

Geometric Algorithms for the Analysis of 2D–Electrophoresis Gels

Alon Efrat* Frank Hoffmann† Klaus Kriegel‡§ Christof Schultz†§ Carola Wenk†¶

ABSTRACT

In proteomics 2-dimensional gel electrophoresis (2-DE) is a separation technique for proteins. The resulting protein spots can be identified by either using picking robots and subsequent mass spectrometry or by visual cross inspection of a new gel image with an already analyzed master gel. Difficulties especially arise from inherent noise and irregular geometric distortions in 2-DE images. Aiming at the automated analysis of large series of 2-DE images, or at the even more difficult interlaboratory gel comparisons, the bottleneck is to solve the two most basic algorithmic problems with high quality: Identifying protein spots and computing a matching between two images. For the development of the analysis software CAROL at Freie Universität Berlin we have reconsidered these two problems and obtained new solutions which rely on methods from computational geometry. Their novelties are: 1. Spot detection is also possible for complex regions formed by several “merged” (usually saturated) spots; 2. User-defined landmarks are not necessary for the matching. Furthermore, images for comparison are allowed to represent different parts of the entire protein pattern, which only partially “overlap”. The implementation is done in a client server architecture to allow queries via the Internet. We also discuss and point at related theoretical questions in computational geometry.

1. INTRODUCTION

The proteomic research deals with the systematic analysis of complete profiles of the proteins expressed in a given cell, tissue or biological system at a given time. In this field, 2D-gel electrophoresis

*Computer Science Department, University of Arizona, email: alon@CS.Arizona.EDU

†Institut für Informatik, Freie Universität Berlin, Takustr. 9, D-14195 Berlin, email: name@inf.fu-berlin.de

‡Deutsches Herzzentrum Berlin, Augustenburger Platz 1, D-13353 Berlin

§Supported by Deutsche Forschungsgemeinschaft, grant FL 165/4–2.

¶Supported by Deutsche Forschungsgemeinschaft, grant AL 253/4–3.

(2-DE) is a well-established and widely used technique to separate proteins in a sample. A 2-DE gel is the product of two sequentially performed separations in acrylamide gel media: isoelectric focusing as first dimension and a separation by molecular weight as second dimension. The result of that process is a 2D pattern of spots each representing a protein. There is a variety of staining methods to display protein spots: Coomassie blue and silver-staining and / or fluorescent and radioactive labeling. Mass spectrometry and the visual (usually computer aided) comparison (matching) with already analyzed gel images of such a sample are used to identify proteins. The visual analysis of such 2-DE image series intends to identify those proteins that change their expression (size, intensity) and reflect/cause certain biochemical and biomedical conditions of an organism, see [24]. However, this requires high throughput analysis tools and the major challenge is to obtain both robust and reliable algorithmic solutions that work automatically, or at least need only little user interaction.

A second strand of research was initiated in [20]. Here the challenge consists of creating analysis software that is able to perform interlaboratory gel image comparisons and to deal with images from various databases via the Internet.

The intensive research on these questions emphasizes the strong demand from the practitioners for better algorithmic solutions.

The aim of this paper is to demonstrate how ideas from computational geometry help to meet these challenges in the context of the two most basic problems in gel analysis:

- Given a scanned 2-DE gel image identify spot regions.
- Given two images by their spot lists, perform a matching procedure to identify those spot pairs that correspond to each other, so that the matching reflects both geometric and spot intensity resemblance of the images.

In this paper we survey algorithmic studies and implementation work that was done while developing the 2-DE analysis software system CAROL ([8]). It contains both new results (Section 2) and partly published ones (Section 3, [15]).

In Section 2 we address the spot detection question. We assume that each spot has approximately the shape of an axis-parallel ellipse. This is a widely accepted modeling assumption, see for example [5] or [12]. However, spots that are very close to each other may partially overlap and “merge”. This leads to rather complicated local pixel regions, as depicted in Figure 1, compare [18].

In [21] a spot detection algorithm is described that relies on a watershed transformation applied to the gradient image. The most difficult part is the interpretation of twin spots, streaks (left side in Fig.1), and so-called *complex regions* (right side in Fig.1) as unions of ellipses.

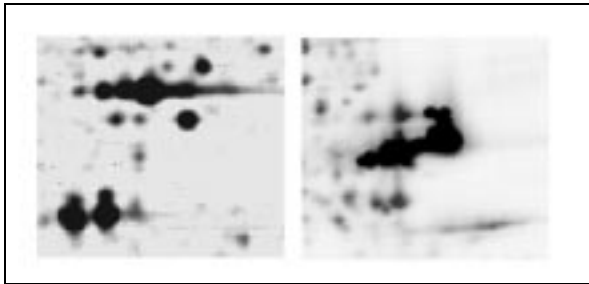


Figure 1: Twin spots, streaks, and a complex region

In fact, in [21] the latter case was left open and in the implementation the user had to edit these complex regions by hand. Here we present algorithmic solutions to this question, one brute force solution and a second based on a Linear Programming formulation. To the best of our knowledge it is the first algorithmic attempt to deal with spot detection in complex regions. There is an inherent uncertainty in the images due to the electrophoresis process itself which is highly susceptible to faults and geometric distortions. The reader may argue just by looking at the images, that there is no hope to come up with a perfect spot detection algorithm. This is certainly true. Yet we can cope with this ambiguity, at least to some extent by computing a ranked list of proposals how a complex region could be covered instead of computing only the “best” covering, which could be erroneous. Then, in the matching algorithm, we have the possibility to accept the matching of two ambiguous regions if there is a pair of proposed coverings that match.

The matching algorithm itself is described in Section 3. Assume that a source and a target image are given by their spot lists. We first transform the lists into geometric point patterns, where a spot is represented by the ellipse center coordinates and additionally we code the real size/intensity of the spot into a single integer value. The lists can be of significantly different size and, additionally, we allow the images to show only partially overlapping sections of the gels. Thus the task is to find a maximal-cardinality one-to-one matching of the spot lists, so that this mapping respects both the geometry of the images and the relative spot intensities i.e., relatively intensive source spots should be mapped to relatively intensive target spots. The geometric matching criterion is based on similarity of imaginary edges which link spots. Two edges are similar if their length ratio and the difference of their angles formed with the x -axis are within preset tolerance bounds. Our algorithmic solution, which we call global-via-local matching, works in two stages. Firstly, we compute matchings for several small local patterns, which are chosen in a grid-like fashion. For this local matching we use a modified alignment method known from point pattern matching ([2]) combined with geometric hashing. A significant speed-up is gained by reducing the set of all source / target edges to those belonging to the history of the incremental Delaunay triangulation of the spot lists in the order of their decreasing intensities. Having sets of matching candidates for each of the local patterns, we select a subset that is consistent with the overall grid

topology. The matching spot pairs computed this way later serve as “landmarks” in the second stage. Landmarks were also used in previous algorithmic solutions, but they had to be set manually. The novelty of our solution is to generate them automatically. The subsequent extension from landmarks to a maximal global matching is rather standard by local neighborhood comparisons. However, our advantage is the possibility to exploit the proposal lists for ambiguous spot regions, as computed in the spot detection. This way, we try to simulate how an expert would compare images by visual inspection, namely, by intertwining the spot detection with the matching process.

