IRREGULAR PATTERN MATCHING USING PROJECTIONS

M. Ben-Yehuda, L. Cadany, H. Hel-Or

Department of Computer Science
University of Haifa
Haifa, Israel 31905

Y. Hel-Or

School of Computer Science
The Interdisciplinary Center
Herzliya, Israel

ABSTRACT

Recently, a novel approach to pattern matching was introduced, which reduces time complexity by two orders of magnitude over traditional approaches. It uses an efficient projection scheme which bounds the distance between a pattern and an image window using very few operations on average. The projection framework combined with a rejection scheme allows rapid rejection of image windows that are distant from the pattern. One of the limitations of this approach is the restriction to square dyadic patterns. In this paper we introduce a scheme, based on this approach which allows fast search for patterns of any size and shape. The suggested method takes advantage of the inherent recursive tree-structure introduced in the original scheme.

1. INTRODUCTION

Pattern Matching finds appearances of a pattern within a source image. The pattern is typically a 2D image fragment, much smaller than the image. Finding a given pattern in an image is performed by scanning the entire image, and evaluating the similarity between the pattern and a local 2D window about each pixel. Using naive approaches, this task is computationally intensive. Heuristics are often introduced to overcome the time complexity.

	Naive	Fourier	New Approach
Avg # ops per pixel	$+:\ 2k^2$ $*:\ k^2$	$+:\ 36\log n$ $*:\ 24\log n$	$+:\ 2\log k+\epsilon$
Space	n^2	n^2	$2n^2\log k$
Int Ops	Yes	No	Yes
Run time 16×16	1.33 Sec.	3.5 Sec.	54 Msec
Run time 32×32	4.86 Sec.	3.5 Sec.	78 Msec
Run time 64×64	31.30 Sec.	3.5 Sec.	125 Msec

Table 1. Pattern matching approaches - comparison.

In [1, 2], a novel approach was introduced which reduces run times by almost 2 orders of magnitude as shown in Table 1. The approach is based on a projection scheme where lower bounds on the distance between a pattern and image windows are obtained by projection onto a set of kernels. The projection framework is combined with a rejection scheme which reject those windows whose distance bounds imply that they do not match the pattern. The set of projection kernels are chosen such that they are fast to apply. Therefore, tight lower bounds can be produced with very few operations, which in turn, enable very fast rejection of a large portion of the image. Experiments show that the approach is efficient even under very noisy conditions.

One of the limitations of the approach as suggested in [1, 2], is the restriction to dyadic square patterns. In this paper we introduce a new scheme, based on this approach which allows fast search for patterns of any size and shape. The suggested method takes advantage of the tree-structure introduced in the original scheme.

2. PATTERN MATCHING USING PROJECTIONS

Assume a $k \times k$ pattern $\mathbf{p}$ is to be sought within a given image. Pattern $\mathbf{p}$ is matched against a similarly sized window $\mathbf{w}$ at every location in the image. Referring to the pattern $\mathbf{p}$ and the window $\mathbf{w}$ as vectors in $\Re^{k^2}$, the Euclidean distance between them is given as:

$$d_E^2(\mathbf{p},\mathbf{w}) = \|\mathbf{p}-\mathbf{w}\|^2 \qquad (1)$$

The smaller the distance value, the more similar are $\mathbf{w}$ and $\mathbf{p}$. If the distance is below a given threshold, then window $\mathbf{w}$ is considered similar to pattern $\mathbf{p}$.

