Efficient and Small Representation of Line Arrangements with Applications

David P. Dobkin *
Department of Computer Science
Princeton University
Princeton, New Jersey 08544
dpd@cs.princeton.edu

Ayellet Tal
Department of Electrical Engineering
Technion
Haifa, Israel
ayellet@ee.technion.ac.il

ABSTRACT

This paper addresses the problem of lossy compression of arrangements. Given an arrangement of n lines in the plane, we show how to construct another arrangement consisting of many fewer lines. We give theoretical and empirical bounds to demonstrate the tradeoffs between the size of the new arrangement and the error from lossiness.

For the specific application of computing discrepancies of point sets, we demonstrate that speedups by factors of several hundred are possible while introducing small errors. This research has been enabled by various visualization techniques and is accompanied by a video.

Categories and Subject Descriptors

H.4.m [**Information Systems**]: Miscellaneous; D.2 [**Software**]: Software Engineering; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

1. INTRODUCTION

The line arrangement is a fundamental data structure for computational geometry problems. The list of application areas where line arrangements are used includes motion planning [38, 20, 29], assembly planning [19, 21], aspect graphs [17, 1], molecular modeling [22], hidden surface removal and visibility computations [32, 35], largest empty polygons [16, 11], ham sandwich cuts [15], and the computation of the discrepancy of point sets for sampling applications [10, 9, 5]. There has also been considerable interest in finding properties of portions of a line arrangement [2, 7].

One of the difficulties that arises in using arrangements to solve real problems is their complexity. Many geometric algorithms have running times that are quadratic or cubic in the number of regions. For an arrangement, the input is measured as a number n of lines or points. That gives rise to an arrangement of n^2 regions. This leads to algorithms of running times potentially as high as $O(n^4)$ or $O(n^6)$. Such running times are prohibitive in practice. In such cases, applications developers take shortcuts to avoid this blow up in complexity. Examples of these shortcuts include the development of probabilistic algorithms for motion planning [4, 28] and the limiting of motion planning algorithms to "fat" objects [42]. We propose here a different alternative to handling this problem. Rather than developing a faster algorithm, we reduce the input size while maintaining a good approximation to the original input.

Our approach is to use techniques of compression to reduce the size of a line arrangement. The underlying belief is that a line arrangement defined by a large collection of lines consists mainly of regions of small area. If the lines were thicker (e.g. if the arrangement had been drawn with a thick pen), these regions would have disappeared into ink smudges. For many applications, what is most interesting is not the smudges but rather approximations to the regions that survived. We present an algorithm that reduces the number of lines defining the arrangement by a constant fraction. This makes the standard algorithms for computing on arrangements run much faster. For example, consider a situation where an algorithm of running time $O(n^6)$ will be applied to an input of size n. Imagine that we reduce the input size by a factor of 10. The result will be an algorithm that runs 1,000,000 times faster. We show in this paper that such improvements are possible.

As an example of our work, we consider the problem of computing the discrepancy of a set of points [39, 10, 9, 5]. Previously, we showed a method to reduce the complexity of this computation from $O(n^3)$ to $O(n^2)$. Building upon that implementation, we now give an approximation scheme that reduces the constant factor. Our empirical studies show that for 5000 input points, we can get speedups by factors of 500 while introducing errors of less than 2.2%. The techniques used to do this are precisely the techniques described for reducing the complexity of arrangements.

Our ideas build on the use of duality in the representation of an arrangement. We use the dual transform to map the input lines to points in the dual space. Points in the dual space are then matched to near neighbors. Near neighbor point pairs are then replaced by a single point. This introduces an error into the arrangement. We demonstrate that

*Work by David Dobkin supported in part by NSF Grants CCR-97-31535 and CCR-99-88173, ARO Grant DAAD19-99-1-0205 and a Fullbright Fellowship.

SCG'01, June 3-5, 2001, Medford, Massachusetts, USA.

such errors are bounded. Our theoretical analysis shows that any arrangement of points in the unit square must have a sufficient number of near neighbor pairs. Since points that are close in dual space correspond to lines that are close in the primal space, we are able to prove theoretical bounds on the error introduced. We have also implemented our algorithms to show that our proven error bounds are quite conservative in practice.

We view the contributions of this paper to be two-fold. We propose a solution to a problem in computational geometry – that of doing efficient computation on the large structures that arise when computing with arrangements. We also give a general technique that we believe is very powerful and will find application to other problems in computational geometry. We show the application of this technique to the problem of computing the discrepancy and suggest other problems where it is likely to be useful.

Visualization has played a central role in this research, helping us to test conjectures, to develop ideas, and to visualize the algorithm [41]. The accompanying video [12] was developed along with the development of the paper.

In the next section, we give background. Section 3 describes our algorithm. In section 4, we derive theoretical error bounds on our approach. Section 5 describes our implementation and gives our empirical results. We conclude with a discussion of other applications of our techniques.

2. BACKGROUND

We provide in this paper an algorithm to approximate and so to compress arrangements in a lossy manner. In this section we first define arrangements and then discuss some compression schemes used in the geometric domain.