Figure 2 shows two 2-DE gel images originated from a transfection experiment of cultured EaHy cells with 800 (left) and 1000 protein spots. Their original size is about 23cm by 29cm. For the purpose of illustration in Figure 3 and 4 more details are shown of the small rectangular window regions marked in Figure 2. (For simplicity ellipses were replaced by circles.)

In both Sections 2 and 3 we also refer to related theoretical issues and open problems in computational geometry that could further improve our solution. In Section 4 we outline implementation issues and describe experimental results.

2. SPOT DETECTION AS AN ELLIPSE COVERING PROBLEM

2.1 Modeling of the Core Problem

Spot detection is to a wide extent an image processing problem. In fact, if the spots are well separated, classical methods like filtering and watershed transformation on the gradient image provide a satisfying algorithmic solution, see [21]. As mentioned above, the core combinatorial problem is the interpretation of complex pixel regions as unions of axis parallel ellipses.

The following formalization is a compromise stemming from discussions with practitioners who solve these covering instances by peer review. First, we remark that complex regions are typically saturated (black), and therefore gray level information does not help for the detection. We assume that a connected pixel pattern R is given, which is fat in the sense that there are neither short horizontal nor vertical cuts consisting of only two pixels. In the application R is usually a simply connected subpattern of a 100×100 -pixel square. We identify a pixel with its center and denote by ∂R the rectilinear polygonal curve traversing the boundary pixels. We shrink and expand this boundary to the inside and the outside by defining the two pixel sets $\partial R_- = \{p \in R | d_\infty(p, \partial R) = 1\}$ and, analogously, $\partial R_+ = \{p \notin R | d_\infty(p, \partial R) = 1\}$, where d_∞ is the distance function of the max norm.

Now we can formulate the approximative covering problem from the application point of view. For numbers $0 \leq p, q \leq 1$ and a natural number $\rho \geq 1$, the problem is to find a smallest-cardinality set \mathcal{E} of axis-parallel ellipses fulfilling the following conditions.

1. **(Shape)** For each $E \in \mathcal{E}$ the ratio of its halfaxes is in the interval $[1/\rho, \rho]$.
2. **(Fitting)** Each ellipse $E \in \mathcal{E}$ respects ∂R_+ , i.e., it does not intersect ∂R_+ .
3. **(Intersection)** The boundaries of every pair of ellipses $E, E' \in \mathcal{E}$ intersects in at most 2 points, and $\text{area}(E \cap E') \leq q \cdot \min\{\text{area}(E), \text{area}(E')\}$.

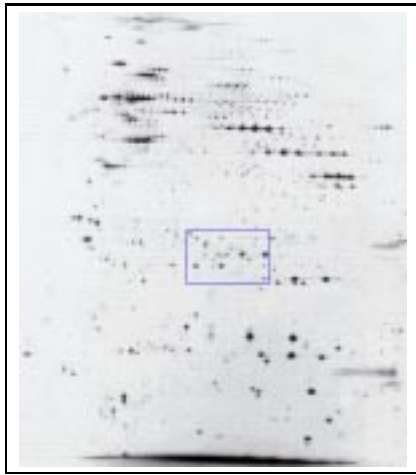


Figure 2: Two gel images of cultured EaHy cells from a transfection experiment

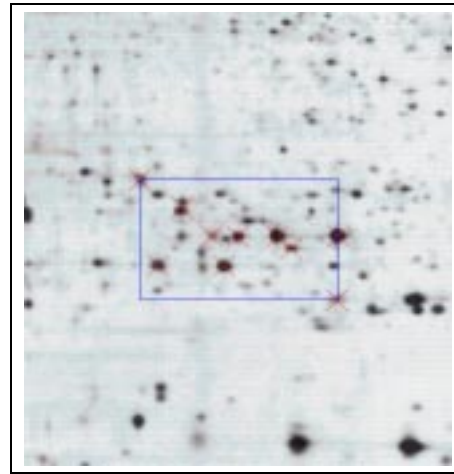
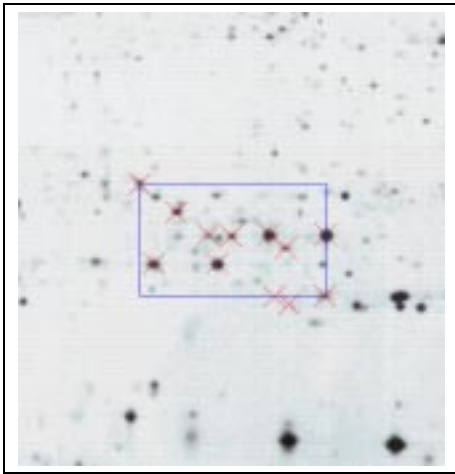


Figure 3: Detailed local images of Fig.2, a selected pattern on the left side and a partial matching

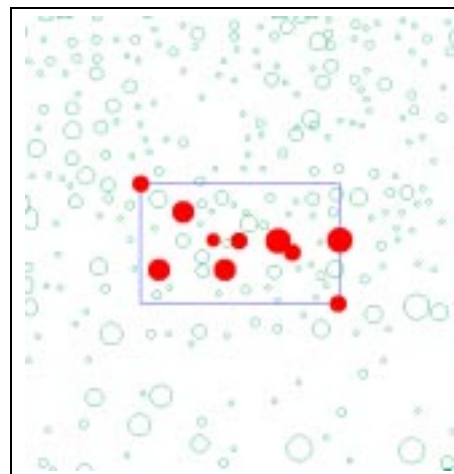
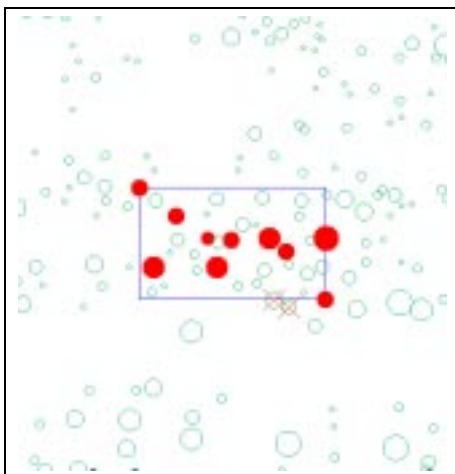


Figure 4: Spot sets detected with black spots indicating the local matching

4. **(Covering)** $\bigcup_{E \in \mathcal{E}} E$ covers at least a $(1 - p)$ -portion of pixels in ∂R_- .

