A|} \sum_{\substack{(k,n):\\ M(k,n)\in\{S,NS\}}} \Psi[k,n] P(e_{k,n}, D=d) n \tag{32}$$
$$= \sum_{\substack{(k,n):\\ M(k,n)\in\{S,NS\}}} \Psi[k,n] (\Omega_S[k,n] + \Omega_{NS}[k,n]) n \tag{33}$$

Appendix IV
Computation of expected additive loss in the backward induction algorithm

Let w_1 and w_0 be the loss weights for false positive error and false negative error respectively. The expected additive loss for each decision given that we sampled n samples, out of which k were non-similar is:

$$E[\text{addLoss}(S)|k,n] = P(D > t|e_{k,n}) w_1 \tag{34}$$
$$= \frac{P(D > t, e_{k,n})}{P(e_{k,n})} w_1 \tag{35}$$
$$= \frac{\sum_{d=t+1}^{|A|} P(D=d, e_{k,n})}{\sum_{d=0}^{|A|} P(D=d, e_{k,n})} w_1 \tag{36}$$
$$= \frac{\Omega_{NS}[k,n]}{\Omega_S[k,n] + \Omega_{NS}[k,n]} w_1 \tag{37}$$

$$E[\text{addLoss}(NS)|k,n] = P(D \leq t|e_{k,n}) w_0 \tag{38}$$
$$= \frac{P(D \leq t, e_{k,n})}{P(e_{k,n})} w_0 \tag{39}$$
$$= \frac{\sum_{d=0}^{t} P(D=d, e_{k,n})}{\sum_{d=0}^{|A|} P(D=d, e_{k,n})} w_0 \tag{40}$$
$$= \frac{\Omega_S[k,n]}{\Omega_S[k,n] + \Omega_{NS}[k,n]} w_0 \tag{41}$$

$$E[\text{addLoss}(C)|k,n] = \tag{42}$$
$$=1 + P(\text{next feature similar}|e_{k,n})\text{addLossOpt}(k,n+1)+ \tag{43}$$
$$P(\text{next feature non-similar}|e_{k,n})\text{addLossOpt}(k+1,n+1) \tag{44}$$
$$=1 + \frac{P(\text{next feature similar}, e_{k,n})}{P(e_{k,n})}\text{addLossOpt}(k,n+1)+ \tag{45}$$
$$\frac{P(\text{next feature non-similar}, e_{k,n})}{P(e_{k,n})}\text{addLossOpt}(k+1,n+1) \tag{46}$$
$$=1 + \frac{P(e_{k,n+1})}{P(e_{k,n})}\text{addLossOpt}(k,n+1)+ \tag{47}$$
$$\frac{P(e_{k+1,n+1})}{P(e_{k,n})}\text{addLossOpt}(k+1,n+1) \tag{48}$$
$$=1 + \frac{\Omega_S[k,n+1]+\Omega_{NS}[k,n+1]}{\Omega_S[k,n]+\Omega_{NS}[k,n]}\text{addLossOpt}(k,n+1)+ \tag{49}$$
$$\frac{\Omega_S[k+1,n+1]+\Omega_{NS}[k+1,n+1]}{\Omega_S[k,n]+\Omega_{NS}[k,n]}\times \tag{50}$$
$$\text{addLossOpt}(k+1,n+1) \tag{51}$$

APPENDIX V
BACKWARD INDUCTION SOLUTION THEOREMS

Theorem 1: Let M^* be a decision matrix which is the solution to Eq. 8. Then it is the solution to the original minimization problem Eq. 7 with $\alpha = P_{M^*}(\text{false negative})$ and $\beta = P_{M^*}(\text{false positive})$.

Proof: Let M' be another decision matrix of the same size and smaller/equal error probabilities. Then:

$$\begin{aligned}
&loss(M^*, w_0, w_1) \\
&\quad = P_{M^*}(\text{false negative})P(D \le t)w_0 + && (52)\\
&\quad\quad P_{M^*}(\text{false positive})P(D > t)w_1 + \\
&\quad\quad E_{M^*}[\#\text{samples}] \\
&\quad \le P_{M'}(\text{false negative})P(D \le t)w_0 + && (53)\\
&\quad\quad P_{M'}(\text{false positive})P(D > t)w_1 + \\
&\quad\quad E_{M'}[\#\text{samples}] \\
&\quad \le P_{M^*}(\text{false negative})P(D \le t)w_0 + && (54)\\
&\quad\quad P_{M^*}(\text{false positive})P(D > t)w_1 + && (55)\\
&\quad\quad E_{M'}[\#\text{samples}]
\end{aligned}$$

$$\Updownarrow$$

$$E_{M^*}[\#\text{samples}] \le E_{M'}[\#\text{samples}]$$

Explanations:
(52) Definition of the $loss$ function (Eq. 8)
(53) M^* is optimal (Eq. 8)
(54) $P_{M'}(\text{false negative}) \le P_{M^*}(\text{false negative})$
(55) $P_{M'}(\text{false positive}) \le P_{M^*}(\text{false positive})$

