Shape Decomposition Using Farey Sequence and Saddle Points*

Sanjoy Pratihar
Computer Science and Engineering
University Institute of Technology
University of Burdwan, Burdwan, India
sanjoy.pratihar@gmail.com

Partha Bhowmick†
Computer Science and Engineering
Indian Institute of Technology, Kharagpur
India
bhowmick@gmail.com

ABSTRACT

A novel algorithm for boundary-based shape decomposition is proposed. The algorithm uses *Farey sequence* for determining several measures, such as slopes of edges and turn types at vertices of the polygonal cover P corresponding to the concerned shape. The *fraction ranks* (indices) in the Farey sequence have been used in the related procedures to enable computations in the integer domain while merging straight edges of P, which are almost collinear, and to perform *turn checking* at the vertices of P. Such turn checking aids in extracting the *saddle points* and constructing the *visibility graph* to finally output the decomposition. Experimental results demonstrate the efficiency and elegance of the proposed algorithm.

Keywords

Farey sequence, saddle points, shape decomposition

1. INTRODUCTION

Detection of meaningful components from a given shape is related to many contemporary paradigms like modeling [21], pattern recognition [13], shape analysis and retrieval [18], etc. Hence, several works on shape decomposition have come up over the last few decades. In [8], a technique based on erosion model has been suggested, which first extracts the shape contour from a silhouette image to compute an erosional vector representing a force of erosion at each concavity, and then obtains a shape-splitting boundary. Shape decomposition methods using morphology are also there. One such work [19] presents a recursive morphological operation in order to perform efficient shape representation. This operation uses a structuring element as a geometrical primitive to evaluate the shape of an object. Another work of shape decomposition based on mathematical morphology is presented in [12], which targets for object recognition and binary image coding. The work in [20] reports decomposition of 3D meshes in meaningful sub-components using the idea of "geodesic distance".

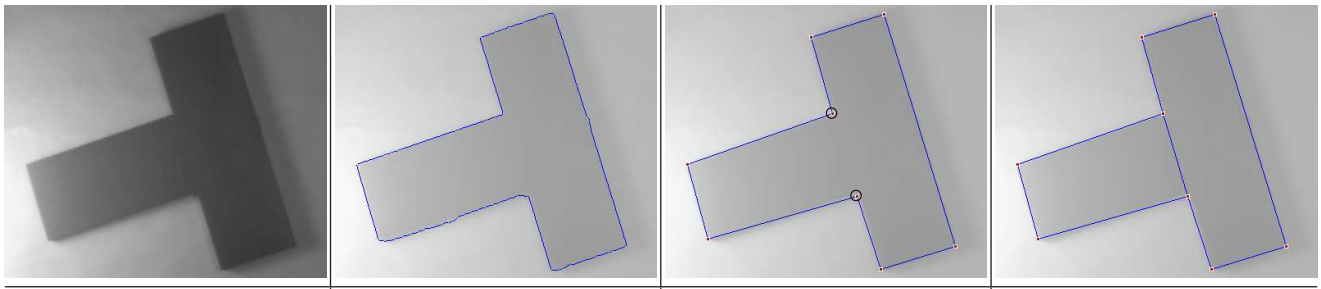
In this paper, we present a novel algorithm to find the *critical points* of a shape boundary obtained as a polygonal cover using the *Farey sequence* [5]. Some of these critical points are categorized as *saddle points*, and for their extraction, we have encoded the polygon boundary as a numeric sequence representing the *differential Farey indices*, which are newly introduced in this paper. A differential Farey index, being integer and in correspondence with the angular deviation, provides an efficient measure to obtain a fast polygonal cover of a digital contour, as shown in this paper. In particular, the differential Farey indices aid in adopting integer computation to find a polygonal cover, opposed to floating-point computation and allied procedural complexities in the existing methodologies. We have shown how a Farey sequence can be used to represent the edge slopes, and the vertex angles thereof, to describe a polygon for its readiness to subsequent applications. Once we get the Farey-based descriptor of a polygon, an efficient method is applied to detect the the *saddle points*, form which the components of resultant decomposition are obtained using the *visibility graph*. The major steps are as follows.

We first obtain the *digitally straight edges* defining the boundary of an object, and then merge the *almost collinear* edges to derive a tighter description of the object (Fig. 1b,c; Sec. 2). We extract each straight edge by inspecting the corresponding edge points one by one until there is a break because of the *digital straightness properties* [10]. The notion of *Farey sequence* [5] and the resultant *Farey indices*, which are in correspondence with the edge slopes, have been used to get the final polygonal cover (Fig. 1c). For an efficient implementation, the Farey indices are stored in the *augmented Farey table* ($\mathcal{F}$). Each Farey index corresponds to the slope of a line segment connecting two points with integer coordinates, and is available by a single probe in $\mathcal{F}$. The decision on collinearity of two or more straight edges is taken using addition/subtraction operations in the integer domain only. The Farey indices, corresponding to the (slopes of) edges of the polygonal cover, form a numeric sequence that serves as a high-level geometric abstraction of the underlying shape. We use this numeric sequence to obtain a *turn sequence*, which is a 2-element string consisting of L (left turn) and R (right turn) only. From this turn sequence, a sequence of *saddle points* and the corresponding L-containing substrings are obtained to form a *visibility graph*, which is finally used for component detection (Fig. 1d; Sec. 3).

*The work is carried out under APA project sponsored by DST (GoI), Ref. NRDMS/11/1586/2009.

†Corresponding author.

ICVGIP '10, December 12-15, 2010, Chennai, India



(a) Input shape. (b) Extracted straight edges, using *exponential averaging* and *digital straightness*. (c) Polygonal cover, using AFT ($N = 200, T_f = 5000$). (d) Final decomposition, using *saddle points* and *visibility graph*.

Figure 1: Basic steps of our proposed algorithm.

