QUIP: Querying Significant Patterns from Image Databases

Vishwakarma Singh Arnab Bhattacharya Ambuj K. Singh
Dept. of Computer Science, UCSB
{vsingh,arnab,ambuj}@cs.ucsb.edu

Chris Banna Geoffrey P. Lewis Steven K. Fisher
Dept. of Molecular, Cellular and Developmental Biology, UCSB
{banna,g_lewis,fisher}@lifesci.ucsb.edu

Abstract

Images have become an extremely important dataset in many areas of science including biology, geography and astronomy due to their ability to reveal spatial information not immediately available from other data sources. In this paper, we introduce a novel approach, QUIP (QUerying Image Patterns), to retrieve significant spatial patterns from a large collection of such images. Such an ability will provide important clues to the domain scientists regarding the underlying processes that produce those images.

The query pattern of interest is specified as a rectangular region from a tiled image. A scoring formula is designed to discriminate the significant foreground patterns from the irrelevant background of the region. Candidate database regions that match the query are translated into a score matrix of the pairwise aligned tiles. We show that the problem of finding the maximal scoring connected sub-region from the matrix is NP-hard and develop an effective dynamic programming heuristic. To assist the user, each retrieved database pattern is assigned a p-value to indicate its statistical significance. Finally, in order to accelerate QUIP, we adopt the threshold algorithm to efficiently retrieve the candidate database matches and a bounding method to speed up p-value computation.

We experiment with three datasets of microscopy images of retina. For each dataset, the results are significant for the domain scientists. Our method also has practical running time and scales well with database and query sizes.

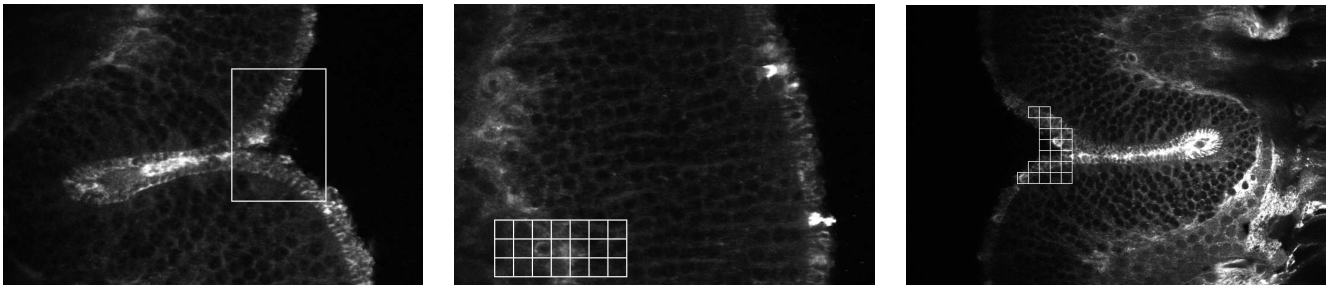
1. Motivation

Any image repository, be it a database of biomedical images, an aerial photo archive or a surveillance system, must support analysis, comparison, retrieval, and mining of images in order to be more useful than an online file cabinet. Over the years, images have become an extremely important dataset for scientific explorations due to their ability to reveal spatial information and relationships not immediately available from other data sources. Automated analysis tools have the potential for changing the way images are used to answer domain-specific questions. For example, in aerial photography or in facial images, they are used for automatic image annotations [21, 32]; in astronomical satellite images, they are used to accurately detect point sources and their intensities [14]; and in biology, they are used for mining interesting patterns of cell and tissue behavior or for high-throughput identification of abnormal samples [4, 11]. Quantitative analysis is central to studying changes in cells, tissues and organs and may provide the impetus needed to emulate the success of applying computational methods in genetics to the field of microscopy.

Querying specific patterns within a region helps domain scientists understand useful information and trends otherwise masked in the whole image. For example, biologists may be interested in examining only a single kind of cell in an image and finding similar patterns of just that cell from a bioimage database or astronomers may be interested in detecting only a particular kind of stars. Whole image matching methods are not relevant in this context. Image pattern matching can also be used to quantify the temporal, experimental, and inter-species differences.

Figure 1 shows examples of cross-sections of feline retina with different rearrangements of peanut-agglutinin protein. Such microscopy images are used to understand the change in the distribution of proteins in different experimental conditions, such as when the retina is detached or when different treatments are used, or to visualize specific cells across these conditions [11]. The ability to discriminate and classify on the basis of patterns (e.g., the intensity of fluorescence labeling or texture as shown in the figure) can help identify differences and similarities of various cellular processes within the retina. Determining if the pattern of a particular tissue under retinal detachment conditions is specific to cats, or is a common occurrence across mammals including humans may help significantly in developing cures for retinal diseases [11].

The marked region in Figure 1(a) depicts a fold in the retinal tissue. If a retinal detachment encompasses a large percentage of the total retinal area, then the retina can pull back upon itself and cause a fold. Within the folded tissue, the rearrangement of protein labeling is different from unfolded tissues. Knowledge of how the distribution and expression levels within a fold differs provides insight into how cells respond



(a) Query pattern (b) Result with distance-based scheme on entire region (c) Result with scoring-based scheme and sub-region finding

Figure 1. (a) Example of a biologically interesting pattern. The marked pattern highlights a fold of the retinal tissue labeled with peanut-agglutinin conjugated to a fluorescent probe. (b) Retrieved result when distance-based matching on entire region is used. (c) Retrieved result when score-based matching on sub-regions is used.

to injury [11]. Querying similar patterns from a database of such micrographs helps to answer these questions. Figure 1(b) shows that distance-based whole region matching schemes are not sufficient in this context. Figure 1(c) shows an actual result produced by our proposed method *QUIP* (*QUerying Image Patterns*).

In order to find meaningful results such as these, access methods in image databases have to provide the following:

- It is necessary to develop a *scoring* scheme to separate relevant foreground patterns from irrelevant background. Distance-based methods include background regions that contain little or no information and thus produce many false results. A mechanism to assign negative or low scores to similar background matches and high positive scores to foreground matches should be used. Score-based methods are also sensitive at finding interesting regions irrespective of their extent.

- The result of a query can be a *sub-region*. As shown in Figure 1(c), the pattern is best captured by the shape outlined in the figure. The portion of the region not highlighted constitutes the background and should, therefore, be discarded. Scoring the pairwise aligned tiles of the query to a database region define a *score matrix*. We show that finding the maximal scoring connected sub-region by examining all possibilities from this score matrix is NP-hard. This means that an effective heuristic should be devised to identify the most interesting sub-region.

- The method should handle *translation*, *rotation* and *reflection*. Images are often not registered or oriented properly. *Magnification* needs to handled as well to ensure that the result contains patterns originating from similar underlying mechanisms (e.g., same cells or tissues).

- The results from the method should be *semantically meaningful* and *statistically significant*. It is valuable to compute the p-value (a measure of statistical significance) of the result in order to understand whether the queried pattern is rare or frequent in the database. Scanning the entire database to ascertain the p-value of the score of the match is not practical; consequently, an efficient approximation algorithm must be devised.