Now, assume that $\mathbf{p}$ and $\mathbf{w}$ are not given, but only the values of their projection ${\mathbf{v}_1}^T\mathbf{p}$ and $\mathbf{v}_1^T\mathbf{w}$ onto a particular projection vector $\mathbf{v}_1$, where $\|\mathbf{v}_1\| = 1$. Since the Euclidean distance is a norm, it follows from the Cauchy-Schwartz inequality, that a lower bound on the actual Euclidean distance can be inferred from the projection values [2]:

$$d_E^2(\mathbf{p},\mathbf{w}) \geq d_E^2(\mathbf{v}_1^T\mathbf{p},\mathbf{v}_1^T\mathbf{w}) \qquad (2)$$

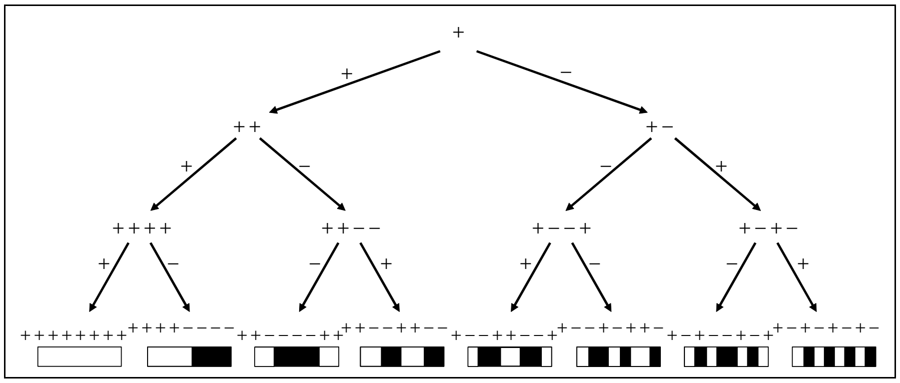

Fig. 1. The tree-scheme for computing projections of all windows of a 1D signal onto all WH kernels of order 8.

If an additional projection vector $\mathbf{v}_2$ is given along with the projection values: $\mathbf{v}_2^T\mathbf{p}$ and $\mathbf{v}_2^T\mathbf{w}$, it is possible to tighten the lower bound on the distance. Define the distance vector $\mathbf{d} = \mathbf{w} - \mathbf{p}$, and assume that the projection values of $\mathbf{d}$ onto r orthonormal projection vectors are given:

$$M^T\mathbf{d} = \mathbf{b}$$

$M = [\mathbf{v}_1\ \mathbf{v}_2\ \cdots \mathbf{v}_r]$ is a matrix of the r orthonormal projection vectors, and $\mathbf{b} = [b_1\ b_2\ \cdots b_r]$ is a vector of projection values $b_i = \mathbf{v}_i^T\mathbf{d} = \mathbf{v}_i^T\mathbf{w} - \mathbf{v}_i^T\mathbf{p}$. It is straightforward to verify that the lower-bound LB_E on the distance is:

$$d_E^2(\mathbf{p},\mathbf{w}) = \mathbf{d}^T\mathbf{d} \geq \mathbf{b}^T\mathbf{b} = LB_E^{(r)}(\mathbf{p},\mathbf{w})$$

Thus

$$LB_E^{(r)}(\mathbf{p},\mathbf{w}) = \sum_{i=1}^{r}(v_i^T\mathbf{p} - v_i^T\mathbf{w})^2 \qquad (3)$$

Note, that as the number of projection vectors increases, the lower bound on $d_E(\mathbf{p},\mathbf{w})$ tightens. In the extreme, when $r = k^2$ and the projection vectors are linearly independent, the lower bound reaches the actual Euclidean distance.

This is the basis for the Pattern Matching process:

1. The sought pattern $\mathbf{p}$ is projected onto a set of n normalized projection vectors $\{\mathbf{v}_i\}$, resulting in n values: $\hat{p}^{\,i} = \mathbf{v}_i^T\mathbf{p}$, for $i = 1 \ldots n$.

2. All image windows $\{\mathbf{w}_j\}$ are projected onto the first projection vector $\mathbf{v}_1$: $\hat{w}_j^1 = \mathbf{v}_1^T\mathbf{w}_j$

3. This projection sets a lower bound on the true distance between each window $\mathbf{w}_j$ and the pattern: $LB_E^{(1)}(\mathbf{p},\mathbf{w}_j) = (\hat{w}_j^1 - \hat{p}^1)^2$. According to the lower bound values, any window j whose $LB_E^{(1)}$ value is greater than the given threshold can be rejected.