2. FORMING THE FAREY-BASED POLYGONAL COVER

To detect the straight edges of locally maximum lengths, we have used the properties of *digital straightness* [9, 10, 15] and *exponential averaging* [14] of edge strengths (Prewitt responses) [4]. A curve C is *digitally straight* if and only if its chain codes have at most two values in $\{0, 1, 2, \ldots, 7\}$, differing by $\pm 1 (\text{mod } 8)$, and for one of these, the run-length must be unity ([9]: Property R1). Also, if s and n be the respective singular code and non-singular code in a digital curve C, then the runs of n can have only two lengths, which are consecutive integers ([9]: Property R2).

To obtain the start point of a straight edge, each point p of the input (gray-scale) image is visited (in row-major order). If the Prewitt response at p exceeds the threshold value, T (= 100 in our experiments), and the response is a local maximum in the 8-neighborhood (8N) of p, then p is the start point, p_s. The next point on the edge commencing from p_s is obtained from the responses in 8N of p_s. The direction d_s from p_s is the chain code from p_s to its neighbor having the maximum response. In case of multiple maxima (which indicates multiple edges incident at p_s), we consider each of them, one by one, for finding the straight edges from p_s.

To get the (straight-)edge point next to any current point p, we need not apply the convolution at each neighbor (in 8N) of p with the Prewitt operator (in order to get their responses, and the maximum/maxima, thereof). Instead, in our algorithm, checking the Prewitt responses at three neighbors corresponding to three directions suffices: d, $(d+1) (\text{mod } 8)$, and $(d+7) (\text{mod } 8)$, where d is the chain code of p. For, from Property R1, no other neighbor can be the next point on the current edge. We have used an effective method of *exponential averaging* to estimate the strength of an edge point using its own response and the "exponentially weighted" contribution of responses at the previous edge points. In other words, to compute the exponential average of the responses in and around a point p, we consider the responses — which have been already computed and stored — at the points preceding p up the straight edge. A detailed explanation of the algorithm is given in [14].

2.1 Farey-based Integer Computation

Farey sequences are named after the British geologist, John Farey, Sr. [*Philosophical Magazine*, 1816][1]. Formally defined, the Farey sequence F_N of order N is the sequence of completely reduced (i.e., simple/irreducible), proper, and positive fractions that have denominators less than or equal to N, and are arranged in increasing order of their values. There are several studies and related works related with Farey sequences and their indices, some of which may be seen in [5, 7, 11, 17]. As shown in this work, a Farey sequence can be of interesting and practical use to decide whether three points, sorted lexicographically by their x and y (integer) coordinates, are collinear. It involves only addition, comparison, and memory access, but no multiplication, to check the linearity. Thus, it helps in reducing the running time for the linearity-checking function compared to the existing procedures.

For example, for three given points $p_1(i_1, j_1)$, $p_2(i_2, j_2)$, and $p_3(i_3, j_3)$ in succession, one of the common practices is to use the metric $\Delta(p_1, p_2, p_3) / \max(|i_1 - i_3|, |j_1 - j_3|)$ for computing the deviation of p_2 from $\overline{p_1 p_3}$ [2]. However, computation of the triangle area given by $\Delta(p_1, p_2, p_3)$ involves multiplication, and is therefore computationally expensive. Such multiplications are avoided by us using the Farey sequence. When we have a huge database of images to be processed one after another, we can compute a Farey table of an appropriate size and use it for computational optimization. As a result, the total time of polygonal approximation for all images in the database would be significantly reduced.

2.2 Augmented Farey Table

A Farey sequence starts with the fraction $\frac{0}{1}$ and ends with the fraction $\frac{1}{1}$. For example, the Farey sequences of orders 1 to 5 are as follows:

$F_1 = \left\langle \frac{0}{1}, \frac{1}{1} \right\rangle$,
$F_2 = \left\langle \frac{0}{1}, \frac{1}{2}, \frac{1}{1} \right\rangle$,
$F_3 = \left\langle \frac{0}{1}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{1}{1} \right\rangle$,
$F_4 = \left\langle \frac{0}{1}, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{3}{4}, \frac{1}{1} \right\rangle$,
$F_5 = \left\langle \frac{0}{1}, \frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \frac{2}{5}, \frac{1}{2}, \frac{3}{5}, \frac{2}{3}, \frac{3}{4}, \frac{4}{5}, \frac{1}{1} \right\rangle$.

Interestingly, each F_N can be computed from F_{N-1}. If $\frac{p}{q}$ has neighbors $\frac{a}{b}$ and $\frac{c}{d}$ in a Farey sequence, then $\frac{p}{q}$ is the

[1]He guessed that each new term in a Farey sequence is the mediant of its neighbors, but, so far as is known, he did not prove this property [1]. Later, Cauchy gave a proof in *Exercises de mathmatique* and attributed this result to Farey.

		Denominator								
		-4	-3	-2	-1	0	1	2	3	4
N	4	19	18	16	14	13	12	10	8	7
u	3	20	19	17	15	13	11	9	7	6
m	2	22	21	19	16	13	10	7	5	4
e	1	24	23	22	19	13	7	4	3	2
r	0	25	25	25	25	–	1	1	1	1
a	-1	26	27	28	31	37	43	46	47	48
t	-2	28	29	31	34	37	40	43	45	46
o	-3	30	31	33	35	37	39	41	43	44
r	-4	31	32	34	36	37	38	40	42	43

Figure 2: AFT $\mathcal{F}_4$ of order 4 containing indices of all fractions in $\{\frac{a}{b} : -4 \leq a, b \leq 4\}$.

mediant of $\frac{a}{b}$ and $\frac{c}{d}$. In other words, $\frac{p}{q}=\frac{a+c}{b+d}$.. For example, since $\frac{3}{5}$ is the mediant of $\frac{1}{2}$ and $\frac{2}{3}$ in F_5, $\frac{3}{5}$ is obtained for F_5 by adding the corresponding numerators and denominators of $\frac{1}{2}$ and $\frac{2}{3}$ from F_4.