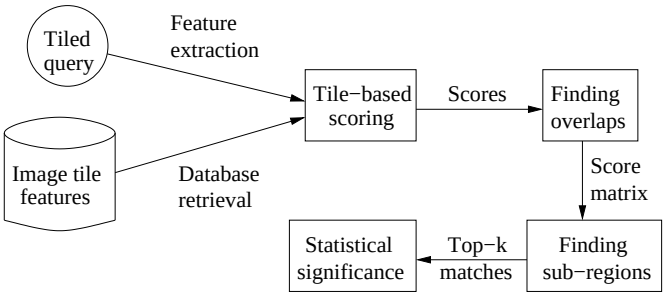


Figure 2. Overview of *QUIP*.

- The method should be *scalable* to database and query sizes and the running times should be *practical*. It is necessary to identify and prune regions of the database that are guaranteed to be absent from the top-k matches for the query. Since scoring-based mechanisms are used, fast similarity search methods that can incorporate such scores need to be designed.

Figure 2 shows a schematic view of our method *QUIP* that achieves all the above goals. Both the database images and the query pattern are tiled; image features are extracted from these tiles and indexed (Section 2). A query is specified as a rectangular region of tiles. A mechanism is designed to score the match of a query tile against a database tile (Section 3). An algorithm to choose the maximal scoring connected sub-region is applied (Section 4) to the score matrices produced by overlapping candidate database regions with the query (Section 5). For each of the top-k results retrieved, the statistical significance of the match is computed against the database (Section 6).

2. Tiling and Image Features

Comparison of a pair of image regions requires *region specification*, an *image feature* extracted from the region and a *distance metric*. In addition to these, biomedical images bring

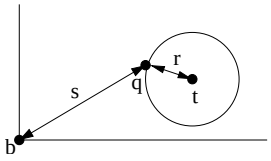


Figure 3. Scoring a query tile $\mathbf{q}$ against a database tile $\mathbf{t}$. $\mathbf{b}$ denotes the perfect "background" tile. $\mathbf{score(q,t) = s - \lambda r - c}$.

some subtle but practical complications: i) *Orientation*: Most images are not oriented in the same manner; for some kind of images, there may not be any canonical orientation at all [4]. ii) *Registration*: It is not always possible to register the images; the same areas of the image will not contain the same tissue or the same cells due to individual variations [11]. iii) *Magnification*: Images may not be of the same magnification [11].

We first adjust for the magnification of each image by rescaling them using the GraphicsMagick CubicFilter [1] and then partition the scaled image into non-overlapping tiles such that the physical size of each tile is the same. A region is specified as a collection of tiles. Tiling also helps to bypass the practical issues of image registration, image orientation, and image segmentation. The optimal tile size depends on the nature of images. The tiles must be large enough to capture the spatial patterns. On the other hand, making a tile too large will confuse between more than one pattern and will also decrease the specificity of characterizing a region. Based on domain knowledge, we set the tile size to 32 $\times$ 32 pixels.

The different image features that we experimented with include (i) Gabor texture [21], (ii) Simple intensity binning, and (iii) Color Structure Descriptor (CSD) from MPEG-7 [23]. We evaluated the effectiveness of each feature by examining tiles that are retrieved by running a similarity search on a database with that feature. Gabor texture feature failed when there was no uniform texture across the tile. For multi-channel images, where each channel is pseudo-colored, the texture features performed poorly. Simple intensity binning worked better with colored tiles. However, it could not capture the different patterns. The CSD maintains color histograms that is sensitive to both the color distribution of the image and the local spatial structure of the color. Here, the image is first transformed into the HMMD color space. Then, for each position of a sliding 8 $\times$ 8 structural element, if a color is present, its corresponding bin is incremented. CSD captured the patterns well for both color and gray-scale images.

In order to avoid the "curse of dimensionality" associated with high-dimensional feature vectors like the 256-dimensional CSD features, we applied principal component analysis (PCA) [17]. As a rule of thumb, we retained at least 90% of the energy from each dataset. We indexed the resulting feature vectors using an R-tree [13] and used the L_1 metric to compute the distances.

3. Scoring Mechanism

In this section, we devise an automatic scoring mechanism that translates the distance between two tiles into a score. For each tile, a value is computed to measure its information content. Tiles with little or no information form the background and, therefore, should get negative or low scores whereas tiles with more pattern information should get high positive scores when matched with similarly distant tiles. Also, for a particular tile, more distant tiles should score less than nearer ones. These ideas are captured in our scoring mechanism.

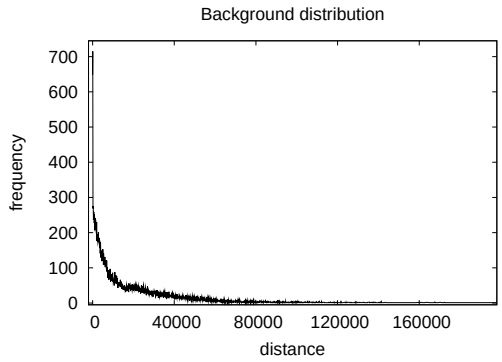


Figure 4. Distribution of database tiles according to their distances from the perfect background tile. For each tile, this distance measures its "backgroundness" value.

The tile space can be assumed to have two distributions. The first distribution is that of similar tiles from a database tile which we call the *true distribution*. We model this as an exponential distribution. Our reasons for choosing this distribution are two-fold: first, its simplicity, and second, its utility in capturing small variations over related images. For a database tile t and a query tile q (Figure 3), if $r = d(q,t) =$ the L_1 distance between the feature values of q and t, then we can characterize this distribution as

$$P(q|true\ distribution) = \lambda_1 e^{-\lambda_1 r} \quad (1)$$

The second distribution is that of background tiles in the database, which we call the *background distribution*. To measure the "backgroundness" of a tile, we computed the total intensity of all its pixels. The perfect background tile b in Figure 3 is all black and has 0 total intensity. As shown in Figure 4, the distance distribution from b was found to be exponential. If $s = d(q,b) =$ the difference of the total intensities of q and b, then this distribution is modeled as

$$P(q|background\ distribution) = \lambda_2 e^{-\lambda_2 s} \quad (2)$$

The score of a query tile q matching a database tile t is given by the *log-odds ratio*:

$$\begin{aligned} score(q,t) &= ln \frac{P(q|true\ distribution)}{P(q|background\ distribution)} \\ &= \lambda_2 s - \lambda_1 r + ln(\lambda_1/\lambda_2) \quad (3) \end{aligned}$$

Since scoring is only used to discriminate between foreground and background matches and the actual score is not important, the scores can be conveniently translated and scaled with constants. Denoting λ_1/λ_2 by a constant λ, scaling by λ_2 and translating the score gives

$$\begin{aligned} score(q,t) &= s - \lambda.r - c \quad (4) \\ &= d(q,b) - \lambda.d(q,t) - c \quad (5) \end{aligned}$$

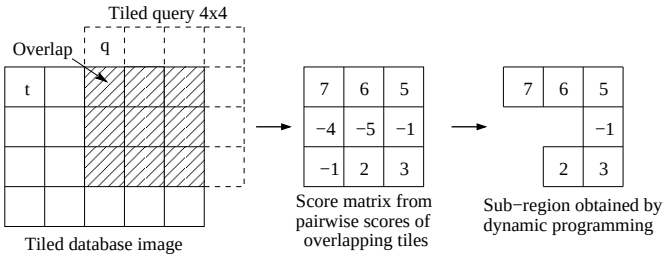


Figure 5. A 4×4 query is overlapped with a database image. For each tile in the 3×3 overlapped region, a score for the match is computed. DP is run on the score matrix to obtain the maximal scoring connected sub-region.

where λ and c are independent constants.