The aim to find minimal-cardinality coverings is in accordance with Occam's razor principle, since the smallest cardinality set is, in absence of a master gel, the simplest hypothesis to explain how a complex region could have been evolved. Moreover, we especially emphasize that the somehow strange intersection condition 3 is justified by the application because two spots (ellipses) can only partially merge and do not form a cross.

In the following we present two algorithmic solutions to our covering problem that have been implemented and tested. In the implementation we assumed that the regions R are simply connected and rectilinear. However, it is straightforward how to extend the algorithms to arbitrary polygonal regions. Aiming at better runtimes, the sets ∂R_- and ∂R_+ are sampled by subsets S_- and S_+ . Since a convex boundary segment is more likely to be part of a single ellipse, S_- and S_+ are chosen in such a way that convex segments have denser samples.

2.2 Computing a Brute Force Solution

The basic idea behind the brute force solution is to generate all ellipses that fulfill condition 1 and 2. A voting scheme is then set up to select the covering.

In a first step we discretize the parameter space of possible ellipses. Recall that (the boundary of) an axis-parallel ellipse is formed by all points (x, y) fulfilling the equation

$$\frac{(x - c)^2}{a^2} + \frac{(y - d)^2}{b^2} - 1 = 0 \quad (1)$$

with parameters $a, b, c, d \in \mathbb{R}$.

To this end we restrict the ellipses to have centers which are pixel centers and one of the halfaxes, say a , has to have integer multiple pixel side length. Finally, to model condition 1, for a fixed a we restrict the second halfaxis to values from the set $\{as^i \mid -k \leq i \leq k\}$, with $s = \sqrt[k]{\rho}$ for a preset k .

Now, the algorithm simply computes for each center (c, d) and for each i the maximal halfaxis a so that the ellipse with parameters (a, as^i, c, d) does not intersect ∂R_+ . For each ellipse we store the points of S_- it covers. Eventually, we choose the covering in a greedy fashion. Assume a partial covering is already chosen. The next ellipse E is selected among all ellipses satisfying condition 3 (with respect to already chosen ellipses) according to the following criteria ranking in the order of appearance:

1. E covers a maximal number of previously uncovered points from S_-
2. E maximizes the length of a longest chain of consecutive covered points from S_-
3. E minimizes the maximal intersection area with an already chosen ellipse.

After selecting a best E (ties are broken randomly) we update the scores of all other remaining ellipses by deleting the points covered by E . We stop augmenting ellipses when condition 4 is met.

The obvious drawback of the brute force approach is that too many ellipses are tested. Moreover, by discretizing ellipse parameters we

may miss interesting ellipses like the big one in the right hand solution in Figure 5.

Remark: Let \mathcal{F} be the family of ellipses (a, as^i, c, d) , as described above, and let k_{opt} be the minimal number of ellipses of \mathcal{F} needed to cover all points of S_- and avoiding the ones of ∂R_+ . Then the standard $\log |S_-|$ -approximation (see [10]) of the optimal solution by the greedy approach cannot be guaranteed, at least for arbitrary sample sets S_- in polygons with holes. This is due to the restrictive intersection property. An example that illustrates this observation consists of a set of rather thin ellipses in a grid-like arrangement. Besides the recent paper [4] we are not aware of approximation results for set covers with restrictive intersection properties.

2.3 Using an LP Approach to Generate Ellipses

Compared to the brute force approach we do not want to restrict the set of ellipses under consideration for covering a region by an a priori parameter discretization and we want to avoid generating too many explicit ellipses.

Observe that each element in the pixel set ∂R_+ , now sampled by S_+ , forms a constraint for each ellipse in the cover, since any ellipse must not contain such pixels. On the other hand, we want at least a few points (say, at least three) from S_- to be included in an ellipse of the covering.

Therefore, we start from a randomly chosen triplet of mutually visible points from S_- and ask whether there is an axis-parallel ellipse containing these points so that it does not violate an outer constraint. Once having the information that for a given triplet there is a feasible solution one can efficiently extend the covered sample subset by an LP-based approach.

We start from the observation that the parameters of an ellipse E can be transformed into variables of an LP in such a way that each of the three points which have to be covered by E adds a linear constraint to the outer constraints. The constraints are derived by plugging a point $p = (p_x, p_y)$ into the ellipse formula $\frac{(p_x - c)^2}{a^2} + \frac{(p_y - d)^2}{b^2} - 1$ which has to be zero (negative, positive) for points on (inside, outside) E . Since this is not a linear constraint we start with one of the form

$$p_x x_1 - p_y^2 x_2 + p_y x_3 - x_4 = p_x^2 \quad (\text{resp. } <, >) \quad (2)$$

By elementary calculation, setting $t = \frac{x_1^2}{4} + \frac{x_3^2}{4x_2} - x_4$, we derive as ellipse parameters

$$a = \sqrt{t} \quad b = \sqrt{\frac{t}{x_2}} \quad c = \frac{x_1}{2} \quad d = \frac{x_3}{2x_2} \quad .$$

Given the existence of a feasible solution (not an explicit ellipse yet!) one wants to extend the point set that can be covered. Clearly, it makes sense to add and check first the neighbors of the already covered sample points. This is also done in a random way. However, it is clear that one can easily get stuck when there is a very restrictive partial solution, in the sense that we have still a feasible system but the actual solution set of ellipses is very small and does not allow to add a new sample point. To implement the necessary backtracking step, we have adapted a simplified version of the so-called Metropolis methodology, see for example [17], where it has

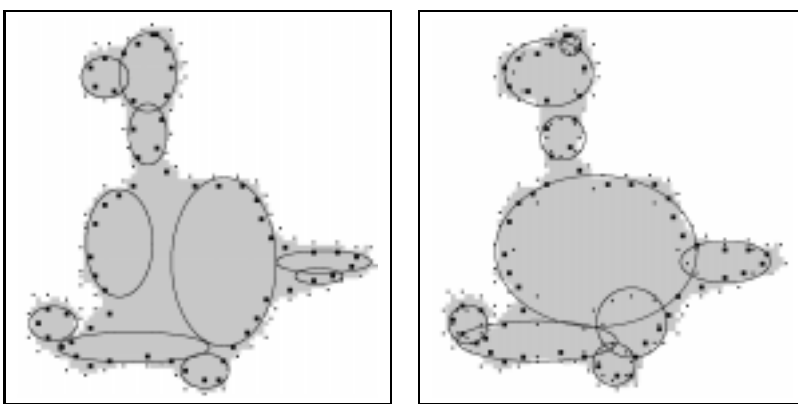


Figure 5: Ellipse covering computed by brute force method (left) and by LP approach (right)

been used for the generation of large cliques in graphs, a situation similar to ours.