Theorem 2: Let M^* be the optimal decision matrix returned by Alg. 3 for some w_0, w_1. If $w_0 = \frac{|A|}{\alpha P(D \le t)}$ then $P_{M^*}(\text{false negative}) \le \alpha$. If $w_1 = \frac{|A|}{\beta P(D > t)}$ then $P_{M^*}(\text{false positive}) \le \beta$

Proof: Let M' be a decision matrix such that the SEQUENTIAL algorithm parametrized with it always returns the true answer by sampling all the corresponding features. Then:

$$\begin{aligned}
|A| &= \text{Loss}(M', w_0, w_1) && (56)\\
&\ge \text{Loss}(M^*, w_0, w_1) && (57)\\
&\ge P_{M^*}(\text{false negative})P(D \le t)w_0 && (58)\\
&= P_{M^*}(\text{false negative})P(D \le t)\frac{|A|}{\alpha P(D \le t)} && (59)
\end{aligned}$$

$$\Updownarrow$$

$$P_{M^*}(\text{false negative}) \le \alpha$$

Explanations:
(56) loss for taking all samples
(57) M^* is optimal
(58) part of sum of non-negatives
(59) $w_0 = \frac{|A|}{\alpha P(D \le t)}$

The proof of if $w_1 = \frac{|A|}{\beta P(D > t)}$ then $P_{M^*}(\text{false positive}) \le \beta$ is similar. ■

APPENDIX VI
COMPUTATION OF PROBABILITIES FOR THE P-SPRT FRAMEWORK

The likelihood ratio derivation:

$$\begin{aligned}
\lambda(e_{k,n}) &= \frac{P(e_{k,n}|D > t)}{P(e_{k,n}|D \le t)} && (60)\\
&= \frac{\sum_{d=t+1}^{|A|} P(e_{k,n}, D = d|D > t)}{\sum_{d=0}^{t} P(e_{k,n}, D = d|D \le t)} && (61)\\
&= \frac{\sum_{d=t+1}^{|A|} \frac{P(e_{k,n}, D=d, D>t)}{P(D>t)}}{\sum_{d=0}^{t} \frac{P(e_{k,n}, D=d, D<t)}{P(D \le t)}} && (62)\\
&= \left(\frac{P(D \le t)}{P(D > t)}\right)\left(\frac{\sum_{d=t+1}^{|A|} P(e_{k,n}, D = d)}{\sum_{d=0}^{t} P(e_{k,n}, D = d)}\right) && (63)
\end{aligned}$$

Explanations:
(61) disjoint and complementary events

(62) conditional probability definition

The initialization of the cache (in Alg. 5) is:

$$P(e_{1,1}, D = d) = P(e_{1,1}|D = d)P(D = d) \tag{64}$$
$$= \frac{d}{|A|}P(D = d) \tag{65}$$

The update of the cache (in Alg. 5), where $b = 0$ or 1, is:

$$P(e_{k+b,n+1}, D = d) \tag{66}$$
$$= P(D = d)P(e_{k+b,n+1}|D = d) \tag{67}$$
$$= P(D = d)P(e_{k,n}|D = d)P(\text{next } b|e_{k,n}, D = d) \tag{68}$$
$$= P(e_{k,n}, D = d)P(\text{next } b|e_{k,n}, D = d) \tag{69}$$

where:

$$P(\text{next } 0|e_{k,n}, D = d) \tag{70}$$
$$= \min\left(1, \frac{\max\left(0, ((|A| - d) - (n - k))\right)}{|A| - n}\right) \tag{71}$$
$$P(\text{next } 1|e_{k,n}, D = d) \tag{72}$$
$$= \min\left(1, \frac{\max\left(0, (d - k)\right)}{|A| - n}\right) \tag{73}$$