We can make the following observations from Eq. (5): (i) When distance to a database tile is kept invariant, a query tile with less background has a higher score, (ii) For a particular query tile, a more distant database tile has a lesser score.

We trained the parameters λ and c in order to fine tune the results according to the given image repository. The details are explained in Section 7.2.

An index structure built on the feature distances can be used because of the fact that the *ascending order of distances* for the database tiles is equivalent to the *descending order of scores* from a particular query tile. Thus, a nearest neighbor search readily returns the most scoring database tiles.

4. Finding Sub-Regions

In this section, we describe how the individual scores of matching a query tile against a database tile can be combined into a score of matching the entire query to a database region. As shown in Figure 5, overlapping the query region with a portion of the database image produces a score matrix of the pairwise aligned tiles where each score can be either positive or negative. We want to choose a *connected sub-region* that has the *maximal* possible cumulative score from this score matrix. Figure 5 illustrates that the maximal score may include negative scores and may not be rectangular in shape. We next show that the problem is NP-hard.

4.1. NP-completeness Proof

The problem "Find the maximal weighted connected subregion inside a matrix with positive and negative weights" is *NP-hard*. We prove this by showing that the corresponding decision problem in the graph equivalent of the matrix is *NP-complete*.

The graph problem is posed as follows: *Given a planar graph $G = (V, E)$ of degree at most 4, and with weight $w(v)$ on each vertex $v \in V$, is there a connected subgraph of weight $\geq W$?* We denote this problem by MAXIMAL WEIGHTED CONNECTED SUBGRAPH or MWCS.

Theorem 1. MWCS *is NP-complete, even for planar graphs of degree at most 4.*

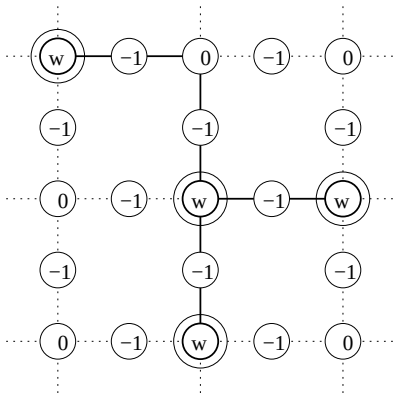


Figure 6. Construction from Rectilinear Steiner Tree instance to Maximal Weighted Connected Subgraph (MWCS) instance. The double lined vertices are the original terminal points and the bold lines represent a feasible solution to both the problems.

Proof. We first show that MWCS is in NP by allowing a nondeterministic Turing machine to guess a connected subgraph and check whether it has a weight of at least W.

We next reduce a known NP-complete problem RECTILINEAR STEINER TREE (RST) [12] to this problem. The RST problem asks: Given a set of n terminal points that are embedded in an integer grid in a plane, is there a spanning tree of total length l or less such that the vertices of the spanning tree are the points of the set and the grid points, and the length of an edge is the L_1 distance between the corresponding vertices?

Given an instance of the RST, we construct an instance of the MWCS as follows: We first find the bounding box of the points of the RST. Then, we replace each terminal point by a vertex of weight $w \gg l$. At each grid point that is not already occupied by the n terminal points, we place a vertex with weight 0. Between a pair of consecutive vertices on the same grid line (e.g., on the half-grid positions), we place a vertex with weight -1. Each vertex is connected to at most 4 of its neighbors along the directions of the grid, i.e., only to its horizontal and vertical neighbors. Figure 6 shows an example of the construction. The original points are shown by double circles. Clearly, the construction takes polynomial time and the graph G thus constructed is planar with degree at most 4.

We claim that the original RST on n points has a rectilinear Steiner tree of length $\leq l$ if and only if the MWCS graph has a connected subgraph of weight $\geq W = n.w - l$.

Only if: Assume that there is a Steiner tree of length at most l. By definition, it spans all the terminal points and is connected. Note that for a length l path between two points, there are exactly l vertices of weight -1. The vertices corresponding to the n terminal points has a weight of w each. Therefore, the weight of this tree is at least $n.w - l$. Figure 6 shows such a Steiner tree in solid lines.

If: Any connected subgraph of weight at least $n.w - l$ in G must include all the n vertices of weight w and at most l vertices of weight -1. There is no way to connect two vertices of weight ≥ 0 without passing through a vertex of weight -1. Therefore, the length of this subgraph is at most l, since other-

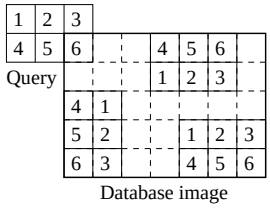


Figure 7. Some possible matching translations, rotations and reflections for a 2×3 query against a database image (shown in bold).

wise, the connected subgraph would have included more than l vertices of weight -1. Also, if the subgraph has the maximal weight, it is a tree, since, if it is not, at least one pair of vertices has more than one path between them. Removing that path increases the weight of the tree by the length of the path. Therefore, this subgraph defines a Steiner tree for the original n points. An example of such a subgraph is shown in Figure 6 in solid lines. □

4.2. Heuristic: Dynamic Programming

In this section, we design a simple dynamic programming (DP) heuristic as an alternate to examining all possible sub-regions for finding the maximal score. Assume that the score in tile (i, j) of the score matrix is denoted by $w(i, j)$ and the tile itself is denoted by $T(i, j)$. The DP is started from the lower left-hand corner tile and is continued by moving to right ($\rightarrow$) and up ($\uparrow$).

The maximal score of a sub-region ending on a tile $T(i, j)$ examines 4 possibilities: i) the score of the tile itself, ii) the score of the tile plus the maximal score for the bottom sub-region, iii) the score of tile plus the maximal score for the left sub-region, and iv) the score of the tile plus the maximal scores for the bottom and the left sub-regions. Since the bottom and the left sub-regions can intersect, the score of the intersecting region should be subtracted from the cumulative scores of the two sub-regions so that it is not counted twice.

Denoting the sub-region with the maximal score $s(i, j)$ ending at the tile $T(i, j)$ by $R(i, j)$, the DP algorithm computes the following recurrence relation:

$$s(i,j) = \max \begin{cases} w(i,j) \\ s(i,j-1) + w(i,j) \\ s(i-1,j) + w(i,j) \\ s(i,j-1) + s(i-1,j) + w(i,j) \\ \qquad -s(R(i,j-1) \cap R(i-1,j)) \end{cases} \quad (6)$$

The region maintained for the 4 cases are, respectively:

$$R(i,j) = \begin{cases} T(i,j) \\ R(i,j-1) \cup T(i,j) \\ R(i-1,j) \cup T(i,j) \\ R(i,j-1) \cup R(i-1,j) \cup T(i,j) \end{cases} \quad (7)$$

Each query can be matched with a database region in 8 possible ways (4 rotations and 2 reflections). Some of these possibilities are shown in Figure 7. The DP algorithm is run for all the 8 cases and the sub-region with the maximum score is considered as the match.