This random Markov-chain like process is organized in rounds. With probability q we have a round “extend”, that is we choose and try to add a random neighbor; with probability $1 - q$ we have a round “backtrack” in which we delete a randomly chosen point from the already covered point set. After a fixed number of rounds we actually compute an explicit candidate ellipse to be included next into the partial covering. Among all possible ellipses we select the one that yields a halfpax ratio closest to 1. This is repeated a constant number of times and we select the candidate that is optimal according to the criteria above.

There remains to explain the incorporation of condition 3 into the LP-approach. As before we delete already covered points from S_- . But now for each selected ellipse E we add new “outer” constraints to S_+ by sampling the ellipse interior. These new points prevent later chosen ellipses from having large intersections with E . Again, this scheme is run until condition 4 is met. In the example depicted in Figure 5 the black pixels represent S_- . The dots outside the region describe the initial outer constraints, within the chosen ellipses the additional outer constraints are also indicated. Together they form the final set S_+ .

2.4 Related Theoretical Issues

We want to discuss the ellipse covering problem from a more theoretical point of view. Assume $P = \{p_1, \dots, p_m\}$ is an input polygon, now neither necessarily simple nor rectilinear, and let $\varepsilon > 0$ be a given fault parameter. Again we assume that the polygon has no cuts of length $\leq 2\varepsilon$. Moreover, we drop the condition that two ellipses in the covering can intersect in at most 2 points. Let S_- (resp. S_+) denote the vertices of the polygonal chains P_- (resp. P_+) which approximate the boundaries of the ε -neighborhood of P , have Hausdorff distance ε from P , and fulfill the property that the distance between any two consecutive vertices is at most ε .

The *ellipse covering problem* is to find a collection $\mathcal{E} = \{E_1, \dots, E_k\}$ of minimal cardinality of axis-aligned ellipses so that the union $U = \bigcup E_i$ contains all the points of S_- and none of the points of S_+ . Let n denote the total number of points in S_- and S_+ .

Note that we do not require that each point of ∂U has a point of P within distance $\leq \varepsilon$. Thus, we are willing to accept holes in

the solution. We will discuss later the problem of finding a set of ellipses that avoids S_+ and approximately covers the whole interior of P .

The problem of (approximately) covering a shape with ellipses is strictly related to the problem of exact covering of a shape with rectangles, which was shown to be NP-hard. It is also related to the problem of covering a shape with strips [1], and to the range covering problem in a hypergraph [7]. Thus, in the general setting there is not much hope for finding a polynomial-time algorithm. On the other hand, there is no much difficulty to obtain a polynomial time algorithm that approximates the set up to a $\log k_{opt}$ factor, where k_{opt} is the cardinality of the optimal-size solution, using the algorithm of [7]. The runtime of a straightforward implementation of that algorithm is $O(n^3 k_{opt} \log n)$. A much faster algorithm is presented in Section 2.4.2. But first we discuss the $\log n$ -approximation.

2.4.1 Efficient algorithms for the greedy approach

In the “traditional” greedy approach we perform a sequence of iterations, in each of which we find an ellipse that respects S_+ (that is, contains none of its points), and covers the maximal number of points of S_- not covered by previously chosen ellipses. Now we show how to perform each such iteration in time $O(n^3 \alpha(n) \log n)$. This, using standard greedy arguments, yields a runtime of $O(n^3 k_{opt} \alpha(n) \log^2 n)$ where $\alpha(n)$ is the inverse Ackermann function (see [23] for definitions and details).

Note that for each ellipse with parameters a, b, c, d equation (2) yields a dual representation by a point $(x_1, x_2, x_3, x_4) \in \mathbb{R}^4$ where $x_1 = 2c$, $x_2 = a^2/b^2$, $x_3 = 2a^2d/b^2$ and $x_4 = c^2 - a^2 + a^2d^2/b^2$. Consequently, the set of ellipses containing (not containing) a given point (p_x, p_y) is described by the halfspace $p_x x_1 - p_y^2 x_2 + p_y x_3 - x_4 > p_x^2$ (resp. $<$).

Let t_1, t_2 be two points of S_- , and consider the two-dimensional plane \mathcal{D} in \mathbb{R}^4 representing all axis-parallel ellipses that pass through t_1 and t_2 . Then \mathcal{D} is the intersection of the two hyperplanes that correspond to t_1, t_2 . Each point p of $S_- \cup S_+$ determines a line \tilde{p} in \mathcal{D} , and an ellipse E contains p if and only if \tilde{p} is below the point \tilde{E} which is the dual point of E in \mathcal{D} . Accordingly, E respects all points of S_+ if and only if \tilde{E} is below all lines corresponding to point of S_+ . Let \tilde{S}_- (resp. \tilde{S}_+) denote the collections of lines corresponding to points of S_- (resp. S_+). Let Γ denote the

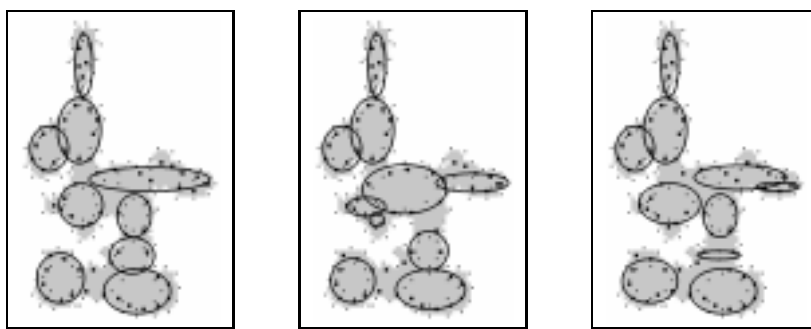


Figure 6: Three covering proposals for a complex region generated by the LP-approach

lower envelope of the lines in \tilde{S}_+ . We seek a point \tilde{E} which is below Γ , and above a maximum number of the lines of \tilde{S}_- . Consider the arrangement \mathcal{A}_- formed by the lines of \tilde{S}_- . Clearly \tilde{E} lies on an edge of one of the cells of \mathcal{A}_- intersected by Γ . The collections of these edges (and the vertices they define) is known in the Computational Geometry literature as the *zone* of Γ in \mathcal{A}_- , see [6] for further reference.