Clearly, the original Farey sequence F_N of order N consists of all the simple, proper, positive fractions with denominators less than or equal to N. Compound fractions (that can be reduced to simple fractions of F_N), improper fractions (with numerators less than or equal to N), and negative fractions do not find any place in F_N. With simple operations, we obtain an *augmented Farey sequence* $\overline{F}_N$ from the Farey sequence F_N in order to include the above fractions as well. The augmented sequence $\overline{F}_N$ aids the linearity checking procedure while merging the end points of straight edges, which are almost collinear. For each member $\frac{a}{b}$ of F_N, we prepare a sub-list containing the equivalent compound fractions with denominators less than or equal to N. Corresponding to each $\frac{a}{b}$, a new fraction $\frac{a+a'}{b+b'}$ is computed, where $\frac{a'}{b'}$ is already a member of the sub-list corresponding to $\frac{a}{b}$, and b' is the highest denominator in the sub-list corresponding to $\frac{a}{b}$, such that $(b + b') \leq N$. This new fraction is kept in the sub-list of $\overline{F}_N$ linked to $\frac{a}{b}$. The first member $\frac{a'}{b'} = \frac{a+a}{b+b}$ of such a sub-list is obtained by adding the numerator a of $\frac{a}{b}$ with itself and the denominator b of it with itself, provided $2b' \leq N$.

Since $\overline{F}_N$ is derived as stated above, it contains all positive fractions (simple and compound) with denominators less than or equal to N. Now we take mirror reflection of this list about $\frac{1}{1}$, such that in the reflected part each member is the reciprocal of its counterpart. The compound fraction in the sub-lists linked to the simple fractions are also treated in the same way, i.e., numerators become denominators, and vice versa. This reflected part is appended to the original list. Next, we again take a reflection of this enlarged list, with the signs of all denominators in the reflected part changed to negative. Thus, finally we get all the fractions with positive numerator and positive/negative denominator. Taking their positions in the list we build the *augmented Farey table*, namely $\mathcal{F}_N$, corresponding to $\overline{F}_N$, as shown in Fig. 2. For example, when compound fractions are included in F_4, it gets augmented to

$\overline{F}_4 = \left\langle \frac{0}{1}, \left(\frac{0}{2}, \frac{0}{3}, \frac{0}{4}\right), \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, \left(\frac{2}{4}\right), \frac{2}{3}, \frac{3}{4}, \frac{1}{1}, \left(\frac{2}{2}, \frac{3}{3}, \frac{4}{4}\right) \right\rangle$.

On including the improper fractions, it is further augmented to

$\overline{F}_4 = \left\langle \frac{0}{1}, \left(\frac{0}{2}, \frac{0}{3}, \frac{0}{4}\right), \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, \left(\frac{2}{4}\right), \frac{2}{3}, \frac{3}{4}, \frac{1}{1}, \left(\frac{2}{2}, \frac{3}{3}, \frac{4}{4}\right), \frac{4}{3}, \frac{3}{2}, \frac{2}{1}, \left(\frac{4}{2}\right), \frac{3}{1}, \frac{4}{1}, \frac{1}{0}, \left(\frac{2}{0}, \frac{3}{0}, \frac{4}{0}\right) \right\rangle$.

When negative fractions (positive numerator and negative denominator) are included, we get

$\overline{F}_4 = \left\langle \frac{0}{1}, \left(\frac{0}{2}, \frac{0}{3}, \frac{0}{4}\right), \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, \left(\frac{2}{4}\right), \frac{2}{3}, \frac{3}{4}, \frac{1}{1}, \left(\frac{2}{2}, \frac{3}{3}, \frac{4}{4}\right), \frac{4}{3}, \frac{3}{2}, \frac{2}{1}, \left(\frac{4}{2}\right), \frac{3}{1}, \frac{4}{1}, \frac{1}{0}, \left(\frac{2}{0}, \frac{3}{0}, \frac{4}{0}\right), \frac{4}{-1}, \frac{3}{-1}, \frac{2}{-1}, \left(\frac{4}{-2}\right), \frac{3}{-2}, \frac{4}{-3}, \frac{1}{-1}, \left(\frac{2}{-2}, \frac{3}{-3}, \frac{4}{-4}\right), \frac{3}{-4}, \frac{2}{-3}, \frac{1}{-2}, \left(\frac{2}{-4}\right), \frac{1}{-3}, \frac{1}{-4}, \frac{0}{-1}, \left(\frac{0}{-2}, \frac{0}{-3}, \frac{0}{-4}\right) \right\rangle$.

For other two types of fractions, i.e., fractions with negative numerator and positive denominator, and fractions with negative numerator and negative denominator, we have to take reflection of the above list $\overline{F}_4$ and then change the signs of numerators and denominators accordingly. Thus, $\overline{F}_4$ becomes a complete list of all possible fractions. Clearly, all fractions $\frac{a}{b}$, where $|a| \leq 4$ and $|b| \leq 4$, can be kept in a list divided into four sub-lists, which keep fractions of types $\frac{a}{b}$, $\frac{a}{-b}$, $\frac{-a}{-b}$, and $\frac{-a}{b}$ respectively; and for each sub-list, the fractions are in increasing order. For each fraction in each sub-list, we can easily access its position in the sub-list by accessing the table $\mathcal{F}$, as shown in Fig. 2.

2.3 Farey-based Polygonal Cover

Extraction of straight edges from a gray-scale image generates an ordered set E (endpoints of straight edges), as explained earlier. Now, in order to reduce the number of straight edges defining the boundary of the object, vertices are taken out from E, and if they are "almost collinear", then they are combined together to form a longer straight edge. If $\langle e_i, e_{i+1}, \ldots, e_j \rangle$ be a maximal (ordered) subset of straight edges that are almost collinear, then these $j-i+1$ edges are combined to a single edge. The process is repeated for all such maximal subsets in succession to obtain a reduced set of (almost) straight edges corresponding to the object boundary. There are several techniques available in the literature to replace the almost-collinear pieces by a single piece [3, 16]. We have used a novel technique using differences of indices corresponding to edge slopes — which are equivalent to fractions — in the augmented Farey table, $\mathcal{F}$. Each $\mathcal{F}$-index is obtained by a single probe in $\mathcal{F}$ and the decision on linearity of three points is taken in the integer domain using addition/subtraction operation only. For a straight edge with end points $p := (x_p, y_p)$ and $q := (x_q, y_q)$, we do access the index of the fraction $\frac{y_p - y_q}{x_p - x_q}$, which is the slope of the line segment $\overline{pq}$, in $\mathcal{F}$. If two line segments L_1 and L_2 are having their respective $\mathcal{F}$-indices as f_1 and f_2, then L_1 and L_2 are merged if the difference of f_1 and f_2 is less than a threshold T_f, which is a differential Farey index and a parameter of our algorithm. Such a result for $T_f = 5000$ (order of Farey table being $N = 200$) is shown in Fig. 1.