Arrangements

Arrangements [2, 7, 36, 18] are fundamental structures of computational geometry. Given a finite set H of hyperplanes in R^d, the arrangement $A(H)$ is the decomposition of R^d into connected open cells of dimensions $0, 1, \cdots, d$ induced by H. A $d-$dimensional cell in $A(H)$ is a maximal connected region of R^d not intersected by any hyperplane in H. A $k-$dimensional cell in $A(H)$, for $0 \leq k \leq d-1$, is a maximal connected region in the intersection of a subset of the hyperplanes in H that is not intersected by any other hyperplane in H. Arrangements can also be defined on other types of objects such as spheres and curved surfaces in R^d.

Specifically, in two dimensions, let L be a set of n lines in the plane. The set L induces a subdivision of the plane that consists of vertices, edges, and faces ($0, 1$ and 2-dimensional cells, respectively). This subdivision is referred to as the *line arrangement* induced by L and is denoted by $A(L)$.

The complexity of an arrangement is the total number of vertices, edges, and faces of the arrangement, $\theta(n^2)$ in the worst case.

The algorithm we propose in this paper works in dual space. There are various known point–line duality transforms. We use here three types of duality transforms.

$$polarduality[6] : L : ax + by = 1 \Leftrightarrow p : (a, b)$$

$$Dual1 : L : y = ax - b \Leftrightarrow p : (a, b)$$

$$Dual2 : L : ax + by = 1 \Leftrightarrow p : (a/(a^2 + b^2), b/(a^2 + b^2))$$

The *Dual2* transform has the feature that the line passes through its dual point.

Compression in Geometry

Compression trades off space and time against error. Numerous techniques have been developed for (lossy and lossless) compression of images [24, 34, 37, 40, 43]. These techniques are aimed at reducing storage space and transmission time. More recently, there has been a flurry of activity aimed at the compression of geometric structures. This work applies to polyhedra and unstructured meshes. The goal is to develop schemes that generate lossy or lossless representations of the topology and geometry of the structure. Two approaches are popular here. The first is the compression of the representation of the topology and the vertex geometry of the meshes, as was done in [8, 3, 27, 26]. The second avenue allows the modification of the topology of the given mesh, through the use of a multi-resolution representation [23, 31, 30, 14].

These compression schemes are evaluated by measuring the tradeoff between the space reduction and the quality of the image produced or the degree to which the reduced mesh approximates the original. Often these measures are aesthetic since there are no measures that can compute the relative quality of different representations.

Our goals here are different. We seek compressed representations of geometric structures that will behave similarly to the original representations when used as inputs to various algorithms. So, rather than reducing transmission speeds, we seek to reduce the running times of these algorithms. We measure the quality of our scheme both aesthetically and empirically. In the former case, we argue that our reduced arrangements "look" like the original arrangements they are meant to approximate. In the latter case, we measure the error that results from using our reduced representations in place of the original inputs to algorithms that compute on arrangements. We also give theoretical arguments to derive conservative error bounds.

3. THE ALGORITHM

Given a collection L of n lines in a bounding box in the plane, the algorithm returns a collection L' of $f(n)$ lines in the same bounding box, such that the arrangement of the $f(n)$ lines of L' is an approximation to the arrangement of the n lines of L.

The main idea of the algorithm is to find a subset of the collection of lines that can be approximated by a set of fewer lines. Since the distance between a pair of points is a more intuitive concept than the distance between a pair of lines, we often shift to the dual space to present our ideas. We show in the next section that this approach is justified. Using identifications made in the dual space, close lines in the primal space can be grouped together. We show that closeness (between points) in dual space implies closeness (between lines) in primal space. In this way, the quality of the matching of points assures the quality of the approximation of arrangements.

Thus, given a set of lines, L, our algorithm proceeds as follows:

1. **Transform the set of lines L to the set P of their dual points**

2. **Identify groups of points of P that lie close together**

3. **Replace each group of points by a smaller group of approximating points**

4. **Transform the points of step 3 to a set of lines L' that approximates L**

The results presented here were computed using a simplified version of this algorithm. In step 2, we computed a set of k non-overlapping pairs of close points. where k is a user specified parameter. Typically, we choose a small value of k (even $k = 1$ works) and iterate between steps 2 and 3 to allow us to create larger groups of points. For example, we might identify points A and B in step 2 and replace them by point X in step 3. On the next iteration, point X is identified with point C and the pair is replaced by point Y. As a result, we have replaced the group A, B, C of points by the point Y.

Several strategies are available for choosing the replacement point for a pair of points in step 3. The obvious thing to do is to replace them by the midpoint of the segment joining them. As seen in Figure 1 (where the original lines are drawn in red and the line dual to the midpoint is drawn in blue), the line dual to this midpoint may not be a natural approximation. There are two problems with this scheme. When two lines intersect they generate two pairs of wedges. One is defined by an acute angle and one by an obtuse angle. For our approximation we would like to choose the bisector of the wedge defined by the acute angle. The midpoint, however, may correspond to a line through the wrong pair of wedges, as shown in Figure 1(a). Also, even when the midpoint line is in the right pair of wedges, it has a slope that is equal to the average of the other two slopes. When one slope is small and the other large (see Figure 1(b)), this line is very close to the line of large slope, and so does not bisect the wedges as we would like.