Using the recent algorithm of Har-Peled [14], the zone of Γ in \mathcal{A}_- can be constructed in time $O(n\alpha(n) \log n)$. Once the zone is created, finding a point in this zone that lies above the maximal number of lines of \mathcal{A}_- can be done by scanning this zone. The runtime is clearly proportional to the number of vertices in the zone. Since t_1 and t_2 are not known in advance, we need to iterate over all pairs of points of S_- , and repeat this process for every new ellipse that we find. Using standard greedy-algorithm arguments (see [10]), the number of ellipses we find is $O(k_{opt} \log n)$, thus the runtime is $O(n^3 k_{opt} \alpha(n) \log^2 n)$, as asserted.

2.4.2 $O(\log k_{opt})$ approximation

To show how one can find a set of $O(k_{opt} \log k_{opt})$ ellipses \mathcal{E} whose union covers the whole interior of P and avoids S_+ we use an efficient implementation of the Brönnimann & Goodrich paradigm:

Let D_- be the vertex set of a uniform grid of edge length $\varepsilon/2$ which is inside P (i.e., all points of the form $(i\varepsilon/2, j\varepsilon/2)$ for i, j integers). Then we replace S_+ by a set S'_+ which approximates the boundary of the $\varepsilon/2$ -neighborhood of P . We first sketch how to find a collection \mathcal{E}' of ellipses $\{E'_1 \dots E'_k\}$ whose union avoids S'_+ and covers D_- . The cardinality k of \mathcal{E}' is $O(k_{opt} \log k_{opt})$, where as above, k_{opt} is the minimal size of such a collection.

The Brönnimann & Goodrich approach assumes that k_{opt} is known (otherwise it is found using unbounded binary search), and is much less than $|S'_+| = O(n)$. It maintains a set \mathcal{E}' of ellipses whose union covers D_- . The solution we seek is a subset of \mathcal{E}' . It gives a weight $w(E')$ to each ellipse of $E' \in \mathcal{E}'$, initially all set to be 1. The algorithm consists of *phases*. At each phase, the algorithm picks a random sample R of size $ck_{opt} \log k_{opt}$ of the ellipses of \mathcal{E}' , where the probability of an ellipse to be picked is proportional to its weight, and c is a constant as computed in [7]. Next the algorithm checks if R covers D_- . If this is not the case, let p be a grid point which is not covered, (arbitrarily chosen if there are more than one) and let $\mathcal{E}'_p \subseteq \mathcal{E}'$ be the set of ellipses containing p . The algorithm doubles the weight of each ellipse of \mathcal{E}'_p , provided

that $\sum_{E' \in \mathcal{E}'_p} w(E') \leq w(\mathcal{E}')/2k_{opt}$. The algorithm stops when a cover is found. Since the Vapnik-Chervonenkis dimension of the problem is clearly finite, it follows from [7] that the algorithm stops after $O(k_{opt} \log n)$ phases.

Once \mathcal{E}' is found, we replace each ellipse E'_i of \mathcal{E}' by an enlarged ellipse E_i . Formally, E_i and E'_i share the same center, but each of the axes of E_i is by $\sqrt{2}\varepsilon/4$ longer than the corresponding axis of E'_i . Clearly, the collection $\mathcal{E} = \{E_1, \dots, E_k\}$ avoids the points of S_+ , since S'_+ was chosen accordingly. For each interior point of P is at distance at most $\sqrt{2}\varepsilon/4$ from a point of D_- the polygon P is completely covered.

The details of the efficient implementation of this paradigm are obtained by carefully choosing a relatively small number of ellipses, and using augmenting search trees for updating the weights of (implicitly represented) subsets of ellipses. They are omitted from this extended abstract and can be found in [11]. Altogether we have the following theorem.

Theorem 1: Let P and S_+ be as above. Then in expected time $O(n^3 \alpha(n) + n^2 k_{opt} \log^2 n)$ one can find a set \mathcal{E} of ellipses that covers the interior of P , and its boundary $\partial(\cup \mathcal{E})$ is in distance $\leq 2\varepsilon$ from the boundary of P . Moreover, the cardinality of \mathcal{E} is $O(k_{opt} \log k_{opt})$.

3. COMPUTING GLOBAL VIA LOCAL MATCHINGS

3.1 Modeling the Matching Problem

Assume that a source image S and a target image T are given by their spot lists. Recall that after the spot detection a “spot” is simply a vector $(x(s), y(s), i(s))$ consisting of its nonnegative point coordinates $(x(s), y(s))$ in the Euclidean plane and a positive real number $i(s)$ describing its intensity.

By this simplification of spots we can treat the geometric matching task as an approximate point pattern matching problem, see [2] for a survey on this field. There, the general setting for our bottleneck matching type problem is as follows. For a real number $\varepsilon \geq 0$, a group of admissible transformations \mathcal{A} (e.g. translations, rigid motions and/or scalings) and a metric d , one seeks a bijection f between as large as possible subsets $S' \subseteq S$ and $T' \subseteq T$, so that there exists a transformation $g \in \mathcal{A}$ for which $d(g(s), f(s)) \leq \varepsilon$ for every $s \in S'$. Thus, the transformation g describes the geometric resemblance between S' and T' . Additionally, we want that

the intensity $i(s)$ for $s \in S'$ resembles the intensity of $f(s)$. However, this is only the general framework for our solution. There are several major difficulties stemming from the inherent noise in the electrophoresis process:

1. 2–DE images, even of the same probe, can differ significantly and the offset vectors for matching spots are only locally almost equal.
2. The spot resolution of the images can be different. Moreover, given a correct matching the intensity orderings of matched spots in both images are similar but not identical.
3. It often happens that one has to compute a matching of images that only partially “overlap”. Previous algorithmic solutions assume that both images show the same frames, or at least there are preset landmark pairs that allow to initialize the matching procedure. We neither want to choose the matching frames nor landmarks by hand.

To cope with these problems we choose the following 2–stage approach, which we call the global-via-local matching paradigm.