4. The windows of the image that have not been rejected, are projected onto the second projection vector $\mathbf{v}_2$: $\hat{w}_j^2 = \mathbf{v}_2^T\mathbf{w}_j$. This produces updated lower bounds: $LB_E^{(1)} = LB_E^{(2)} + (\hat{w}_j^2 - \hat{p}^{\,2})^2$.

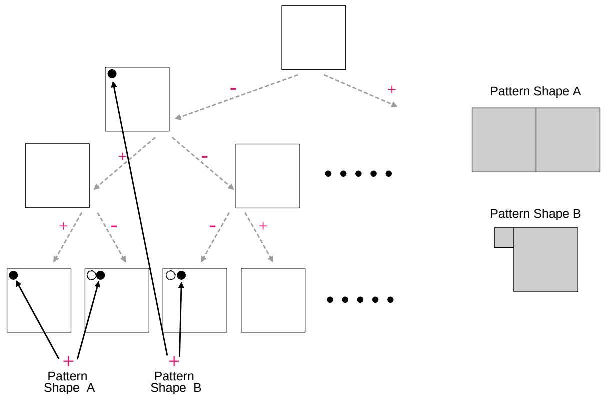

Fig. 2. Combining coefficients in the WH-tree allows detection of non-square patterns. Two examples are shown.

5. Steps 3 and 4 are repeated for the subsequent projection vector.

6. The process terminates after all n kernels have been processed or until the number of non-rejected image windows reaches a predefined value.

The efficiency of this Pattern Matching approach relies on choosing projection vectors, for which the lower bound becomes tight following very few projections and for which projections can be computed efficiently. In [1, 2] it was shown that the Walsh-Hadamard Kernels ordered in decreasing sequency, fulfil the tight bound requirement. Additionally, a method for efficiently computing the projection onto the Walsh-Hadamard kernels, was presented which reduces computation time to $O(1)$-$O(logk)$ operations per pixel per kernel. The projections are computed using a tree structure called the *Walsh Hadamard Tree*.

2.1. Projections Using the Walsh-Hadamard Tree

In order to efficiently compute the projections of all image windows onto all kernels, the Walsh-Hadamard Tree described in [2] is used.

Consider first, the 1D pattern matching case. Given a signal vector of length n, and a "pattern" vector of length k, the goal is to find appearances of the pattern in the signal. Towards this end, we project *each* window of the signal, of length k, onto a set of 1D Walsh-Hadamard basis vectors. These projections can be computed efficiently using the tree scheme depicted in Figure 1. There are $\log_2(k) + 1$ levels in the tree. Every node in the tree represents a vector of intermediate values used in the computation. The root node represents the original signal vector. The i-th leaf represents a vector containing the projection values of all signal windows onto the i-th Walsh-Hadamard basis vector. The symbols $+$ and $-$ represent the operations performed at a given tree level. A symbol $+$ $(-)$ on an edge connecting nodes at level i and level $i+1$ denotes the computation performed on the signal at level i, in which to every entry x of the signal, the value of entry $x + \Delta$ is added (subtracted), where $\Delta = 2^i$. Thus, the 2 signals at level 1 are obtained by adding or subtracting consecutive entries in the signal of level 0 which is the original signal. The 4 signals at level 2 are obtained by adding/subtracting entries at distance 2 in the signals at level 1, and so on.

This approach can be extended to higher dimensions. Searching for a $k \times k$ pattern in a 2D image of size $n \times n$, each window of the image is projected onto a set of 2D Walsh-Hadamard kernels using a Walsh-Hadamard tree. However, for the 2D case, the tree depth is $2\log_2(k)$ rather than $\log_2(k)$, and there are k^2 leaf nodes. The operations performed at each level of the tree are now performed along the rows or along the columns of the image (see [2] for details).

The order of projection kernels (associated with the tree leaves) in both 1D and 2D, is crucial to the efficiency of the matching process. The operations $+/-$ on the tree branches were designed to create projections onto Walsh-Hadamard vectors ordered in increasing dyadic sequency.