Running time: For a score matrix of size $m \times n$, the DP computes the maximal score for the sub-region ending at each cell. Calculating the scores for each cell requires finding an intersection of the largest scoring sub-regions on its bottom and left. This may require a running time of the order of $O(mn)$ in the worst case. Thus, the total running time of the DP algorithm is $O(m^2n^2)$. For a particular score matrix, the DP needs to be run from all the 4 corners in the following combinations of moves: (i) $\uparrow$ or $\rightarrow$, (ii) $\uparrow$ or $\leftarrow$, (iii) $\downarrow$ or $\rightarrow$, (iv) $\downarrow$ or $\leftarrow$. Thus, the worst case running time for the DP is *quadratic* in the size of the score matrix.

Shape: The DP algorithm cannot investigate all the possible connected sub-regions; it chooses the maximum scoring connected sub-region from only a certain class of shapes. We next analyze the class of such shapes. For a particular shape P, a tile t *sinks* another tile s, denoted by $t \triangleleft s$, if t can be reached from s in P by taking only a pre-defined set of moves. A tile t *sinks* a shape P, denoted by $t \triangleleft P$, if and only if for all tiles s belonging to P, t sinks s, i.e., $t \triangleleft P \iff \forall s \in P, t \triangleleft s$. A particular shape P can be captured by DP if and only if there exists a tile $t \in P$ that sinks P by either of the following four combinations of moves: (i) $\uparrow$ or $\rightarrow$, (ii) $\uparrow$ or $\leftarrow$, (iii) $\downarrow$ or $\rightarrow$, (iv) $\downarrow$ or $\leftarrow$. Examples of shapes that can be captured by DP are: $\sqcap, \square$. Shapes that cannot be captured include $+, \times$.

5. The QUIP Algorithm

This section describes the overall *QUIP* algorithm that includes the scoring scheme and the sub-region finding heuristic—how the top-k matches (in terms of cumulative scores) for a query pattern are retrieved from a database of images. For each query, there are many possible overlaps, each producing a score matrix on which DP needs to be run for extracting the maximal scoring connected sub-region.

If a database image has m tiles, there are m possible ways to place the query on the image. This takes into account all the possible translations. However, to take into account the orientation, the pattern has to be rotated at four angles—$0°$, $90°$, $180°$, and $270°$—and reflected for each such rotation (Figure 7). We do not handle rotations of arbitrary angles. Most images, including the ones that we experiment with (Section 7.1), are generally aligned to the x- and y-axes and hence these 4 angles are sufficient to capture the similar patterns. Thus, for a database of n images containing m tiles each, the cost of running the linear scan is of the order of $O(8mn) \times O(DP)$. Thus, the total *number of tiles* in the database is the most important parameter in finding the overlaps and not the number of images. In order to avoid the high cost of linear scan and make our algorithm scalable to large databases, we adopt the threshold algorithm (TA) [10] in *QUIP* to find the best scoring regions.

The *QUIP* algorithm (outlined in Figure 8) proceeds by maintaining a priority queue M that contains the k best matches found so far and a threshold score T that is the best possible score for a match not yet explored. Initially, M is empty and T is ∞. A bit vector B for all the database regions is also initialized with 0 to indicate that none of the regions have been explored.

QUIP Algorithm
Input: Q, query pattern; k, number of matches
Output: M, priority queue of the top-k matching regions
1. **for each** query tile Q_i
2. $\quad currtile[i]$:= First-NN of Q_i
3. **end for**
4. T := score of DP on score matrix of tiles in $currtile[i]$
5. $M := \phi$
6. **for each** query tile Q_i
7. $\quad N_i$:= next κ nearest neighbors of Q_i after $currtile[i]$
8. $\quad$ **for each** tile $N_{ij} \in N_i$
9. $\quad\quad$ retrieve N_{ij} from disk, if not already cached
10. $\quad\quad$ **for each** orientation
11. $\quad\quad\quad O$:= overlapping region (N_{ij})
12. $\quad\quad\quad$ **if** O has not been explored
13. $\quad\quad\quad\quad S$:= score of DP on O
14. $\quad\quad\quad\quad$ update M with S
15. $\quad\quad\quad\quad$ mark O as explored in bit vector B
16. $\quad\quad\quad$ **end if**
17. $\quad\quad$ **end for**
18. $\quad\quad currtile[i] := N_{ij}$
19. $\quad\quad$ update T
20. $\quad\quad$ **if** kth score in $M > T$
21. $\quad\quad\quad$ **return** M
22. $\quad\quad$ **end if**
23. $\quad$ **end for**
24. **end for**

Figure 8. The *QUIP* algorithm.

In every iteration of the algorithm, κ best neighbors from each of the query tiles are retrieved from the database by using either an index structure like the R-tree [13] or by sequential scan. Since many nearest neighbor searches may be required to be run for each query tile, we implemented incremental nearest neighbor search [15]. Once a similarity search for a query tile is run, it pays little overhead for subsequent nearest neighbor searches. We maintain $\kappa = 2000$.

For each retrieved database tile, henceforth called "seed", we retrieve its corresponding image from the memory cache or disk in case of cache miss. The seed and the position of the corresponding query tile in the query region are used to complete the overlapping region in the image in each of the 8 orientations. Each overlapped region produces a score matrix as shown in Figure 5 on which DP is run to obtain a maximal scoring connected sub-region. The sub-region is inserted in M if its score is within the top k scores found so far. Thus, the k^{th} best score in M monotonically increases. The corresponding bit for the region in B is set to 1 so that DP is not run more than once on it.

The current score for each query tile forms an *upper bound* on the scores of database tiles yet to be explored. The threshold score T is thus maintained as the maximal score found by DP on the score matrix formed by such scores. T thus stands for the best possible cumulative score for a region not yet accessed. As the algorithm proceeds, seeds with smaller scores are accessed leading to a decrease in T. The algorithm stops when the k^{th} score in M exceeds T.

6. Significance Computation

In order to assist the domain scientists in understanding whether the query pattern is rare or matches frequently in the database, we can compute the statistical significance or *p-value* of each result retrieved by *QUIP*.

The p-value of score s of a result against a query is defined as the probability of randomly obtaining a match with score s or higher from the database for the same query. P-value of score s can be computed by first scoring every database region with the query and then computing the probability of scores greater than s from the distribution. Mathematically, p-value is calculated as the area under the score distribution greater than s, i.e., $pvalue = 1 - cdf(s)$ where cdf is the cumulative distribution of the scores of the database regions. The lower the p-value, the more significant is the match.

Clearly, it is impractical to compute the p-value in the naïve way by scoring all the database regions at run time. We next describe an approximation technique to compute p-values efficiently. There are three ideas in our algorithm: (i) use *histograms* to approximate score distributions from each query tile, (ii) *cascade* convolution of query tile histograms to obtain the score histogram of the entire query, and (iii) use *bounds* to convolute histograms.