3. SHAPE DECOMPOSITION

The shape of an object S is given by its polygonal cover P, which, in turn, is captured (partially) in the sequence of internal angles of P. For example, for a rectangular shape, the angular description will be $\langle 90^o, 90^o, 90^o, 90^o \rangle$ irrespective of its orientation; for an equilateral triangle, it will be $\langle 60^o, 60^o, 60^o \rangle$. In general, for a polygon having n vertices, there will be n entries in the description. The process involves computation of the angle magnitudes, which is ultimately based on multiplication and division in the real/floating-point domain.

For every edge of P, we need to compute its slope, and then the angle between each pair of consecutive edges. Instead of writing these angles in succession, we use the Farey indices to provide a description in the integer domain. Every straight edge of P corresponds to an index, namely the *Farey*

```
if (y_k ≥ y_{k+1} ≥ y_{k+2} or y_k ≤ y_{k+1} ≤ y_{k+2})
        if (f_k > f_{k+1}) then f'_{k+1} ← 2 · N_f − (f_k − f_{k+1})
        else f'_{k+1} ← 2 · N_f + (f_{k+1} − f_k)
else
    if |f_k − f_{k+1}| < 2 · N_f
        if (f_k > f_{k+1}) then f'_{k+1} ← 2 · N_f − (f_k − f_{k+1})
        else f'_{k+1} ← 2 · N_f + (f_{k+1} − f_k)
    else
        if (f_k > f_{k+1}) then f'_{k+1} ← f_k − f_{k+1}
        else f'_{k+1} ← 4 · N_f − (f_{k+1} − f_k)
```

Figure 3: Procedure for computing the differential Farey index, f'_{k+1}, for two consecutive edges, e_k and e_{k+1}.

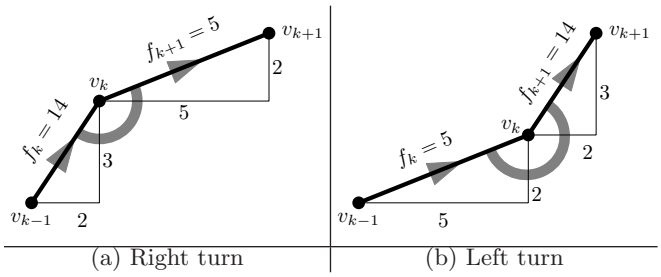


Figure 4: Examples of turn checking with $N = 5$: $\overline{F}_5 = \left\langle \frac{0}{1}, \frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \mathbf{\frac{2}{5}}, \frac{1}{2}, \frac{3}{5}, \frac{2}{3}, \frac{3}{4}, \frac{4}{5}, \frac{1}{1}, \frac{5}{4}, \frac{4}{3}, \mathbf{\frac{3}{2}}, \ldots \right\rangle$; $N_f = 11$. (a) Right turn, as $f'_{k+1} = 2N_f - (f_k - f_{k+1}) = 13 \leq 2N_f = 22$. (b) Left turn, as $f'_{k+1} = 2N_f + (f_{k+1} - f_k) = 27 > 2N_f = 22$.

index, which, for a "positive slope" of the edge, is the *rank*[2] of the fraction (equaling the slope of the edge) in F_N. For example, the rank of $\frac{3}{4}$ in F_4 is 6, since there are five smaller fractions ($\frac{0}{1}$, $\frac{1}{4}$, $\frac{1}{2}$, $\frac{2}{3}$, $\frac{3}{4}$) in F_4. For other fractions, the indices are computed in a convenient way (Fig. 2) and used for subsequent analysis, as explained in the following sections. Every two consecutive indices generate a difference, and the sequence of these differences is used as a (circular chain) description for the (closed) polygon, P.

3.1 Turn Checking Using $\mathcal{F}$-indices

The difference between the slopes of an edge e_k and of its next edge e_{k+1} is estimated as the difference of their Farey indices. We call it the *differential Farey index*. If the respective Farey indices of e_k and e_{k+1} be f_k and f_{k+1}, then the differential Farey index of e_{k+1} from e_k is realized using an *appropriate difference of f_{k+1} from f_k*. The procedure of computing these Farey differences has been given in Fig. 3. It is based on the analysis of various possible cases apropos the signs of the numerators and the denominators of f_k and f_{k+1}.

We consider the traversal of a polygon P such that the interior of P, and the corresponding object thereof, always lies right during the traversal. For three consecutive vertices of P, namely $v_{k-1} := (x_{k-1}, y_{k-1})$, $v_k := (x_k, y_k)$, $v_{k+1} := (x_{k+1}, y_{k+1})$, which define two consecutive edges e_k and e_{k+1} of P, we decide whether there is a left turn or a right turn at the vertex v_k, using the respective Farey indices f_k and f_{k+1} of e_k and e_{k+1} incident at v_k. Depending on a few combinatorial cases and their sub-cases, the differential Farey index, namely f'_{k+1}, corresponding to v_k, is hence obtained from f_k and f_{k+1}, as given in Fig. 3. If N_f denotes the number of positive, simple (proper or improper fractions) in the Farey sequence of order N (corresponding to slopes in $[0, \pi/2]$), then the total number of simple fractions in $\mathcal{F}_N$ is $4(N_f - 1)$, which increases with the order N of $\mathcal{F}_N$; for example, $N_f(\mathcal{F}_4) = 13$, whereby the total number of fractions in $\mathcal{F}_N$ is 48 (Fig. 2).