1. In a first step we compute a rather dense set of landmarks. This is realized via local matchings. A local matching is the registration of a small local source pattern (consisting of locally most intensive spots) in the target. For a single pattern the result is typically not unique. But for a set of grid–like located local source patterns we can choose a consistent common registration (i.e., matching) by comparing the underlying transformations, which should be similar for neighboring patterns.
2. A second step implements the extension of the landmark matching to neighboring spots. This is done by a variant of the standard neighborhood graph comparison. However, our advantage is the possibility to use information gathered in the spot detection preprocessing.

Next we formalize the geometric matching criterion for the registration of a local pattern $P \subset S$ in the target T .

We call two line segments $\overline{ss'}$ and $\overline{tt'}$ (λ, α) –similar if the absolute difference of their angles with the x –axis is smaller than α and for their lengths we have:

$$1 - \lambda \leq \frac{|\overline{ss'}|}{|\overline{tt'}|} \leq 1 + \lambda$$

Two point patterns $P \subset S$ and $Q \subset T$ (λ, α) –match if there is a bijection f between the point sets so that $\overline{ss'}$ and $\overline{f(s)f(s')}$ are (λ, α) –similar for all $s, s' \in P$. In sum, from the geometric point of view we want to find (λ, α) –matchings f between as large as possible subpatterns $P' \subset P$ and local target patterns Q , compare also [19].

As described in [15], we cannot model intensity resemblance by directly comparing $i(s)$ and $i(f(s))$, instead we rank, in both source and target, the spot intensities into 10 groups and require that the matching respects those ranks within a tolerance of 1.

3.2 How to Compute Local Matchings

In this subsection we present an efficient algorithm which computes partial approximate matchings of a pattern P in a target point set T .

The algorithm has to find maximal subsets $P' \subseteq P$ (at least 50 %), corresponding subsets $Q \subseteq T$ and transformations t approximating the matching between P' and Q . Firstly, we discuss the transformation class of translations, but, we always keep in mind possible generalizations to homothetic transformations (translations plus scalings). In the runtime analysis k and n will denote the number of points in P and in T , respectively.

The algorithm is basically a twofold refinement of the naive alignment approach which computes a geometric matching by aligning a pattern edge $e \in P$ with a similar edge $e' \in T$. If one wants to find full matchings of P , it is sufficient to fix e and search for all similar target edges. However, looking also for partial matchings, one has to check all pattern edges against all edges in T . Each edge similarity induces a translation $t_{e,e'}$ which maps the midpoint of e onto the midpoint of e' . Then candidate points for a matching can be found by nearest neighbor queries for the k points of the transformed pattern $t_{e,e'}(P)$. Hence, the runtime is bounded by $O(k^3 n^2 \log n)$ which combines the $k^2 n^2$ edge comparisons with the $O(k \log n)$ time to compute a matching. However, the algorithm is much faster in practice because most of the edges similarity tests are answered negatively.

Remark 1: It is not hard to construct examples where the nearest neighbor $q \in T$ of a transformed pattern point does not satisfy the requirements of a geometric matching whereas another neighbor q' does. Therefore all neighbors within a certain distance have to be checked.

Remark 2: Looking only for matchings approximated by translations it would be sufficient to check all transformations defined by point pairs $(p, q) \in P \times T$. However, such an approach is not very suitable in practice because it is not extendable to homothetic transformations and, moreover, the resulting $O(k^2 n \log n)$ time bound is tight.

Remark 3: In [16] a randomized version of the alignment method has been studied. One of the results is a $O(kn^2 \log n)$ Monte Carlo algorithm computing all partial matchings which match at least a constant fraction of P .

Instead of randomization we make use of the intensity resemblance which is an additional matching requirement. To this end we order our point sets according to decreasing intensity. A triangulation of a point set X in the plane is called *Delaunay triangulation* if for each triangle in the triangulation its circumcircle contains only the three triangle points. One can construct such a triangulation in an incremental way by inserting points one by one in the given order, compare [13]. The insertion of a point can destroy some of the previous edges and add some new edges. The *history* $\text{Hist}(X)$ of the *incremental triangulation* is the set of all Delaunay edges which occur at some time in this process.

There are two results from Computational Geometry which together make up a fast and very robust heuristic.

Theorem 2 (see [3]): Let X be a point set with an intensity ordering. Assume that a pattern $Q \subset X$ consists of the most intensive points in a rectangle R . Consider a triangle Δ occurring during the incremental Delaunay triangulation of Q . If the circumcircle of Δ is contained in R , then each edge of Δ belongs to $\text{Hist}(X)$.

This implies that under ideal assumptions (of nearly identical im-

ages) it suffices to consider all similarities between edges from $\text{Hist}(P)$ and $\text{Hist}(T)$. In practice however one cannot start from these assumptions: the matching is only approximate, sometimes partial, and moreover the intensity orders can slightly vary. To cope with these difficulties we augment the history with all flip edges which occur in the incremental triangulation. We denote the extended history by $\text{Hist}^*(\cdot)$. The following fact implies an estimate on its expected size.

Theorem 3 (see [22]): Let X be a random permutation of n points. Then the incremental Delaunay triangulation of X can be computed in $O(n \log n)$ expected time. The expected number of edges in $\text{Hist}(X)$ is $O(n)$.

Therefore the expected size of $\text{Hist}^*(X)$ is also linear. Moreover, this construction ensures that many more edge similarities can be found, see [15] for more details and experimental results for randomly generated examples.

Summing up, the restriction of the alignment approach to the comparison of edges from $\text{Hist}^*(P)$ and $\text{Hist}^*(T)$ implies a matching algorithm with expected time complexity $O(k^2 n \log n)$.

In fact, although this first improvement reduces the computation time remarkably, there is still a lot of wasted work because for any edge similarity $e \sim e'$ the algorithm attempts to construct a matching approximated by translation $t_{e,e'}$ in $O(k \log n)$ time. However, most of these attempts are bound to fail. We call a transformation $t_{e,e'}$ *good* if it approximates a matching of at least half of the points in P and *bad* otherwise.

The standard approach of how to avoid testing bad transformations is geometric hashing. Originally hashing has been worked out for the exact matching problem. The idea is to collect first all candidate transformations in a hash table. Note that in contrast to bad transformations the good ones occur at least $\binom{k/2}{2}$ times. In the end, the good transformations can be selected and, moreover, for each of them the corresponding matching has to be computed only once.