For every database tile, the score distribution of matching it to all database tiles is pre-computed and maintained as a histogram. Since the top results are likely to include more positive scores, greater details are maintained for positive scores (bins are in intervals of 10) than negative scores (bins in intervals of 10,000). The score distribution of a query tile is approximated by the score histogram of its closest database tile.

The score histogram of an entire query region is the convolution of the histograms of the individual query tiles. For r tiles with b bins each, the convolution requires b^r operations. To avoid such exponential costs, we convolute the histograms in a cascading fashion. Initially, the histograms of the first two tiles are convoluted to yield another score histogram that is binned in the same manner—intervals of 10 for positive scores and 10,000 for negative scores. Then, this histogram is convoluted with the next histogram and so on till all the r histograms have been convoluted. Denoting the convolution of histograms up to i tiles by ρ_i and the i^{th} histogram by q_i, we compute $\rho_i = \rho_{i-1} \oplus q_i$ up to $i = r$. Each histogram convolution requires *quadratic* number of operations in terms of the number of bins in the histograms. To make it more efficient, we applied the following bounding procedure.

The bounding method aims to collapse multiple bins into a single bin. This is achieved by computing score bounds in each histogram. The scores below this bound cannot contribute to the score s of the match whose p-value is being computed. Hence, the details of the bins that are completely below this bound need not be maintained. They are combined into a single bin. This reduces the number of bins and speeds up the histogram convolutions.

Figure 9 shows an example. Assume that q_{i+1} is the last histogram, i.e., $i + 1 = r$. If $s = 100$ and the maximum score in the histogram of q_{i+1} is 40, then any score less than 60 in ρ_i cannot add up to 100. Thus, all the bins in ρ_i that contains

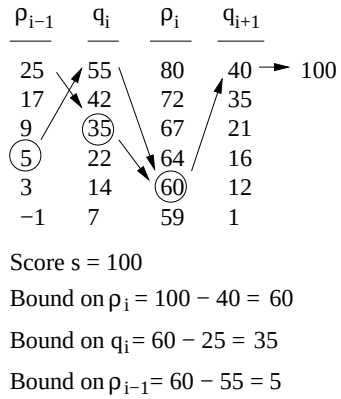


Figure 9. Efficient convolution of histograms. The bins below the highlighted bounds in each histogram can be combined together.

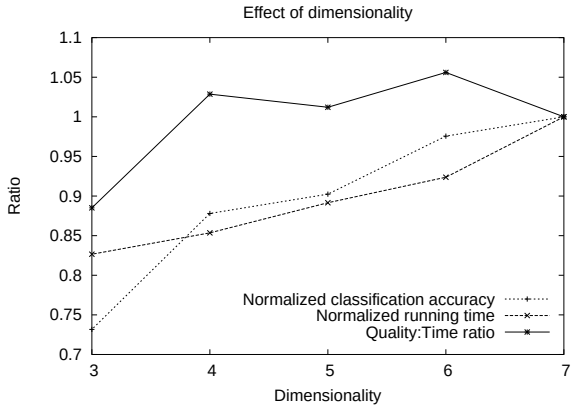


Figure 10. Effect of dimensionality. Dimensionality of 6 has the best quality-time trade-off.

scores below 60 can be combined together. In Figure 9, the bins in ρ_i with upper boundaries 59 and smaller are combined together. The bounds can be computed backwards for q_i and ρ_{i-1} as well. The maximum in ρ_{i-1} is 25. Since, (details of) scores of only 60 and above are required in ρ_i, any score less than 35 in q_i are not required. Thus, as shown in the figure, the bins with upper boundaries 55, 42, and 35 are left as they are. All the three other bins can be collapsed reducing the number of bins from 6 to 3. Similarly, the last two bins in ρ_{i-1} that are below the bound 5 can be combined together. Mathematically, these bounds can be calculated by:

$$B(\rho_{i-1}) = s - \sum_{j=i}^{r} \big(\max(q_j)\big) \tag{8}$$

$$B(q_i) = s - \sum_{j=i}^{r} \big(\max(q_j)\big) - \max(\rho_{i-1}) \tag{9}$$

7. Experimental Results

In this section, we present the results: both qualitative and quantitative. Due to the usefulness of micrographs in biology and their availability, we conducted our experiments with 3 datasets of confocal microscopy images of cross-sections of feline retina. We also synthetically generated a bigger dataset.

7.1. Datasets

The first retinal dataset, PA, consisted of 80 images (30,601 tiles) labeled with the lectin peanut-agglutinin (PNA). The second dataset, NF, consisted of 37 images (31,343 tiles) labeled with the anti-neurofilament (anti-NF) antibody. The third dataset, GR, consisted of 121 images (160,378 tiles) labeled with anti-glial fibrillary acidic protein (anti-GFAP) and anti-rhodopsin antibodies. The details of the imaging and the antibodies can be found in Appendix A and in [11].

Other than these real biological datasets, a synthetic dataset was also generated to test the scalability of *QUIP* with database and query sizes. The base of the dataset was 61,944 tiles of the PA and the NF datasets combined, denoted by PANF (117 images). Perturbations were applied both at image level and at tile level. The image level changes were: (i) Intensity of each pixel was changed randomly, (ii) Based on a random number generated, either the upper half or the lower half of the image was made background, and (iii) Based on a random number generated, the image was reflected either horizontally or vertically along the mid-axis. The tile level changes were: (i) Intensity of 100 pixels in a tile were changed randomly, (ii) Pixels were distributed randomly inside a tile, and (iii) Each pixel intensity was modified to the average intensity of its 4 neighbors. In this way, a dataset of 805,272 tiles (1,521 images) was created.

We emphasize the point that the number of potential matches for a query is equal to the number of tiles in the database times the number of rotations and reflections times the number of sub-regions within each candidate region. Thus, the *number of regions* a query needs to handle is *equal* to the *number of tiles*—805,272.

7.2. Parameter Selection

We first describe the choice of different parameters used in our experiments: the parameters for the scoring scheme and the reduced dimensionality of the datasets.

The parameters λ and c for the scoring scheme in Eq. (5) described in Section 3 required manual training in order to fine tune the score according to the nature of the image repository. The parameters were chosen based on a classification scheme similar to that used in the Walrus method [24]. For each queried pattern, we tagged a database region as either a *true* match or a *false* match. The classification accuracy was computed as the ratio of the number of true matches to the total number of matches. The entire database was split into two parts in a stratified cross-validation manner. We experimented with different values of λ and c and evaluated the top-5 results from each query.

Since the process of identifying the ground truth is manual (therefore, subjective) and also very labor-intensive, we trained our parameters for the PA dataset on 10 queries. The dimensionality of the dataset was reduced from 256 to $d = 7$ by PCA [17] which retained 99.99% of the energy. The highest

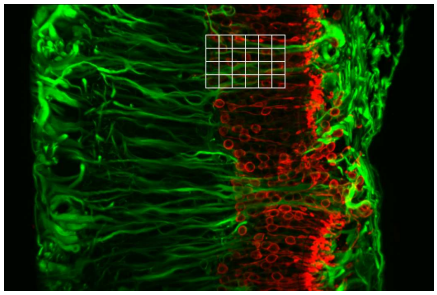


Figure 11. A query from the GR dataset.