To fix this, we do the computation in the primal space. First, we find the lines dual to the points. Next, we find the bisector of the wedge pair defined by the acute angle at the intersection of these lines. The point dual to this bisector is the replacement point for the two original points.

Our implementation of the algorithm iterates on this process of finding good matches and replacing the matched lines by their bisectors, building a coarser arrangement at each stage. The algorithm iterates until a required level of compression has been achieved or until there is no close pair of lines to be matched. Both decision parameters can be controlled by the user.

4. ERROR BOUNDS

We next prove that the above strategy works. As we demonstrate in the empirical section of the paper, the error bounds given here are extremely conservative estimates of the worst case. In practice, the results are significantly better.

We begin by defining the distance measures we will be using. For two points p and q in the plane, we define $d(p, q)$, the distance between them, as the Euclidean (i.e. L_2) distance between them. We extend this to the definition of Hausdorff distance ([25]) for point sets as follows:

DEFINITION 1. *Let $S_n = \{s_1, ...s_n\}$ and $T_m = \{t_1, ...t_m\}$ be point sets of size n and m. $d(S_n, T_m)$, the distance between these sets, is defined as the larger of*
$\max_{1 \le i \le n} \min_{1 \le j \le m} d_L(s_i, t_j)$ *and*
$\max_{1 \le i \le m} \min_{1 \le j \le n} d_L(t_i, s_j)$

Our goal is to find a set T_m that approximates S_n where $m < n$. As a first step, we show that there is a subset of S consisting of pairs of points that lie close together.

In what follows, we will define R_L as the axis-aligned square rooted at the origin with opposite corner (L, L).

LEMMA 4.1. *Let S_n be a set of n points in R_L. Then, there are k disjoint pairs of the points of S_n such that the points of each pair lie within distance D of each other where $D = L * \sqrt{2/(n-2k)}$ (or $k = n/2 - L^2/D^2$).*

PROOF. Divide R_L into l^2 boxes each of side length L/l. For each box that contains an odd number of points, discard one of its points (at most l^2 points are discarded). Now, all of the remaining points can be paired at a distance of at most $L\sqrt{2}/l$. If $D = L\sqrt{2}/l$ then $(n - l^2)/2$ pairs can be created, so $k = (n - l^2)/2 = n/2 - L^2/D^2$. □

We can replace the pairs identified in this lemma by their midpoints which gives the result:

THEOREM 4.2. *Given a set S_n of n points in R_L and $D > 0$, we can always find a set T_m such that $d(S_n, T_m) \le D$ where $m \le n/2 + L^2/(4D^2)$.*

PROOF. Using the lemma above we can always find $k = n/2 - L^2/(4D^2)$ point pairs that lie within $2D$ of each other. Each of these pairs is replaced by its midpoint – a point that lies within D of the points of the pair. We then build the set T_m as the union of the $n - 2k$ points that are not matched with these k midpoints. This yields a set of $m = n - k = n/2 + L^2/(4D^2)$ points and $d(S_n, T_m) \le D$. □

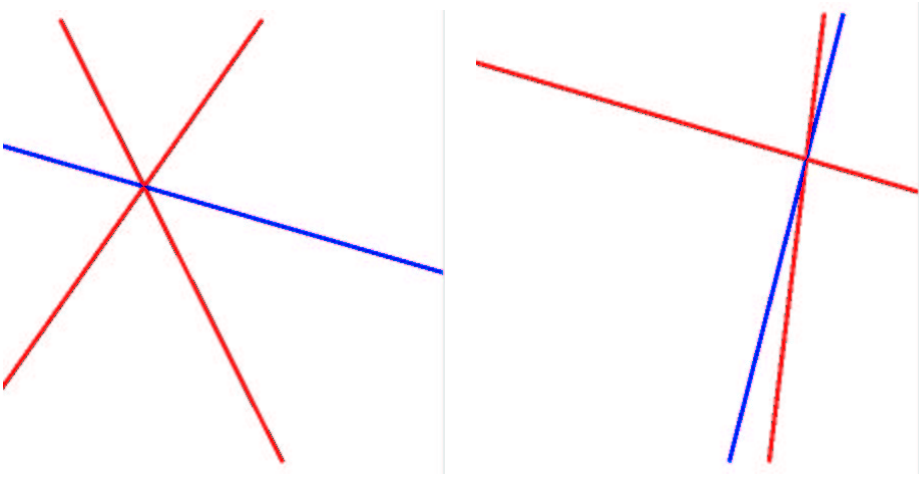
This theorem applies to points. Next we show that close for points in dual space implies close for lines in primal space. This allows us to extend the theorem to lines. We begin by defining a distance measure between lines. Because lines are infinite, we must take care in defining this distance to keep it finite and meaningful. We do so by defining distances within R_L.