Here we develop a special variant of geometric hashing. We replace the hash table by a *scoring scheme* which uses a regularly spaced $m \times m$ grid over T with m^2 variables counting scores for all grid nodes. We discuss later how to choose m . This structure is suitable to compute also approximate matchings and it beats hash tables in both time and storage complexity. One has to pay for these advantages by accepting a certain heuristic flavor in this method. Let c_P denote the center of the bounding box of the pattern P . Again we start with all translations $t_{e,e'}$ defined by similar edges $e \in \text{Hist}^*(P)$ and $e' \in \text{Hist}^*(T)$. We store $t_{e,e'}$ adding scores to the four nodes of the grid cell which contains $t_{e,e'}(c_P)$. It is straightforward that good transformations yield clusters of points which in turn are represented by high scores in the neighboring grid nodes. Altogether we get an expected runtime of $O(n \log n + kn + m^2 + |\mathcal{T}_{\text{good}}| k \log n)$, where $|\mathcal{T}_{\text{good}}|$ denotes the number of good transformations, i.e., the number of computed partial matchings. The first and second term estimates the triangulation construction and the edge comparisons. Since the last term is of smaller order the number of grid nodes, i.e. m^2 , plays the key role in the algorithm's analysis. Finer grids can display the cluster positions more precisely than coarser grids (with less nodes).

There is one more trick that allows to have both rather coarse grids

and a sufficient precision of cluster positions. Each translation representative $t_{e,e'}(c_P)$ subdivides its grid cell into four rectangles. Instead of unit scores, each of the four grid nodes adds to its current score an amount proportional to the area of the opposite rectangle. This way the precise position of a single point $t_{e,e'}(c_P)$ can be recomputed from its scores. Let $\text{Score}(i, j)$ be the total score accumulated in grid node (i, j) after probing all edges from $\text{Hist}^*(P)$. All local maxima in the grid that are greater than a threshold value depending on $|P|$ are considered to correspond to potential matching locations.

Eventually, we can approximate the actual center (i_c, j_c) of the vector cluster stemming from a local maximum at node (i, j) by computing a weighted average of the scores at (i, j) and all scores at neighboring grid nodes:

$$(i_c, j_c) = \frac{\sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} \text{Score}(k, l) \cdot (k, l)}{\sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} \text{Score}(k, l)}.$$

The generalization from translations to homothetic transformations is straightforward. (λ, α) -similarity has to be replaced by α -similarity (the angle between the edges is at most α) and for each α -similar pair (e, e') the transformation $t_{e,e'}$ is defined by the scaling factor $|e'|/|e|$ and the translation mapping the midpoint of the scaled e onto the midpoint of e' . Consequently, the two-dimensional scoring scheme has to be replaced by a three-dimensional structure with the third dimension representing the scaling factors. In our implementation we used an exponential scale of the third axis where the factor between two grid units is λ .

3.3 Extending Local Matchings to a Global Matching

The local point pattern matching described in the previous subsection is the central module of our algorithm. We compute the global matching of two images S and T in several stages. First the source image is segmented into equally sized, rectangular subimages. In our implementation we used a regular 5×5 -grid for this subdivision. Then for each of the 25 subimages the pattern formed by its 12 most intensive spots is selected (25 and 12 are parameter values which proved to be optimal in numerous tests). Applying the local matching to those local patterns we get a list of matching proposals for each pattern. Now we apply a simple consistency test to those proposal lists. Two local matchings are consistent if the underlying transformations are approximately the same. Finally a first skeleton for the global matching is established by searching for the largest family of pairwise consistent local matchings. Note that this skeleton has at most 300 spot pairs. To get more pairs we follow the standard approach: The pairs in the skeleton M form a set of landmarks which defines a piecewise affine transformation t_M . Next for each point $p \in S$ we find the nearest neighbor q of $t_M(p)$ in T . Such pair (p, q) will be included in the matching depending on the similarity of the spot neighborhoods. This is the place where different proposals from the decomposition of complex spot regions come into play. If it is impossible to find in T a spot with similar neighborhood, there might be one in another decomposition proposal.

Another advantage of our approach is the possibility to compute a "global" matching for images which only partially overlap. Up to now commercial programs assume a global alignment of the images. Figure 7 shows an example. This feature is useful for reconstructing big gel images from several partially overlapping subimages.

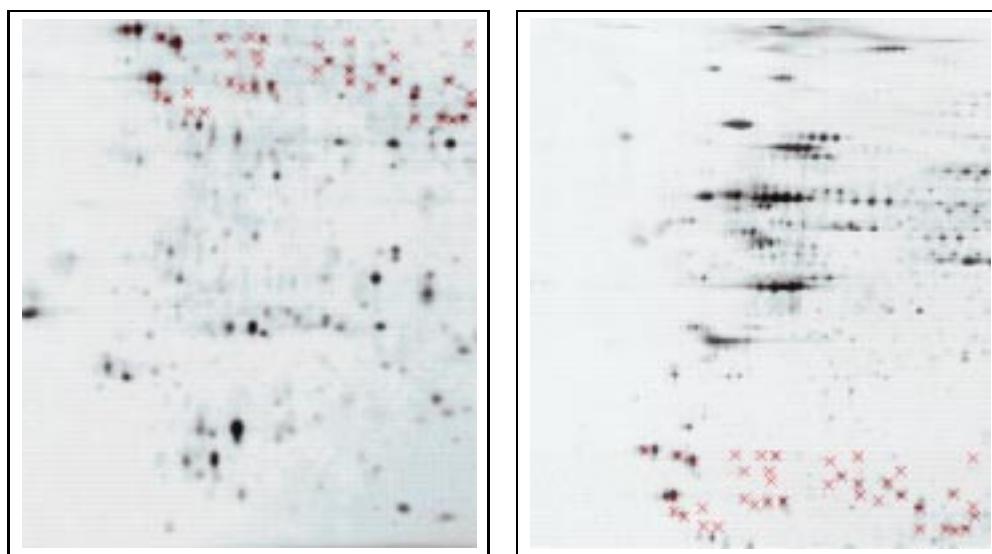


Figure 7: A global matching of two images which have only a small horizontal strip in common

4. IMPLEMENTATION ISSUES AND EXPERIMENTAL RESULTS

The 2-DE analysis software system CAROL ([8]) contains the described spot detection and matching algorithms. In order to facilitate its usage over the Internet it consists of two main parts: The first part, the combinatorial and geometrical kernel of the spot detection and matching algorithms, has been implemented in C++. It makes essential use of the Standard Template Library (STL) and of the Computational Geometry Algorithms Library (CGAL) [9]. The latter library provides several geometric data structures and functions and especially an implementation of the incremental Delaunay triangulation. For the LP approach of the spot detection, which we implemented for internal use only, we use the commercial LP-solver CPLEX.