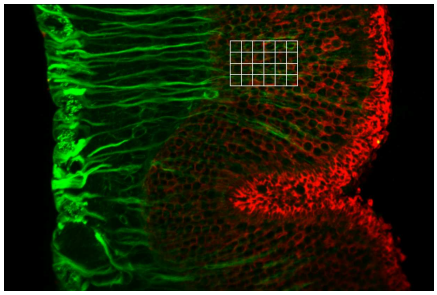


Figure 12. Top result from the GR dataset.

classification accuracy (82.0%) was achieved with $\lambda = 1$ and $c = 23000$. With these parameters, 8 other queries were classified over the entire PA dataset. An accuracy of 78.6% was achieved. The same values of $d = 7$ (99.99% energy), $\lambda = 1$ and $c = 23000$ obtained good results (86.7% for 3 queries) for the NF dataset. The GR dataset contained color information in both red and green channels and thus were more varied; 90% of the energy was retained using $d = 12$ dimensions. The classification accuracy of the results with the same values of λ and c was 82.4% for 3 queries. All the 14 queries except two returned at least 3 true matches out of the top-5. For all of them, the first match was a true match.

The next set of experiments was performed on the reduced dimensionality d. Figure 10 shows the effect of dimensionality on the classification accuracy and the running time for the PA dataset. The parameters for the scoring scheme did not have appreciable effect on the dimensionality and were thus maintained across the different dimensions. Since $d = 7$ retained almost all the energy (99.99%), higher dimensional features were not explored. The classification accuracy and the running time have been normalized with respect to their values for $d = 7$. Higher dimensional feature values retrieved better quality results, but took a longer time to finish. To determine the best quality-time trade-off, we calculated the ratio of normalized classification accuracy to normalized running time. Even though $d = 7$ achieved the best quality, the gain in quality over $d = 6$ was not much when compared to the amount of extra time it took to complete. Dimensionality of 6 had the best quality-time trade-off and was subsequently used in all our experiments.

7.3. Biological Significance of Retrieved Results

The next set of experiments assess the performance of our method in the context of retinal biology. Figure 1(c) shows the best result for the query in Figure 1(a) from the PA dataset. Both capture folds in the outer segment layer in retinas that have been detached for 7 days. Figure 11 shows a query from the GR dataset that captures new growth of glial cells into the outer nuclear layer and a corresponding migration of rhodopsin from the rod and cone layer into the outer nuclear layer during retinal detachment. The result shown in Figure 12 has found a similar simultaneous migration of both GFAP and rhodopsin. Appendix B shows an example query and the top result from the NF dataset. All these results show that *QUIP* is able to find matches that are semantically useful.

PA	NF	GR
4.8×10^{-16}	2.6×10^{-14}	4.3×10^{-39}
1.6×10^{-15}	3.9×10^{-14}	1.8×10^{-38}
1.4×10^{-14}	4.5×10^{-14}	1.5×10^{-32}
1.6×10^{-11}	2.1×10^{-13}	9.4×10^{-32}
3.0×10^{-11}	1.0×10^{-12}	1.1×10^{-31}

Table 1. P-values of the top-5 results.

7.4. Statistical Significance of Retrieved Results

We next report the p-values of the top-5 matches for each of the queries described in Section 7.3. Table 1 shows that the results were significant with low p-values of the order of 10^{-11} or less. In general, the results for the GR dataset were much more significant than the other two datasets due to its bigger size. This shows that *QUIP* returns true and significant matches for patterns of biological interest.

7.5. Running Time and Scalability

In this section, we first report the running times of returning top-5 results for queries of size 25 (for 32×32 pixel tiles, this translates to 25,600 pixels) on the real datasets. It took 5.5s and 5.6s to finish for the PA and the NF datasets respectively. Computing the p-values added an overhead of 0.2s or 4% time. The queries on the bigger and higher dimensional GR dataset finished in 33.8s with an additional overhead of 1.1s or 3% of the time for p-value computation.

To ascertain the scalability of *QUIP* with respect to large databases and queries, we performed two scalability experiments, one with database size and the other with query size.

Since the number of regions (equivalently, tiles) in the database is the most important parameter in testing the scalability with database size, we created different smaller sized datasets in multiples of 200,000 tiles from the synthetic dataset. We experimented with both bulk-loaded R-trees [37, 19] and sequential scan for nearest neighbor searches. Due to the large number of nearest neighbors required to lower the threshold score below the k^{th} best score and to halt the algorithm (Section 5), using R-trees was costly. The datasets were all memory-resident and, therefore, sequential scan was faster. The times reported here are with sequential scan. In future, we plan to experiment with even bigger disk-resident datasets. We also envision applying some early stopping criteria with probabilistic guarantees by characterizing the rate of decrease of threshold score and the rate of increase of the k^{th} best score.

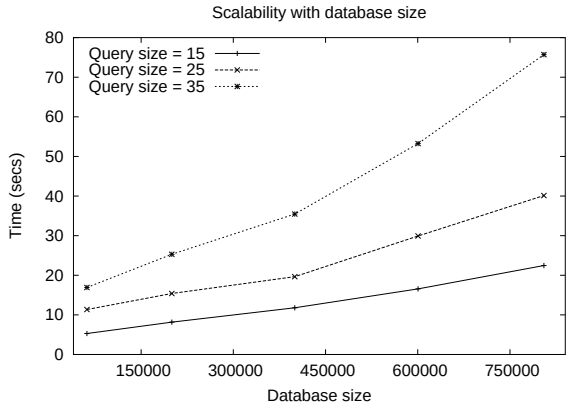


Figure 13. Scalability with database size.

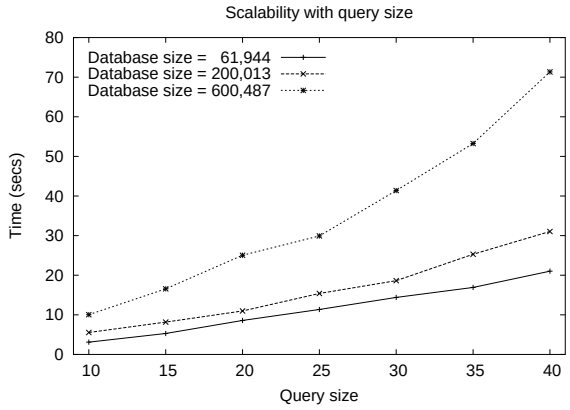


Figure 14. Scalability with query size.

Figure 13 shows the running time for these datasets and the real dataset PANF (of size 61,944) for 3 different query sizes. Even for a huge database of 805,272 regions and a query of size 35, *QUIP* finishes in less than 76s. All other queries finished in less than a minute. P-value computation takes less than a second to complete. For larger datasets, the scalability becomes linear due to the sequential scans.