DEFINITION 2. *If l_1 and l_2 are lines that intersect R_L, $d_L(l_1, l_2)$, the distance between l_1 and l_2 in R_L is $A_L(l_1, l_2)/L^2$ where $A_L(l_1, l_2)$ is defined as follows:*

1. *If l_1 and l_2 do not intersect within R_L, then $A_L(l_1, l_2)$ is the area of the polygon defined by l_1 and l_2 within R_L.*

2. *If l_1 and l_2 are perpendicular lines that intersect within R_L, their intersection creates 2 pairs of wedges whose areas can be computed. In this case, $A_L(l_1, l_2)$ is the smaller of these areas.*

3. *If l_1 and l_2 are non-perpendicular lines that intersect within R_L, their intersection creates 2 pairs of wedges with wedge angles of θ and $\pi - \theta$ where $0 < \theta < \pi/2$. In this case, $A_L(l_1, l_2)$ is the sum of the areas of the pair of wedges of wedge angle θ.*



(a) line through the wrong pair of wedges (b) line close to the line of large slope

Figure 1: Midpoint replacement

We extend this in Hausdorff fashion to collections of lines (i.e. arrangements) that intersect R_L.

DEFINITION 3. *Let A_n be an arrangement of the n lines $\{l_1, ..., l_n\}$ and B_m be an arrangement of the m lines $\{s_1, ..., s_m\}$. $d_L(A_n, B_m)$ the distance between A_n and B_m is defined as the larger of* $\max_{1 \le i \le n} \min_{1 \le j \le m} d_L(l_i, s_j)$ *and* $\max_{1 \le i \le m} \min_{1 \le j \le n} d_L(s_i, l_j)$

We now extend the previous theorem to lines.

THEOREM 4.3. *If p and q are points of R_L with $d(p, q) = D$ then $d_L(Dual1(p), Dual1(q)) \le D\sqrt{4 + L^2}/2L$.*

PROOF. We represent p as (a, b) and q as $(a + \epsilon, b + \delta)$ where $D = \sqrt{\epsilon^2 + \delta^2}$. So, $l_p : ax - y = b$ and $l_q : (a+\epsilon)x - y = b + \delta$ are $Dual1(p)$ and $Dual1(q)$ respectively.

We consider the trapezoid defined as the convex hull of the points $(0, -b)$, $(0, -(b+\delta))$, $(L, aL-b)$ and $(L, (a+\epsilon)L-(b+\delta))$ defined as the intersection points of the lines l_p and l_q with the vertical lines $x = 0$ and $x = L$ that bound R_L. The area of this trapezoid is $L(|\delta| + (|\epsilon L - \delta|)/2$. which is at most $L(|\epsilon|L/2 + |\delta|)$. This is maximized to $LD\sqrt{4 + L^2}/2$ when $\epsilon = LD/\sqrt{4 + L^2}, \delta = 2D/\sqrt{4 + L^2}$. Note that the area of this trapezoid is at least $A_L(l_p, l_q)$. Hence, $A_L(l_p, l_q) \le LD\sqrt{4 + L^2}/2$ from which the result follows.

□

We can now state our main result:

THEOREM 4.4. *Let A_n be an arrangement of n lines each of which intersects R_L and is dual to a point of R_L.*

i) For any $D > 0$, we can find an arrangement $A'_{n'}$ of n' lines in R_L such that $d_L(A_n, A'_{n'}) \le D$ and $n' \le n/2 + (4 + L^2)/(4D)^2$

ii) For any r such that $1/2 < r <= 1$, we can find an arrangement A'_{rn} such that $d_L(A_n, A'_{rn}) \le \sqrt{(L^2 + 4)/(16 * (r - 1/2)n)}$

PROOF. Let S_n be the set of points dual to the lines of the arrangement A_n. Let $D' = 2LD/\sqrt{4 + L^2}$ and observe that if the distance between two points is less than D', then the distance between their dual lines is less than D (by Theorem 4.3).

Using Theorem 4.2, we note that we can find a set $T_{n'}$ of n' points such that $d(S_n, T_{n'}) < D'$ and $n' \le n/2 + L^2/4D'^2$. Substituting for D' yields part i) of the theorem.

Part ii) of the theorem is proved by substituting rn for n' and solving the inequality $rn \le n/2 + (4 + L^2)/D^2$ for D.

□

These results show that one round of our algorithm is certain to give a good approximation. It remains an open problem to extend our theoretical underpinnings to support the empirical observations that applying our algorithm in multiple rounds continues to yield good approximations.

5. EXPERIMENTAL RESULTS

We implemented our algorithm in C on a Silicon Graphics workstation. To visualize the experiments and the results, we used the geometric animation system GASP [41]. We measured the approximations of arrangements and also applied these approximations to the problem of computing discrepancies [39].

APPROXIMATIONS

To test the quality of our algorithm, we created approximations for line arrangements generated by different random distributions [33], and created their dual lines according to the duality transform $Dual2$. This set of lines was the input to our algorithm.

To illustrate the quality our results, each of the Figures 2–6 presents the arrangement of a set of 50 lines generated with a specific distribution, and the approximating arrangement generated by a set of about half the number of lines, as produced by our algorithm. Typically the approximation would be run on larger inputs to yield better compression but the images from such runs are too cluttered to include in this document. In Figure 2 the input points were generated using a uniform distribution in the unit square. In

Figure 3 the input points were generated using a uniform distribution on the edge of a circle. In Figure 4 the input points were generated using a uniform distribution inside a circle. In Figure 5 the input points were generated using a multi-variant normal distribution at ten points in the unit square. In Figure 6 the input points were generated using a uniform distribution on the diameter of the unit square.