The second part of the CAROL system is the graphical user interface which has been implemented in Java. It can be run as an applet started out of an internet browser or as an application. The communication with the algorithmic procedures is established via internet sockets, whereby the C++ program works as a server which waits for matching or spot detection requests from the Java-client, performs the computation, and eventually sends the results back to the client.

Using CAROL it is possible to analyze gel images from databases all over the Internet. This feature is strongly supported by the client-server architecture and the possibility to run the user interface out of an internet browser. However, a direct Internet access is required that is not restricted by a firewall. CAROL offers the possibility to open GIF images from any 2-DE database, to carry out the spot detection, to perform local or global matchings between two gel images, and to set parameters like tolerance bounds, pattern size, etc. The current version of CAROL can be found at <http://gelmatching.inf.fu-berlin.de>.

The local matching algorithm executed on a Sun Ultra Sparc 300 MHz computes the best ten matchings for a pattern of twelve spots in about 0.2s. The first stage of the global matching between two

gel images with 900 and 1000 spots takes about 7s and yields about 110 matching spot pairs. After this step the user can optionally correct the proposed landmarks. A subsequent extension step takes 2s. The computation of the spot detection for a full gel image (1200 x 1500 pixels, 4500 spots) takes about half a minute without complex regions; in such large images we observed between 0 and 20 complex regions. The brute force greedy approach for the complex region in Figure 5 (subset of a 76 x 80 array, with 59 inner sample points drawn in black) needs 24s to compute the 10 ellipse covering indicated. The LP-based solution (with initially 76 points defining outer constraints) takes 13s to compute the indicated solution. However, we remark that usually a spot detection is only computed once before storing the image in a database.

Tests of the CAROL system on several series of gel images as well as the positive feedback from external test users have confirmed the advantages of our geometric approach. Further work will concentrate on a more accurate mathematical modeling of the biochemical and physical phenomena within the electrophoresis process.

5. REFERENCES

- [1] P.K. Agarwal and C.M. Procopiuc, Covering Points by Strips in the Plane, *Proc. 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2000, 538–547.
- [2] H. Alt and L. Guibas, Discrete Geometric Shapes: Matching, Interpolation, and Approximation, A Survey, In J. Urrutia, J.-R. Sack, eds., *Handbook of Computational Geometry* (North Holland), 1999, 121–153
- [3] H. Alt, L. Knipping, and G. Weber, Point Pattern Matching in Astronautics, *J.Symbolic Computation* 17 (1994) 321–340
- [4] V. Anil Kumar, S. Arya and H. Ramesh, Hardness of Set Cover with Intersection 1, *Proc. ICALP'2000*, Lect. Notes in Comp. Sc. 1853, 624–635
- [5] R. Appel, J. Vargas, P. Palagi, D. Walther and D. Hochstrasser, Melanie II, a third-generation software package for analysis

- of two-dimensional electrophoresis images: II. Algorithms, *Electrophoresis* 18 (1997), 2735–2748
- [6] M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, *Computational Geometry, Algorithms and Applications* (Springer), 1997.
- [7] H. Brönnimann and M. T. Goodrich, Almost Optimal Set Covers in Finite VC-Dimension, *Discrete Comput. Geom.* 14 (1995), 463–479.
- [8] CAROL, Software system for Matching 2DE Gel Images, <http://gelmatching.inf.fu-berlin.de>
- [9] CGAL, The Computational Geometry Algorithms Library, <http://www.cs.ruu.nl/CGAL>
- [10] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms* (MIT Press), 1990.
- [11] A. Efrat, F. Hoffmann, K. Kriegel, and C. Schultz, Covering shapes by ellipses for the computer analysis of protein patterns, Technical Report B 00–11, July 2000, Institut für Informatik, Freie Universität Berlin
- [12] J. Garrels, The QUEST System for quantitative analysis of 2D gels, *J. Biological Chemistry*, 264 (1989), 5269–5282
- [13] L. Guibas, D. Knuth, and M. Sharir, Randomized Incremental Construction of Delaunay and Voronoi Diagrams, *Algorithmica* 7(4) (1992) 381–413
- [14] S. Har-Peled, Taking a Walk in a Planar Arrangement, *Proceedings 40th Annual IEEE Symposium on Foundations of Computer Science*, 1999, 100–110.
- [15] F. Hoffmann, K. Kriegel, and C. Wenk, An applied pattern matching problem: comparing 2D patterns of protein spots, *Discrete Applied Mathematics* 93 (1999), 75–88
- [16] S. Irani and P. Raghavan, Combinatorial and experimental results for randomized point matching algorithms, *Computational Geometry: Theory and Applications* 12 (1999), 17–31
- [17] M. Jerrum, Large Cliques Elude the Metropolis Process, *Random Structures and Algorithms* 3 (4) (1992), 347–359
- [18] K. Kriegel, I. Seefeldt, F. Hoffmann, C. Schultz, C. Wenk, V. Regitz-Zagrosek, H. Oswald, and E. Fleck, An alternative approach to deal with geometric uncertainties in computer analysis of two-dimensional electrophoresis gels, *Electrophoresis* 21 (2000), 2637–2640
- [19] H. Ogawa, Labeled Point Pattern Matching by Delaunay Triangulation and Maximal Cliques, *Pattern Recognition* 19(1) (1986), 35–40
- [20] P. F. Lembkin, Comparing two-dimensional electrophoretic gel images across the Internet, *Electrophoresis* 18(3–4) (1997), 461–470
- [21] K.–P. Pleißner, F. Hoffmann, K. Kriegel, C. Wenk, S. Wegner, A. Sahlström, H. Oswald, H. Alt, and E. Fleck, New algorithmic approaches to protein spot detection and pattern matching in two-dimensional electrophoresis gel databases, *Electrophoresis* 20 (1999), 755–765
- [22] R. Seidel, Backwards Analysis of Randomized Geometric Constructions, in J. Pach, ed., *New Trends in Discrete and Computational Geometry* (Springer), 1993
- [23] M. Sharir and P.K. Agarwal, *Davenport-Schinzel Sequences and Their Geometric Applications* (Cambridge University Press), 1995.
- [24] M. R. Wilkins, K. L. Williams, R. D. Appel, and D. F. Hochstrasser (Eds.) *Proteome Research: New Frontiers in Functional Genomics* (Springer), 1997