The turn checking at a vertex is done using its differential Farey index. There is a right turn at the vertex v_{k+1} if $f'_{k+1} \leq 2N_f$, and a left turn if $f'_{k+1} \geq 2N_f$, since $2N_f$ corresponds to the angular measure π. An illustration is given in Fig. 4 for a right and a left turn. For example, f'_{k+1} is computed as $f'_{k+1} = 2N_f - (f_k - f_{k+1})$ because $f_k > f_{k+1}$ (Fig. 3). So, $f'_{k+1} = 13$, which is less than $2N_f = 22$ (as $N_f = 11$ for order $N = 5$), which implies a right turn (Fig. 4(a)). On the contrary, in Fig. 4(b), as f'_{k+1} exceeds $2N_f = 22$, we have a left turn.

A demonstration of preparing the *turn sequence* for a polygon P by our algorithm on a test image is shown in Fig. 5. Starting at v_1, as we traverse the polygon in clockwise manner, the Farey indices of its edges $e_1, e_2, \ldots, e_{27}$ are computed from the vertex coordinates. For example, for edge e_1, the vertices are $v_1(33, 166)$ and $v_2(37, 252)$, wherefore we get its Farey index as $f_1 = 72827$ from the AFT, $\mathcal{F}_{200}$, with $N = 200$. As v_2 and $v_3(90, 225)$ correspond to the edge e_2, we get its index as $f_2 = 42701$. Using the procedure given in Fig. 3, we get $f'_1 = f_1 - f_2 = 30126$, which indicates a right turn at v_2, because $f'_1 \leq 2N_f = 48928$ (as $N_f(\mathcal{F}_{200}) = 24464$).

The feature sequence of the polygon P in Fig. 5 based on differential Farey indices is, therefore, given by

$P_{\mathcal{F}} = \langle 30126, 90019, 55100, 4039, 27982, 16817, 55908, 62684, 55554, 54164, 52740, 15277, 17948, 94673, 88106, 16817, 20359, 59928, 22382, 26464, 67016, 55729, 6005, 9998, 51744, 56564, 17165 \rangle$.

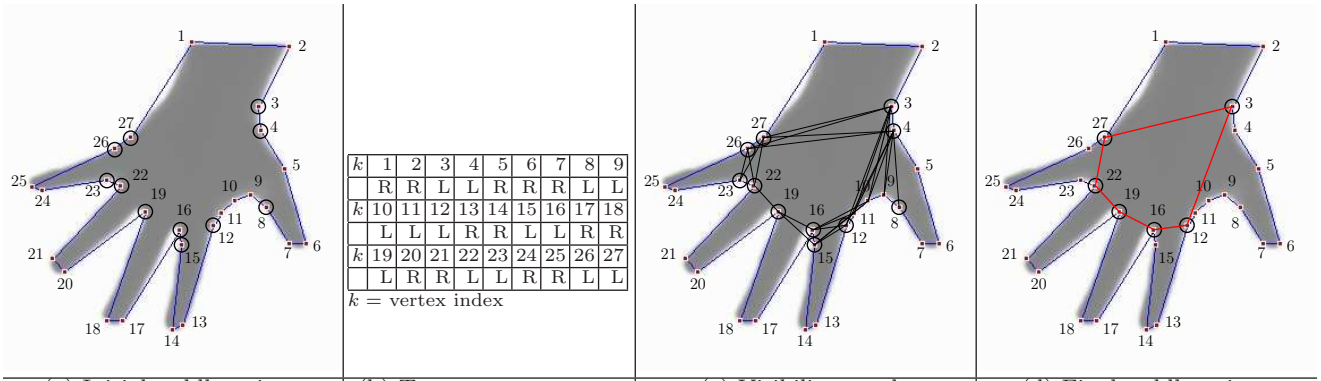
We enumerate the above sequence of differential Farey indices as a *turn sequence*, using the turn types at their vertices, namely L (left turn) and R. Such a turn sequence for the boundary of the shape of Fig. 5a has been shown in Fig. 5b.

3.2 Finding Saddle Points

It can be noticed from the turn sequence that a subsequence (i.e., a contiguous segment) of L represents a local concavity and that of R, a local convexity. In particular, the L-segments represent the concave parts, and the R-segments represent the convex parts. Further, the end of an L-segment (concave part) marks the beginning of an R-segment (convex part), and vice versa. We define a *saddle point* as the first or the last point of an L-segment,[3] before or after which an *R*-segment ends or starts. Evidently, in every L-segment there will be at least one and at most two *saddle* points. For, if an L-segment consists of a single L, then it marks the end of

[2]A simple, proper fraction $\frac{p}{q} \in F_N$ has a rank (i.e., index) f in F_N if and only if there exists $f - 1$ simple fractions in F_N whose values are less than $\frac{p}{q}$.

[3]Mathematically, a saddle point for a curve is a stationary point such that the curve in the neighborhood of that point is not entirely on any side of the tangent space at that point [6].



k	1	2	3	4	5	6	7	8	9
	R	R	L	L	R	R	R	L	L
k	10	11	12	13	14	15	16	17	18
	L	L	L	R	R	L	L	R	R
k	19	20	21	22	23	24	25	26	27
	L	R	R	L	L	R	R	L	L

k = vertex index

(a) Initial saddle points. (b) Turn sequence at the vertices $(v_1, v_2, \ldots, v_{27})$ of P. (c) Visibility graph. (d) Final saddle points.

Figure 5: Critical points (vertices), turn sequence, saddle points (encircled), and visibility graph resulting to the final set of saddle points, obtained by our algorithm.

its preceding R-segment and also the beginning of the next R-segment. And if it is made up of two or more Ls, then its first L marks the end of the preceding R-segment and its last L marks the beginning of the next R-segment. For example, in Fig. 5, $\langle v_3, v_4 \rangle$ is an L-segment; its first vertex v_3 signifies the end of the preceding R-segment $\langle v_1, v_2 \rangle$ and its last vertex v_4 signifies the start of the next R-segment $\langle v_5, v_6, v_7 \rangle$. Vertex v_{19} is the degenerate case of an L-segment with a single L, which marks the end of the R-segment $\langle v_{17}, v_{18} \rangle$ and the start of the R-segment $\langle v_{20}, v_{21} \rangle$.