Computations within the tree are typically performed by descending from a node in the tree to its child. This requires a single operation per pixel. Thus, for every kernel, $2\log_2(k)$ operations per pixel are required in the 2D case. Note, that due to the tree structure, intermediate computations actually serve many windows. Thus, following the projection onto the first kernel, the computation onto the second kernel requires only a *single* operation per pixel, since most of the intermediate computations have already been performed. Additionally, since a rejection scheme is incorporated into the computation, many windows are rejected following the first few projections. Thus the following projections require even fewer computations since they are performed only on the remaining windows. In practise it is shown that the number of operations per pixel required to complete the pattern detection process using the Walsh-Hadamard tree is $2\log k + \epsilon$.

3. NON-SQUARE PATTERN MATCHING

As described in [2] and above, Pattern Matching, assumes pattern and corresponding windows are square shaped of dyadic length. This seriously constrains pattern matching as often the desired pattern is irregularly shaped. We propose a pattern matching approach that overcomes this restriction. The approach relies on the Walsh-Hadamard tree described above. An important characteristic of the Walsh-Hadamard tree is its inherent recursive nature. The sub-trees from root to any level of the tree are themselves Walsh-Hadamard trees for smaller sized image windows. Within the Walsh-Hadamard tree containing $2log_2k + 1$ levels, developed for a window of size $k \times k$, the first $2log_2k - 2$ levels of the tree can be used to match patterns of size $\frac{k}{2} \times \frac{k}{2}$. This has been used for Multi-Scale Pattern Matching [2]. We exploit this in our non-square Pattern matching scheme.

The pattern is decomposed into dyadic components. Each component is matched with an appropriate sized image window. The same Walsh-Hadamard tree is used to compute the projection of all these image windows onto the projection kernels. The projection values may be located at the same node of the tree at different coordinates or even at different nodes of the tree. An example is shown in Figure 2. Pattern matching is based on calculating lower bounds and rejecting the non-square windows whose lower bound exceeds a threshold. Lower bounds on the actual Euclidean distance between the non-square pattern and window are computed from the projections of the dyadic components of the pattern and window. This can be implemented since the Euclidean distance is additive. Issues of kernel normalization and weighting must be carefully considered.

Pattern matching proceeds by calculating the first branch of the Walsh-Hadamard tree. This branch may now contain projection values of several dyadic components of the window. All these projection values are considered, and lower bounds updated accordingly, before the subsequent branch of the tree is computed.

3.1. Pattern Decomposition

The original non-square pattern is decomposed into dyadic components. From experimental results (see Figure 5b), we

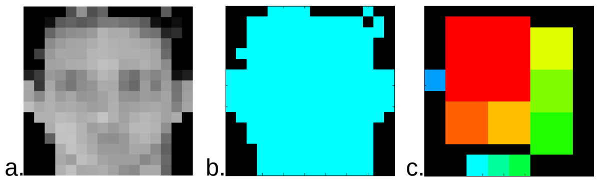

Fig. 3. a) Non-square pattern. b) Mask of pattern. c) Resulting decomposition of pattern. Maximum components was set to 10 and minimum size was set to 4.