To evaluate the resulting approximating arrangements, we measured the Hausdorff distance of the arrangement from its approximation.

Table 1 summarizes some of our results or arrangements of up to $10,000$ lines. It can be seen that the distance from the original arrangement to the resulting one is very small even when the approximating arrangement is one tenth the size of the given arrangement. The exact distance depends on the type of distribution used to generate the lines. Moreover, as expected, as we allow more compression, the quality of the arrangement degrades, and the distance error grows.

No lines in L	No lines in L_1	Distribution	Distance
1000	750	Uniform	0.019436
1000	562	Uniform	0.028740
1000	315	Uniform	0.048049
1000	177	Uniform	0.082842
1000	99	Uniform	0.146336
5000	3750	Uniform	0.010350
5000	2812	Uniform	0.011549
5000	1581	Uniform	0.021419
5000	888	Uniform	0.035084
5000	499	Uniform	0.077760
10000	7500	Uniform	0.005839
10000	5625	Uniform	0.009475
10000	3163	Uniform	0.014833
10000	1779	Uniform	0.030319
10000	1000	Uniform	0.050047
1000	750	Annulus	0.000135
1000	562	Annulus	0.000392
1000	315	Annulus	0.001057
1000	177	Annulus	0.002391
1000	99	Annulus	0.003271
10000	7500	Annulus	0.000303
10000	5625	Annulus	0.000325
10000	3163	Annulus	0.000325
10000	1779	Annulus	0.000386
10000	1000	Annulus	0.000919
1000	750	Clusnorm	0.005101
1000	562	Clusnorm	0.020207
1000	315	Clusnorm	0.044425
1000	177	Clusnorm	0.139449
1000	99	Clusnorm	0.199697
10000	7500	Clusnorm	0.005402
10000	5625	Clusnorm	0.005839
10000	3163	Clusnorm	0.007716
10000	1779	Clusnorm	0.012288
10000	1000	Clusnorm	0.021373
1000	750	Ball	0.020309
1000	562	Ball	0.023880
1000	315	Ball	0.047370
1000	177	Ball	0.075209
1000	99	Ball	0.102761
10000	7500	Ball	0.007984
10000	5625	Ball	0.015156
10000	3163	Ball	0.016478
10000	1779	Ball	0.028706
10000	1000	Ball	0.046705

Table 1: Error Results

Application: Computing the Discrepancy

Supersampling is one of the most common approaches to the anti-aliasing problem in computer graphics. Since it has been shown that uniform sampling can lead to aliasing artifacts, a common approach has been to make use of the theory of discrepancy [39].

The discrepancy of a set S_n of n points is defined as follows. For a given line l, l divides the square into 2 parts, $R_+(l)$ and $R_-(l)$, with $R_+(l)$ (resp. $R_-(l)$) lying above (resp. below) the line l. In an ideal world, the fraction of the points of S_n that lie in region $R_+(l)$ would be exactly the area of $R_+(l)$. The discrepancy of S_n with respect to l is the absolute value of the difference between these quantities. The discrepancy of the point set S_n is measured as the maximum discrepancy of S_n with respect to l over all lines l.

Previously, we showed a method to compute the discrepancy of a set of n points in $O(n^2)$ [10, 9]. Building upon that algorithm, we now use the approximation techniques described in the current paper to reduce the complexity of arrangements by constant factors.

Table 2 describes our empirical results. We get significant speedups with relatively small errors. In particular, our behavior is best for uniform distributions, which are most common in this domain. For instance, for a uniform distribution (with 1000 points) we get a speedup by a factor of 116 while introducing errors of less than 2.5%, or a speedup by a factor of 205 while introducing errors of about 5.5%.

To further evaluate our method, we compared it to the method of merely using random sampling. In this method, we would choose at random a subsample of the original points and compute their discrepancy as an approximation to the actual discrepancy. This subsample had the same size that we got by our approximations scheme, so that the discrepancy computation algorithm ran in the same time for both approximations. The nature of discrepancy computations suggests that this should be a reasonable approach. We were pleasantly surprised to notice that our scheme works much better than random sampling for discrepancy. For instance, in cases where our scheme was off by 5%, this type of random sampling was off by 15% or 25% or more.

Visualization

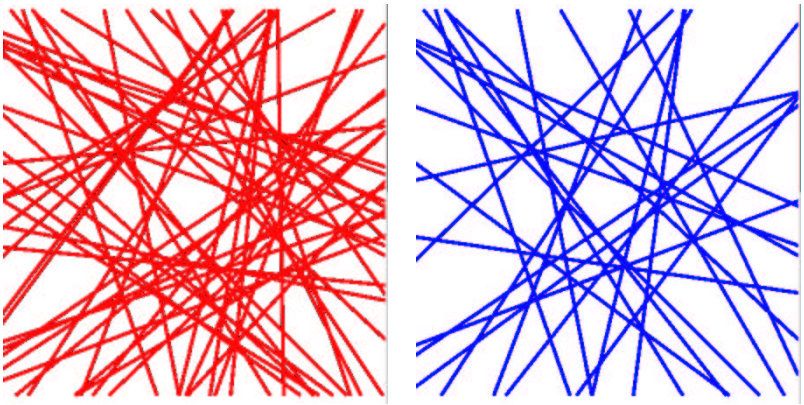
Visualization [41] played an important role in this research. While runs produce numbers that tell us how good a replacement scheme is, they do not help to understand where a scheme can be improved. Through visualization, the weaknesses of a replacement scheme came to light. Through visualization we found out that the intuitive way of replacing two lines by a line which is the dual of the midpoint (of the dual points), is not what we expected. In addition, with visualization we could indeed make our lines thicker and check that the approximating arrangement is indeed what we would get if we drew the original arrangement with thick lines. Finally, visualization helped us to check various duality transforms, and decide which ones fit our domain. The accompanying video [12] was developed along with the development of this research.