3.3 Component Detection

After obtaining the sequence of saddle points, namely $S_P := \langle s_1, s_2, \ldots \rangle$, we construct the *visibility graph*, namely $G_P(V_P, E_P)$, defined on P. The graph G_P is constructed based on S_P for component detection as explained below (Fig. 5c), which is subsequently used for object decomposition. Observe that, each point $s_i \in S_P$ occurs before each other point $s_j \in S_P$ if and only if s_i is traversed before s_j while obtaining the vertex set of P.

To construct the node set V_P, for each $s_i \in S_P$, we consider the corresponding L-segment, S_i, from the turn sequence, and add a vertex v_i in V_P for each saddle point or critical point/vertex (of type L, if any) of S_i. For example, in Fig. 5c, the L-segments are denoted as $S_1, S_2, \ldots, S_6$, and for the L-segment $S_1 := \langle v_3, v_4 \rangle$, we have two saddle points considered as nodes in V_P; for $S_2 := \langle v_8, \ldots, v_{12} \rangle$, we have two saddle points (v_8 and v_{12}) and three other critical points (v_9, v_{10}, v_{11}) in V_P; and so forth. For the jth vertex (saddle point or critical point) $v_{ij} \in S_i$, which is added as a node u_{ij} in V_P, we also store the id i of the segment S_i as an auxiliary information of u_{ij} for construction of the edge set as follows.

For construction of the edge set E_P, we consider each pair of nodes $(u_{ij}, u_{i'j'})$ from V_P such that the following conditions are satisfied:

(e1) $i - i' = 1 \pmod m$, m being the total number of L-segments.

(e2) The line segment $l(v_{ij}, v_{i'j'})$ joining v_{ij} and $v_{i'j'}$ lies entirely inside P.

Condition e1 ensures that there is no edge between two saddle points or critical points belonging to the same L-segment (by this, we do not have any line segment lying outside P). Condition e2 is to guarantee that, after the decomposition, no part of any component lies outside the polygonal cover P of the concerned object. To decide whether $l(v_{ij}, v_{i'j'})$ lies entirely inside P, we consider each side/edge of P and find its intersection, if any, with $l(v_{ij}, v_{i'j'})$. The line segment $l(v_{ij}, v_{i'j'})$ is considered to be lying inside P if it has no point of intersection with any of the edges of P.

The components are extracted based on the visibility graph, G_P, as illustrated in Fig. 5c. The basic steps are as follows. We consider the first two L-segments, and check from G_P whether there exists an edge between the nodes corresponding to the first vertex of S_1 and the last vertex of S_2. If so, then the line segment joining these two vertices—one from S_1 and the other from S_2—is considered to be the first partition line of the decomposition (for example, the line segment connecting v_3 and v_{12} in Fig. 5d); and the part of the object contained in the sub-polygon starting from the first vertex of S_1, ending at the last vertex of S_2, and bounded by the concerned partition line is reported as the first component. Otherwise, we consider the second vertex of S_1 and the last (or its previous, in the next iteration) vertex of S_2, and so on, until there is a success. If there is a failure, then it implies that there exists no edge in G_P between any vertex of S_1 and any vertex of S_2; hence, no partition line is formed between S_1 and S_2, and the process is repeated between S_2 and S_3, starting from the first vertex of S_2 and the last one of S_3. In general, if a partition line is formed between jth vertex of S_i and j'th vertex of S_{i+1}, then in the next iteration, we consider j'th vertex of S_{i+1} as its resultant first vertex and check whether edges exist between it and the last vertex of S_{i+2}. If yes, then the corresponding partition line is formed. Otherwise, we consider the $(j'+1)$th vertex of S_{i+1} and do a similar checking until we get a success or all the vertices of S_{i+1} and S_{i+2} get exhausted. If it is a success, then the vertex of S_{i+2} incident on the partition line is set as the resultant first vertex in the next iteration; otherwise, the actual first vertex of S_{i+2} remains the first vertex in the next iteration. The procedure is repeated until all the

Table 1: Summary of results for some images.

Image	Straight Edges	n	Components	CPU Time (seconds)
t001	77	39	6	0.0289
t002	87	72	8	0.0419
t003	76	48	7	0.0372
bird	64	32	7	0.0256
camel-1	111	69	11	0.0418
camel-2	84	53	10	0.0389
hammer	20	13	4	0.0277
jar	45	29	5	0.0249
palm-1	55	25	7	0.0324
stem	37	28	7	0.0283

L-segments are considered for forming the partition lines in succession. The red-colored lines in Fig. 5d illustrate the final partition lines, as reported by our algorithm.

4. IMPLEMENTATION AND RESULTS

We have implemented the algorithm in C in Linux Fedora Release 7, Kernel version 2.6.21.1.3194.fc7, Dual Intel Xeon Processor 2.8 GHz, 800 MHz FSB. To reduce the number of edges by merging the "almost collinear" edges, we have considered T_f as the Farey threshold and have tested on several datasets for various T_f. If two consecutive edges e and e' are having AFT indices f and f', then they will be almost collinear if $|f - f'|$ is appreciably small, or, $|f - f'|$ is less than the threshold, T_f. Thus, the threshold T_f realizes the tolerance of merging two or more edges in succession, where the value of T_f, in turn, depends on the order N of $\mathcal{F}_N$.

As T_f increases for a fixed AFT, the number of straight edges in P gets reduced. For example, when the order $N = 100$, total number of fractions is 24352. These 24352 slopes/fractions divide the interval of $[0, 360^o)$ into 24352 divisions. Similarly, for $N = 200$, there exist $24464 \times 4 = 97856$ slope vectors, wherefore each division amounts to 0.0037^o. Hence $T_f = 1000$, 2000, and 4000, for $N = 200$, provides tolerances of 3.679^o, 7.356^o, and 14.716^o respectively. Although theoretically all the divisions will not be equal, the impact on practical applications is relatively negligible when the number of divisions is significantly large. Hence, in our implementation, we have taken $N = 200$. Table 1 shows our test results for a few shapes whose final decompositions are shown in Fig. 6.