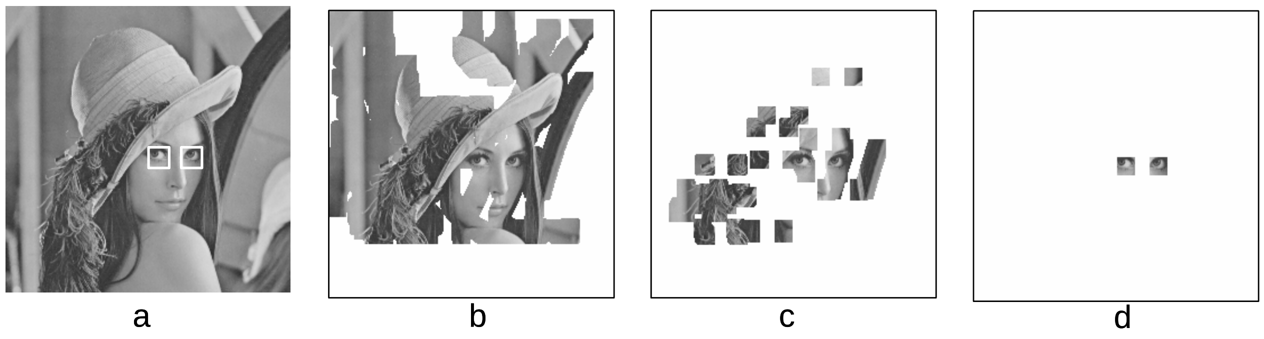

Fig. 4. Non-Square Pattern Matching. a) Image with non-square pattern marked. b-d) Steps in the pattern matching process. Number of remaining windows are b) 2970 c) 100 d) 1.

found that there is an advantage in computation and run-time, the larger the components of the pattern. Thus we use a greedy decomposition algorithm to extract the largest possible components first. We use a 2-pass algorithm to create a map of distances from every pixel in the pattern to the pattern boundary. The distance values may be considered as half the length of the maximal square centered at the pixel and bounded within the pattern. The largest bounded dyadic square is thus found and marked. The process is repeated with a new pattern obtained by removing the marked dyadic square from the original pattern. A sequence of dyadic components of decreasing size are thus extracted greedily from the pattern and form the dyadic decomposition. The process is repeated until the desired number of squares is reached, or the size of the components reaches the minimum size allowed. An example is shown in Figure 3.

4. RESULTS

The proposed scheme was tested on real images and patterns. As an example, Figure 4 shows an original 256x256 image within which a non-square pattern was marked. This example shows that patterns need not be a single connected component. In this case the, the pattern consists of 2 rectangular patches. Following the pattern matching approach described in this paper, non-square image windows were projected onto Walsh-Hadamard kernels and windows with values exceeding the given threshold were rejected. Process continued with remaining windows until one window remained. The remaining windows after each steps of the process is shown in Figures 4b-d. This behavior was consistent over many such examples including matching with noise contaminated images. Figure 5a plots the number of remaining window as a function of the number of projections for various noise levels.

We examined the effects of the size of components on the computation time. Results are shown in Figure 5b where the number of calculation steps is plotted as a function of the maximum allowed size of dyadic component of the pattern. As can be seen, there is an advantage in computation and run-time, when larger component sizes are allowed.

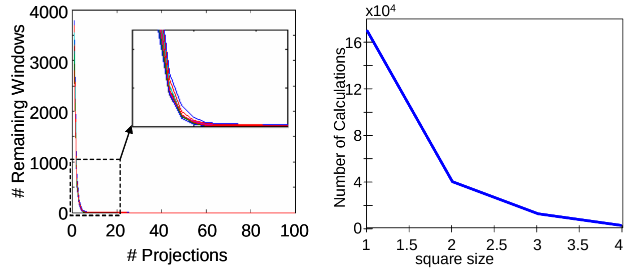

Fig. 5. a) Number of remaining windows following each projection. b) Computations as a function of maximum pattern component size.

5. CONCLUSION

The pattern-matching scheme proposed in this paper deals with irregularly shaped patterns. The pattern is decomposed into dyadic components which are searched for using the projection based pattern matching approach. A single Walsh-hadamard tree is used. This is possible due to the tree's inherent recursive nature. The advantage of the projection scheme is increased by the fact that the Walsh-Hadamard tree needs to be developed only up to the level associated with the maximum component size. This also implies that less run time memory is required.

6. REFERENCES

[1] Y. Hel-Or and H. Hel-Or, "Real time pattern matching using projection kernels," in *9th IEEE International Conference on Computer Vision*, Nice, France, Oct 2003, pp. 1486–1493.

[2] Y. Hel-Or and H. Hel-Or, "Real time pattern matching using projection kernels," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, To Appear.