6. CONCLUSIONS

We described in this paper a technique for approximating arrangements of lines in the plane, so that the number of lines defining the arrangement is reduced by a constant fraction. We showed theoretical error bounds on our approach, as well as practical error bounds. We also showed how these ideas can be applied to compute discrepancies where our em-

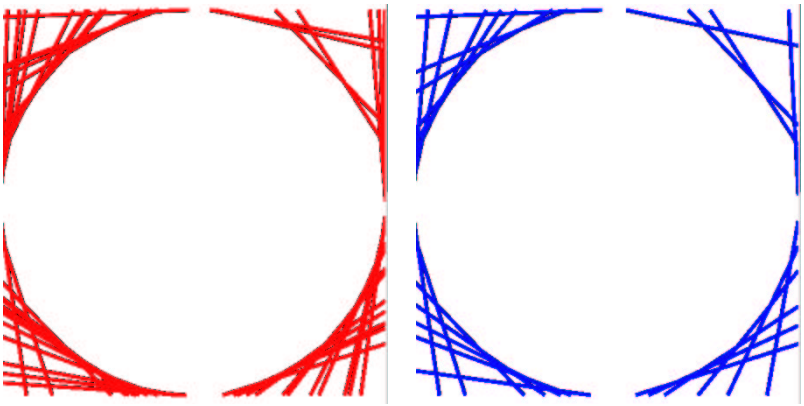
Distribution	Exact Discrepancy			Approximating Discrepancy			Error (%)	Speedup
	No. points	discrepancy	run-time	No. points	discrepancy	run-time		
Uniform	1000	0.251422	14.809	500	0.253156	3.455	0.689	4.286
Uniform	1000	0.252619	14.709	100	0.258625	0.126	2.377	116.738
Uniform	1000	0.251783	14.754	75	0.265734	0.072	5.540	204.916
Uniform	1000	0.252057	14.698	50	0.278013	0.034	10.297	432.294
Normal	1000	0.38605	14.671	500	0.37405	3.506	3.208	4.184
Normal	1000	0.38386	14.802	200	0.371838	0.507	3.233	29.19
Normal	1000	0.381412	14.797	150	0.363747	0.280	4.856	52.846
Normal	1000	0.390162	14.790	100	0.365415	0.125	6.772	118.32
Uniform	5000	0.25088	334.398	2500	0.250998	78.655	0.047	4.2514
Uniform	5000	0.250346	332.544	1000	0.251796	11.466	0.579	29.0
Uniform	5000	0.250538	335.438	500	0.254636	2.662	1.63568	126.0
Uniform	5000	0.250249	339.140	375	0.255739	1.466	2.1938	231.337
Uniform	5000	0.25019	332.904	250	0.255567	0.633	2.149	525.9
Normal	5000	0.379133	334.637	2500	0.368188	78.132	2.972	4.283
Normal	5000	0.380262	335.891	1000	0.358189	11.404	6.162389	29.453788
Normal	5000	0.379393	338.563	750	0.35601	6.291	6.568	53.817
Normal	5000	0.380474	332.526	500	0.357025	2.683	6.568	123.938