5. CONCLUSION

Shape decomposition is a pertinent problem in content-based image retrieval and computer vision. The proposed algorithm is an attempt to solve the problem in a novel manner using the classical concept of Farey sequence with an efficient implementation of augmented Farey table (AFT) for computation of differential Farey indices. These indices, in turn, aid in a fast and elegant procedure of generating the turn sequence, and a sequence of saddle points thereof, corresponding to the vertex sequence of the polygonal cover of the underlying object. The visibility graph based on the turn sequence and saddle points, helps in deriving a successful decomposition of various digital objects. A faster construction of the visibility graph is a prospective option related with this work, and the possibility of obtaining alternative solutions of decomposition using non-successive saddle points is also a challenging issue, which would be addressed in future.

6. REFERENCES

[1] A. H. Beiler. *Recreations in the Theory of Numbers: The Queen of Mathematics Entertains*. Dover, 1964.

[2] M. D. Berg, M. V. Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry Algorithms and Applications*. Springer-Verlag, Berlin, 2000.

[3] P. Bhowmick and B. B. Bhattacharya. Fast polygonal approximation of digital curves using relaxed straightness properties. *IEEE Trans. PAMI*, 29(9):1590–1602, 2007.

[4] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley, California, 1993.

[5] R. Graham, D. Knuth, and O. Patashnik. *Concrete Mathematics*. Addison-Wesley, London, UK, 1994.

[6] L.F. Gray *et al.* Calculus Two: Linear and nonlinear functions, Springer-Verlag, 1990.

[7] G. H. Hardy and E. M. Wright. *An Introduction to the Theory of Numbers*. Oxford University Press, New York, 1968.

[8] F. Kanehara, S. Satoh and T. Hamada. Shape decomposition based on erosion model. *Proceedings of the Workshop on Physics-Based Modeling in Computer Vision*, 1995.

[9] R. Klette and A. Rosenfeld. *Digital Geometry: Geometric Methods for Digital Picture Analysis*. Morgan Kaufmann, 2004.

[10] R. Klette and A. Rosenfeld. Digital straightness: A review. *Discrete Applied Mathematics*, 139(1-3):197–230, 2004.

[11] E. H. Neville. *The Farey Series of Order 1025*. Cambridge University Press, Cambridge, 1950.

[12] I. Pitas and A. N. Venetsanopoulos. Morphological shape decomposition. *IEEE Trans. PAMI*, 12:38–45, 1990.

[13] I. Pitas and N. D. Sidiropoulos. Pattern recognition of binary image objects using morphological shape decomposition. *Proceedings of CVIP*, 279–305, 1992.

[14] S. Pratihar and P. Bhowmick. A thinning-free algorithm for straight edge detection in a gray-scale image. In *Proc. 7th Intl. Conf. on Advances in Pattern Recognition (ICAPR)*, pages 341–344, 2009.

[15] A. Rosenfeld. Digital straight line segments. *IEEE Trans. Computers*, 23(12):1264–1268, 1974.

[16] P. L. Rosin and G. A. W. West. Non-parametric segmentation of curves into various representations. *IEEE Trans. PAMI*, 17:1140–1153, 1995.

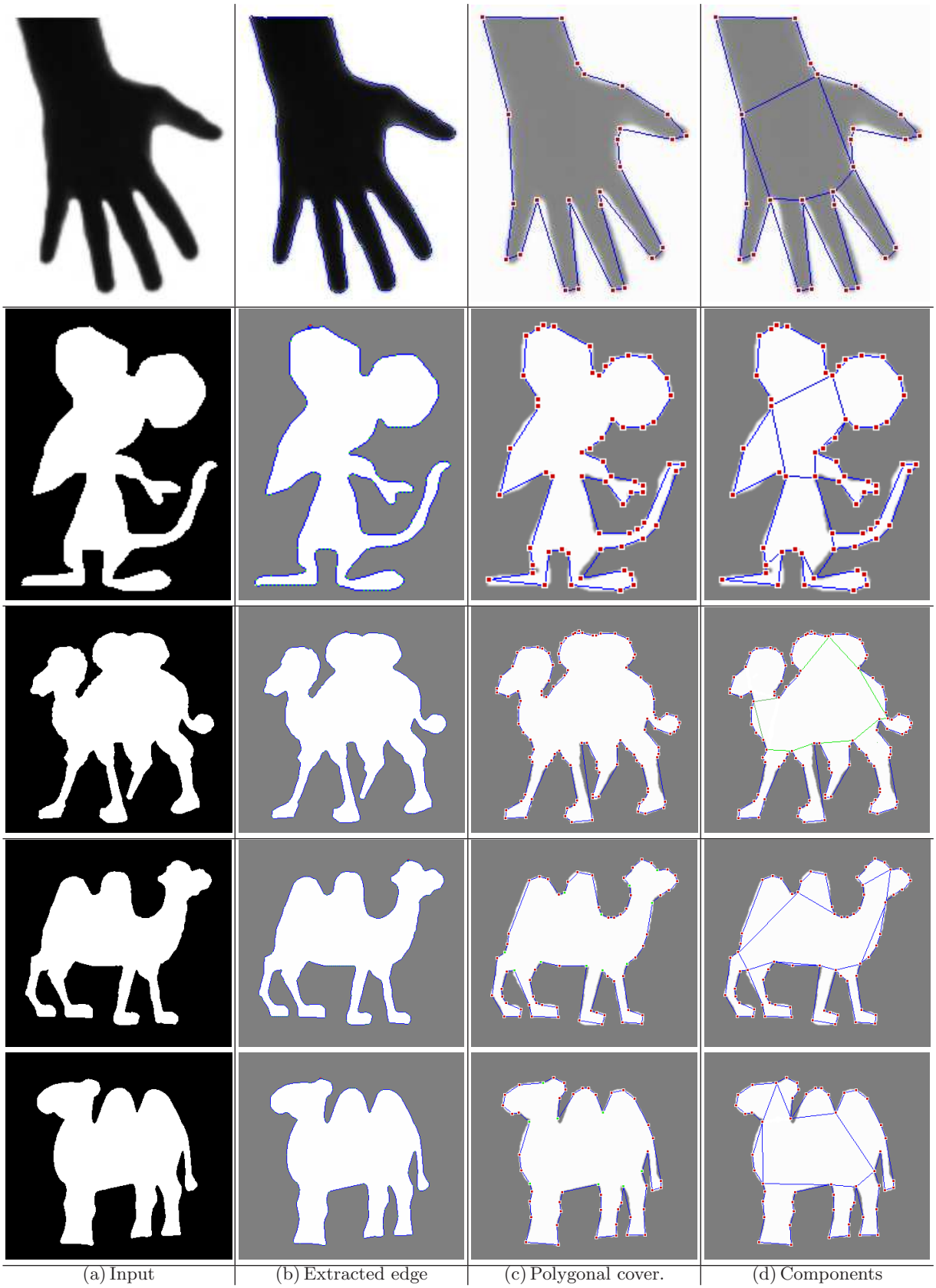
[17] M. Schroeder. Fractions: Continued, Egyptian and Farey (Chapter 5). In *Number Theory in Science and Communication*. Springer Series in Information Sciences, Vol. 7, 2006.