Next, we report the effect of increasing query sizes on *QUIP* (Figure 14). Queries of different sizes ranging from small (10) to huge (40) were run for PANF and two other synthetic datasets. For more query tiles, more nearest neighbor retrievals were required. Further, for larger sized regions, the cost of running the DP for finding sub-regions were increased. Even then, queries of size 40 finished in just over a minute. For real datasets, even the largest queries finished in less than 22s. This shows that *QUIP* is practical.

7.6. Effect of Threshold Algorithm

To understand the effect of running the modified threshold algorithm in *QUIP* instead of just a simple database scan, we measured the number of "seeds" (or database regions) explored by *QUIP* on the different synthetic databases. A linear scan always explores all the regions in the database. Figure 15 shows that using the threshold algorithm prunes large parts of the database. For small to medium query sizes (up to 25), the pruning is over 80%. Even for large query sizes (up to 35),

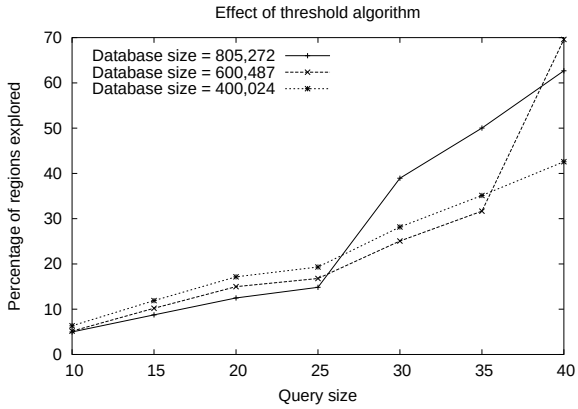


Figure 15. Effect of threshold algorithm.

more than half the database regions are pruned, and the associated DP costs are saved.

8. Background and Related Work

The retina is the part of the eye that contains neurons that respond to light and transmit electrical signals to the brain via the optic nerve. Multiple molecular probes such as lectins and antibodies are used to examine the localization of specific protein expression in retinal cells and the expression patterns of these proteins in the different layers of the retina. The fluorescently tagged probes are visualized by immunohistochemistry using a confocal microscope. Multiple proteins can be visualized in a single image, with each protein represented by a different color channel. Images are taken from retinas that are normal or have been detached for 1, 3, 7, 28 days or re-attached for 3, 6, 28 days after detachment [11].

An excellent survey on the recent methods of content-based image retrieval (CBIR) and region-based image retrieval (RBIR) methods developed by the image processing community can be found in [9]. Some older surveys are in [27, 29]. A semantics-sensitive image retrieval approach was proposed in [34]. Semantic categories, region segmentation with wavelet features and integrated region matching [20] were used as part of it. Wavelet features and region matching with maximum area overlap was used by [24]. Many additional systems for CBIR and/or RBIR have been developed [2, 5, 6, 18, 22, 33, 35, 36]. RBIR has also been extended to incorporate fuzzy logic [7], perceptual grouping rules [16], statistical tests [30], vantage points [25], fractal parameters [26], salient points [31], etc. Most of these methods use automatic or manual region segmentation in order to characterize regions and then use one-to-one or many-to-one mapping to match regions. None of them discern between foreground and background. Dagli et al. [8] partitioned an image into small (4×4) non-overlapping tiles and used local low-level features to get rid of irrelevant background. The query region was segmented into foreground and background and only the foreground region was searched in the database. One important work in medical imagery was by Shyu et al. [28] in the domain of high-resolution computed tomography of the lung. They used expert-annotated perceptual categories and applied various op-

erators to detect presence or absence of these categories in an image. The discriminatory power of the categories were tested by statistical tests. Key patterns (keywords) in a dataset was discovered using a visual vocabulary in [3]. A texture thesaurus was used for pattern annotation in [21].

9. Conclusions

In this paper, we developed a novel method *QUIP* to address the problem of querying significant patterns from a image database. The advancement of high-throughput and high-quality image acquisition techniques coupled with the ability of images to reveal spatio-temporal information not readily available from other data sources makes this an important mechanism of gaining scientific knowledge.

A scoring scheme was developed to capture the relevant pattern information from a tiled image region. Each tile was adjudged a background value and the smaller this value, the higher it scored with a similar tile. A query pattern was specified as a rectangular region of tiles. We proved that the problem of finding the maximal weighted connected subgraph from a vertex-weighted graph of both positive and negative weights is NP-hard, even for planar graphs of degree at most 4. An effective dynamic programming heuristic was proposed to solve it. We adopted the threshold algorithm to efficiently find the overlapping candidate database regions. For each result, we developed an algorithm to calculate the statistical significance of the match.

We showed that *QUIP* was able to retrieve patterns that are scientifically meaningful and important. Finally, we illustrated that our method has practical running times and scales with respect to database and query sizes.

References

[1] GraphicsMagick. http://www.graphicsmagick.org/.

[2] S. Ardizzoni, I. Bartolini, and M. Patella. Windsurf: Region-Based Image Retrieval Using Wavelets. In *DEXA Work.*, pages 167–173, 1999.

[3] A. Bhattacharya, V. Ljosa, J.-Y. Pan, H. Yang, M. R. Verardo, C. Faloutsos, and A. K. Singh. ViVo: Visual Vocabulary Construction for Mining Biomedical Images. In *ICDM*, pages 50–57, 2005.

[4] M. V. Boland, M. K. Markey, and R. F. Murphy. Automated Recognition of Patterns Characteristic of Subcellular Structures in Fluorescence Microscopy Images. *Cytometry*, 3(33):366–375, 1998.

[5] C. Carson, S. Belongie, H. Greenspan, and J. Malik. Blobworld: Image Segmentation Using Expectation-Maximization and Its Application to Image Querying. *PAMI*, 24(8):1026–1038, 2002.

[6] S. Chandran and N. Kiran. Image Retrieval With Embedded Region Relationships. In *ACM Symp. Appl. Comp. (SAC)*, pages 760–764, 2003.

[7] Y. Chen and J. Z. Wang. A Region-Based Fuzzy Feature Matching Approach to Content-Based Image Retrieval. *PAMI*, 24(9):1252–1267, 2002.

[8] C. Dagli and T. S. Huang. A Framework for Grid-Based Image Retrieval. In *ICPR*, pages 1021–1024, 2004.

[9] R. Datta, J. Li, and J. Z. Wang. Content-Based Image Retrieval: Approaches and Trends of the New Age. In *Work. Mult. Inf. Retr. (MIR)*, pages 253–262, 2005.

[10] R. Fagin, A. Lotem, and M. Naor. Optimal Aggregation Algorithms for Middleware. In *PODS*, pages 102–113, 2001.

[11] S. K. Fisher, G. P. Lewis, K. A. Linberg, and M. R. Verardo. Cellular Remodeling in Mammalian Retina: Results from Studies of Experimental Retinal Detachment. *Prog. Retinal Eye Res.*, 24(3):395–431, 2005.

[12] M. R. Garey and D. S. Johnson. The Rectilinear Steiner Tree Problem is NP-Complete. *SIAM J. Appl. Math.*, 32(4):826–834, 1977.

[13] A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In *SIGMOD*, pages 47–57, 1984.