Table 2: Approximating Discrepancy



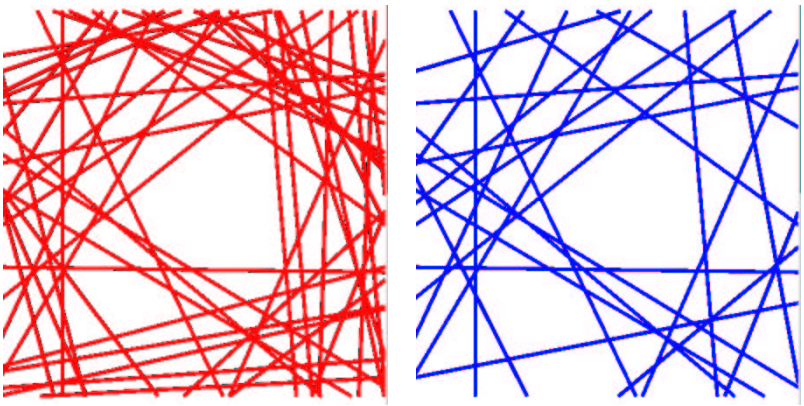
50 lines approximation with 24 lines

Figure 2: Uniform Distribution



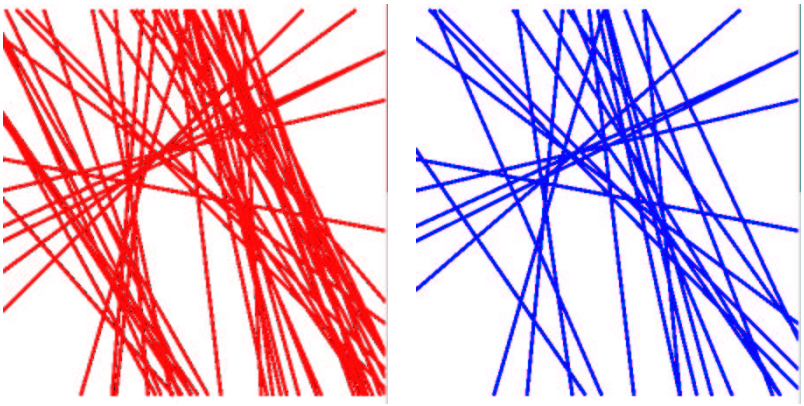
50 lines approximation with 24 lines

Figure 3: Annulus Distribution



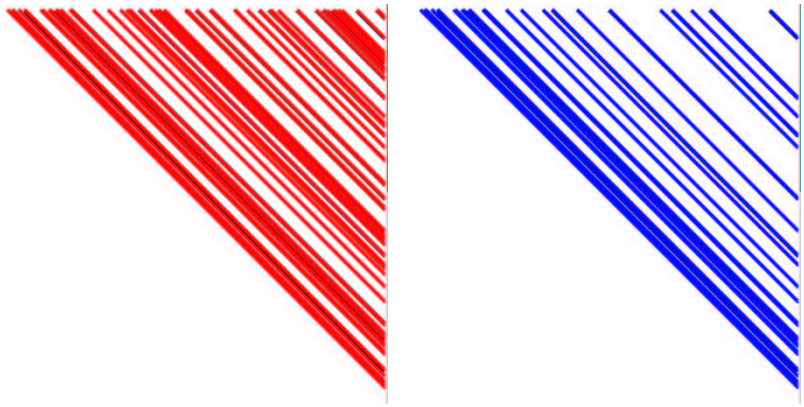
50 lines approximation with 24 lines

Figure 4: Ball Distribution



50 lines approximation with 24 lines

Figure 5: Clusnorm Distribution



50 lines approximation with 24 lines

Figure 6: Cubediam Distribution

pirical studies indicate that we can get speedups by factors of 200 while introducing errors of less than 5%.

Compression of arrangements has great potential in many applications, in addition to computing the discrepancy as discussed in the paper. Two other applications seem promising. The first is the visualization of huge amounts of data. Typically, we are willing to tolerate errors in such situations if the algorithm runs in real-time. We show here how our method applies to lines in the plane. We would like to check similar ideas for other objects in other domains. Another important application is quick and dirty path planning. In robotics it is usually the case that exact path planning is needed, and thus one needs to deduce a real path from the approximation. In computer graphics, however, small errors are often tolerable.

It would be easy to introduce a variable metric to our method. This would allow the user (or a program) to highlight regions (in dual or primal space) where less compression is desired.

We have made preliminary studies of this algorithm as a clustering algorithms [13]. The idea is to use the compressed set of lines as the starting point for some clustering algorithms (i.e., the k means), and see if the algorithm then converges faster. Our preliminary work here is promising. Further work is needed to fully demonstrate the applicability of this approach.

7. REFERENCES

[1] P. K. Agarwal and M. Sharir. On the number of views of polyhedral terrains. In *Proc. 5th Canad. Conf. Comput. Geom.*, pages 55–60, 1993.

[2] P. K. Agarwal and M. Sharir. Arrangements and their applications. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 49–119. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 2000.

[3] C. Bajaj, V. Pascucci, and G. Zhuang. Single resolution compression of arbitrary trianular meshes with properties. In *Proc. of the data compression conference.*, 1999.

[4] J. Barraquand, L. E. Kavraki, J.-C. Latombe, T.-Y. Li, R. Motwani, and P. Raghavan. A random sampling framework for path planning in large-dimensional configuration spaces. *Internat. J. Robot. Res.*

[5] B. Chazelle. Geometric discrepancy revisited. In *Proc. 34th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 392–399, 1993.

[6] H. S. M. Coxeter and S. L. Greitzer. *Geometry Revisited*. Mathematical Association of America, Washington, DC, 1967.

[7] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, Berlin, 1997.

[8] M. Deering. Geometry compression. *Comput. Graph.*, 13–20, 1995. Proc. SIGGRAPH '95.

[9] D. Dobkin and D. Eppstein. Computing the discrepancy. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 47–52, 1993.

[10] D. Dobkin and D. Mitchell. Random-edge discrepancy of supersampling patterns. In *Graphics Interface '93*, 1993.

[11] D. P. Dobkin, H. Edelsbrunner, and M. H. Overmars. Searching for empty convex polygons. *Algorithmica*, 5:561–571, 1990.

[12] D. P. Dobkin and A. Tal. Small representation of line arrangements. In *ACM Sympos. Comput. Geom. Video Review*, 2001.

[13] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley-Interscience, New York, 1973.