[18] A. Shokoufandeh, L. Bretzner, D. Macrini, C.J. Demirci and S. Dickinson The representation and matching of categorial shape. *Computer Vision and Image Understanding*,103, 139-154,2006.

[19] D. Wang, V. Haese-Coat and J. Ronsin. Shape decomposition and representation using a recursive morphological operation. *Pattern Recognition*, 28(11):1783–1792, 1995.

[20] Y. Zhou and Z. Huang. Decomposing polygon meshes by means of critical points. *Proc. 10th International Multimedia Modelling Conference*,2004.

[21] S. C. Zhu and A. L. Yuile. Forms: A flexible object recognition and modelling system. *International Joural of Computer Vision*, 20, 187–212, 1996.



(a) Input | (b) Extracted edge | (c) Polygonal cover. | (d) Components

Figure 6. (Continued to next page)

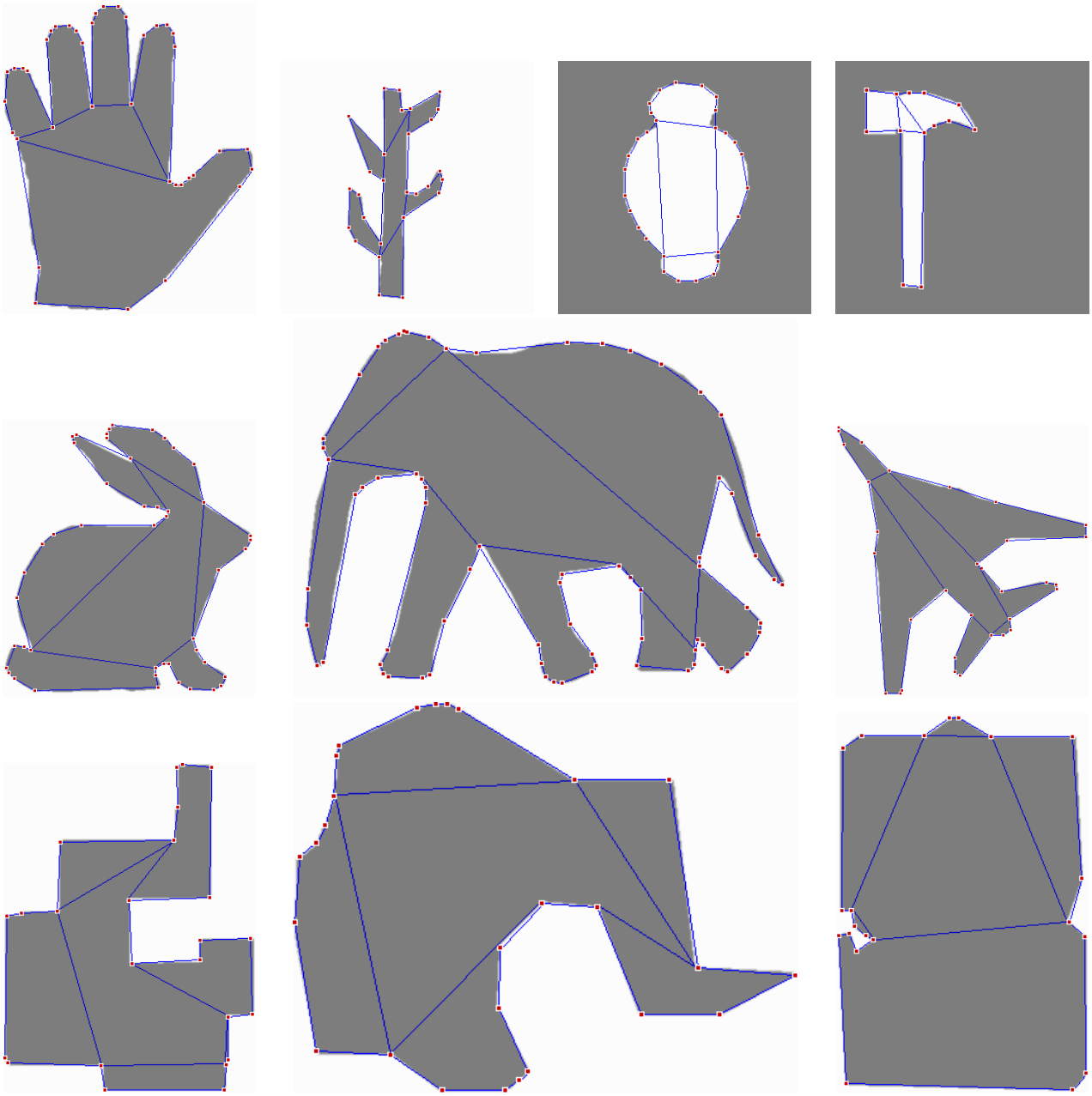


Figure 6: Results of decomposition by our algorithm ($N = 200, T_f = 5000$).