[14] L. X. He. *Pattern Recognition and Image Processing of Infrared Astronomical Satellite Images*. PhD thesis, Iowa State University, 1996.

[15] G. R. Hjaltason and H. Samet. Distance Browsing in Spatial Databases. *ACM Trans. Database Syst.*, 24(2):265–318, 1999.

[16] Q. Iqbal and J. K. Aggarwal. Retrieval by Classification of Images Containing Large Manmade Objects Using Perceptual Grouping. *Patt. Recog.*, 35(7):1463–1479, 2002.

[17] I. T. Jolliffe. *Principal Component Analysis*. Springer, 2002.

[18] C.-R. Kim and C.-W. Chung. XMage: An Image Retrieval Method Based on Partial Similarity. *Inf. Proc. Mgmt.*, 42(2):484–502, 2006.

[19] S. T. Leutenegger, J. M. Edgington, and M. A. Lopez. STR: A simple and efficient algorithm for R-tree packing. Technical report, Institute for Computer Applications in Science and Engineering (ICASE), 1997.

[20] J. Li, J. Z. Wang, and G. Wiederhold. IRM: Integrated Region Matching for Image Retrieval. In *ACM Mult.*, pages 147–156, 2000.

[21] W.-Y. Ma and B. S. Manjunath. A Texture Thesaurus for Browsing Large Aerial Photographs. *J. Am. Soc. Inf. Sc.*, 49(7):633–648, 1998.

[22] W.-Y. Ma and B. S. Manjunath. NeTra: A Toolbox for Navigating Large Image Databases. *Mult. Sys.*, 7(3):184–198, 1999.

[23] B. S. Manjunath, P. Salembier, and T. Sikora. *Introduction to MPEG-7: Multimedia Content Description Interface*. Wiley, 2002.

[24] A. Natsev, R. Rastogi, and K. Shim. WALRUS: A Similarity Retrieval Algorithm for Image Databases. *Know. Data Engg.*, 16(3):301–316, 2004.

[25] A. Natsev and J. R. Smith. A Study of Image Retrieval by Anchoring. In *IEEE Int. Conf. Mult. & Expo*, pages 421–424, 2002.

[26] M. Pi, M. K. Mandal, and A. Basu. Image Retrieval Based on Histogram of Fractal Parameters. *IEEE Trans. Mult.*, 7(4):597–605, 2005.

[27] Y. Rui, T. S. Huang, and S.-F. Chang. Image Retrieval: Current Techniques, Promising Directions, and Open Issues. *J. Vis. Comm. Img. Repr.*, 10(1):39–62, 1999.

[28] C.-R. Shyu, C. Pavlopoulou, A. C. Kak, C. E. Brodley, and L. S. Broderick. Using Human Perceptual Categories for Content-Based Retrieval from a Medical Image Database. *Comp. Vis. Img. Underst.*, 88(3):119–151, 2002.

[29] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-Based Image Retrieval at the End of the Early Years. *PAMI*, 22(12):1349–1380, 2000.

[30] C. Theoharatos, N. A. Laskaris, G. Economou, and S. Fotopoulos. A Generic Scheme for Color Image Retrieval Based on the Multivariate Wald-Wolfowitz Test. *TKDE*, 17(6):808–819, 2005.

[31] Q. Tian, N. Sebe, M. S. Lew, E. Loupias, and T. S. Huang. Image Retrieval using Wavelet-Based Salient Points. *J. Electr. Img.*, 10(4):835–849, 2001.

[32] M. A. Turk and A. P. Pentland. Eigenfaces for Recognition. *Journal of Cognitive Neuroscience*, 3(1):71–96, 1991.

[33] J. Z. Wang and Y. Du. Scalable Integrated Region-based Image Retrieval using IRM and Statistical Clustering. In *Proc. Jt. Conf. Dig. Lib.*, pages 268–277, 2001.

[34] J. Z. Wang, J. Li, and G. Wiederhold. SIMPLIcity: Semantics-Sensitive Integrated Matching for Picture LIbraries. *PAMI*, 23(9):947–963, 2001.

[35] J. Z. Wang, G. Wiederhold, O. Firschein, and S. X. Wei. Wavelet-Based Image Indexing Techniques with Partial Sketch Retrieval Capability. In *ADL*, pages 13–24, 1997.

[36] R. Weber and M. Milvoncic. Efficient Region-Based Image Retrieval. In *CIKM*, pages 69–76, 2003.

[37] D. A. White and R. Jain. Similarity Indexing: Algorithms and Performance. In *SPIE Storage and Retrieval for Image and Video Databases*, pages 62–73, 1996.

Appendix

A. Background on Retinal Images

Peanut-agglutinin (PNA) is a lectin composed of four identical subunits. In the retina, it is found within the extracellular matrix between the photoreceptor outer segments and the retinal pigmented epithelial (RPE) cells. Photoreceptor cells are responsible for the detection and harvesting of light. During retinal detachment, the photoreceptors are detached from the underlying RPE cells. This detachment causes a redistribution of the extracellular matrix surrounding the photoreceptor cells. When the retina becomes reattached, there is further rearrangement of the extracellular matrix surrounding the photoreceptors.

Neurofilament (NF) is an intermediate filament protein that is expressed highly in the axons of neurons. During retinal detachment, there is a significant rearrangement of the neural connections in the retina which often results in loss or compromised function of the photoreceptor signaling pathways via the optic nerve to the brain.

Glial fibrillary acidic protein (GFAP) is an intermediate filament protein expressed in glial cells of retina including the Müller cells. These cells support the functions of nerve cells. When the retina is injured, glial cells react by rapidly producing more GFAP. During retinal detachment, GFAP's expression becomes up-regulated.

Rhodopsin is expressed in photoreceptor cells of the retina. In the normal retina, rhodopsin is localized exclusively to the photoreceptor outer segments. During retinal detachment, the photoreceptor outer segment rapidly degenerates. For reasons unclear to biologists, this causes a redistribution of rhodopsin to other portions of the photoreceptors.

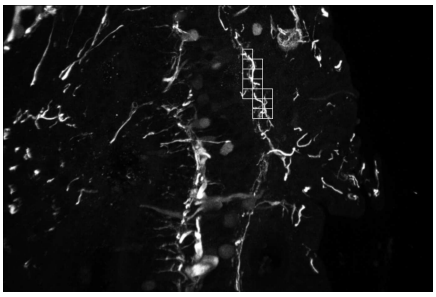


Figure 16. A query from the NF dataset.

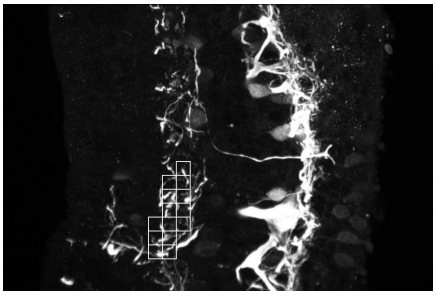


Figure 17. Top result from the NF dataset.

B. More Biological Results

Figures 16 and 17 show a query from the NF dataset and the first result, respectively. The query captures the growth of several axons. The result also centers on a region of new axon growth demonstrating a true result.