[14] M. Eck, T. Derose, T. Duchamp, H. Hoppe, and M. Lounsbery. Multiresolution analysis of arbitrary meshes. In *Proc. SIGGRAPH '95*, Computer Graphics Proceedings, Annual Conference Series, pages 173–182, July 1995.

[15] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*, volume 10 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, Heidelberg, West Germany, 1987.

[16] H. Edelsbrunner and L. J. Guibas. Topologically sweeping an arrangement. *J. Comput. Syst. Sci.*, 38:165–194, 1989. Corrigendum in 42 (1991), 249–251.

[17] Z. Gigus, J. Canny, and R. Seidel. Efficiently computing and representing aspect graphs of polyhedral objects. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(6):542–551, June 1991.

[18] J. E. Goodman and J. O'Rourke, editors. *Handbook of Discrete and Computational Geometry*. CRC Press LLC, Boca Raton, FL, 1997.

[19] L. Guibas, D. Halperin, H. Hirukawa, J.-C. Latombe, and R. Wilson. Polyhedral assembly partitioning using maximally covered cells in arrangements of convex polytopes. *Int. J. of Computational Geometry and its applications*, 8(2):179–199, 1998.

[20] D. Halperin. Robot motion planning and the single cell problem in arrangements. *Journal of Intelligent and Robotic Systems*, 11:45–65, 1994.

[21] D. Halperin, J.-C. Latombe, and R. Wilson. A general framework for assembly planning: The motion space approach. *Algorithmica*, 26:577–601, 2000.

[22] D. Halperin and M. Overmars. Spheres, molecules, and hidden surface removal. *Computational Geometry: Theory and Applications*, 11(2):83–102, 1998.

[23] P. S. Heckbert and M. Garland. Survey of polygonal surface simplification algorithms. Report, Carnegie Mellon University, 1997.

[24] M. L. Hilton, B. D. Jawerth, and A. Sengupta. Compressing still and moving images with wavelets. *Multimedia Systems*, 2(3), April 1994.

[25] D. P. Huttenlocher and K. Kedem. Computing the minimum Hausdorff distance for point sets under translation. In *Proc. 6th Annu. ACM Sympos. Comput. Geom.*, pages 340–349, 1990.

[26] M. Isenburg. Triangle fixer: Edge-based connectivity compression. In *Abstracts 16th European Workshop Comput. Geom.*, pages 18–23. Ben-Gurion University of the Negev, 2000.

[27] Z. Karni and C. Gotsman. Spectral compression of mesh geometry. In *Abstracts 16th European Workshop Comput. Geom.*, pages 27–30. Ben-Gurion University of the Negev, 2000.

[28] L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning

in high dimensional configuration spaces. *IEEE Trans. Robot. Autom.*, 12:566–580, 1996.

[29] A. Lazanas and J.-C. Latombe. Motion planning with uncertainty: A landmark approach. Technical Report, Dept. of Computer Science, Stanford University, 1992.

[30] A. W. F. Lee, W. Swelden, P. Schröder, L. Cowsar, and D. Dobkin. Maps: Multiresolution adaptive parameterization of surfaces. In *Proc. SIGGRAPH '98*, Computer Graphics Proceedings, Annual Conference Series, pages 95–104, July 1998.

[31] M. Lounsbery, T. D. DeRose, and J. Warren. Multiresolution analysis for surfaces of arbitrary topological type. *ACM Trans. Graph.*, 16(1):34–73, Jan. 1997.

[32] M. McKenna. Worst-case optimal hidden-surface removal. *ACM Trans. Graph.*, 6:19–28, 1987.

[33] M. Murphy and S. S. Skiena. Ranger: A tool for nearest neighbor search in high dimensions. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 403–404, 1993.

[34] M. Nelson. *The Data Compression Book*. M&T books, 2nd edition, Nov. 1995.

[35] J. O'Rourke. *Art Gallery Theorems and Algorithms*. The International Series of Monographs on Computer Science. Oxford University Press, New York, NY, 1987.

[36] J. O'Rourke. *Computational Geometry in C*. Cambridge University Press, 1st edition, 1994.

[37] W. B. Pennebaker and J. L. Mitchell. *JPEG - Still Image Data Compression Standards*. Van Nostrand Reinhold, 1993.

[38] M. Sharir and P. K. Agarwal. *Davenport-Schinzel Sequences and Their Geometric Applications*. Cambridge University Press, New York, 1995.

[39] P. Shirley. Discrepancy as a quality measure for sample distributions. In *Proc. Eurographics*, pages 183–193, 1991.

[40] G. Strang and T. Nguyen. *Wavelets and Filter Banks*. Wellesley-Cambridge Press, MA, 1996.

[41] A. Tal and D. P. Dobkin. Visualization of geometric algorithms. *IEEE Trans. Visualization and Computer Graphics*, 1(2):194–204, 1995.

[42] A. F. van der Stappen and M. H. Overmars. Motion planning amidst fat obstacles. In *Proc. 10th Annu. ACM Sympos. Comput. Geom.*, pages 31–40, 1994.

[43] M. Vetterli and J. Kovacevic. *Wavelets and Subband Coding*. Prentice Hall, NJ, 1995.