Automated building simplification using recursive approach

Tomáš Bayer

Department of Applied Geoinformatics and Cartography, Faculty of Science,
Charles University in Prague, Albertov 6, Praha, Czech Republic.
bayertom@natur.cuni.cz

Abstract

Automated or semi automated simplification of buildings is a problem, that is currently being solved in many ways. This article presents a new generalization algorithm for automated or semi automated buildings simplification based on the recursive approach. From the data set with a higher level of detail another data set that contains a lower level of detail can be derived. Proposed algorithm has an ability to detect and simplify buildings rotated in random position and does not depend on ordering of points. To determine the angle of rotation φ of the building a smallest area enclosing rectangle is constructed. The splitting procedure is performed on the basis of splitting criterion σ calculated for each edge of the building. The simplified edges are replaced with regression lines, whose parameters are determined using the least squares method. The algorithm was implemented in C++. With the use of ArcObjects libraries, a new DLL application designed for ArcGIS 9.x was created. This tool is applicable to the polygonal shape files.

Keywords: automated simplification, buildings, generalization, recursion, algorithm

1 Introduction

An automated or semi automated simplification of buildings based on the least square method is currently being solved in many ways, eg [1], [2]. This article brings first information about the new buildings simplification algorithm based on the recursive approach. The presented algorithm was implemented in C++ and it is available as DLL application designed for ArcGIS 9.x

Geometric generalization carries out a controlled reduction of the map content based on the analysis of the geometric properties of elements. It tries to remove those elements, that are not significant in the map context. An algoritmization of the simplification process is not unambiguous. It is not an easy task to find and set a geometric criterion, that is supposed to be satisfied by a simplified element. The simplification represents a process of more interdependent steps, an implementation of one step causes the next step. During the simplification process such edges, that are not significant in the map context, are removed. Commonly used simplifying algorithms can not be applied, they do not maintain internal angles ($\pm\frac{\pi}{2}$), formed by polygon edges representing the building, see Fig. 1. Building simplification has some constrains that make this process more difficult.

2 Basic concept

To maintain the basic characteristics of cartographic outputs, a controlled reduction of information must be performed. Let us consider a non convex rectangular polygon in the plane to be a building. This polygon is bounded by a finite collection of n line segments. Each segment e_i has two vertices $P_i, P_{i+1}, i \in \langle 1, n-1 \rangle$. Let $P_1, P_2, ..., P_n$ be n points representing vertices of the polygon and e_i, e_{i+1} two adjacent and perpendicular line segments. The building usually does not have to be oriented in the "basic position", when all edges are parallel to axis x, y of the coordinate system. In general, the position of the building is rotated. The angle of rotation φ must be detected as a first step of the simplification algorithm.

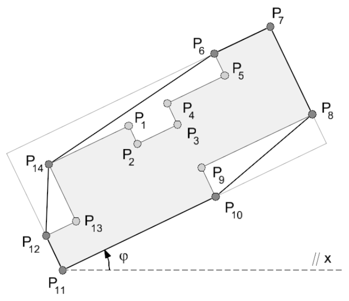


Figure 1: Determination of the building rotation using the smallest area enclosing rectangle.

For an assessment of the complexity of the edge form a criterion based on the standard deviation σ is used. This criterion is calculated repeatedly for each detected edge of the building. It is compared with a maximum value of the criterion σ_{max}. Value of the criterion depends on the complexity of the edge form, for more complex forms becomes of greater values. Until $\sigma > \sigma_{max}$, each edge is recursively decomposed to the sets of new edges. Simplified edges are replaced with the regression lines, whose parameters are determined using the least squares method.

2.1 Scheme of the simplification process

The simplification process can be shortly described using the following steps:

1. Detection of the angle of rotation φ of the building: (a)construction of the convex hull of a set of points, (b) construction of the smallest area enclosing rectangle of a set of points.
2. Set rotation of the building: the angle of rotation $-\varphi$.
3. Detection of the vertices and edges of the building based on the recursion: (a) calculation of the splitting criterion σ, (b) recursive decomposition of the edge to the set of new edges.
4. Set rotation of the building: angle of rotation φ.

In order to simplify mathematical calculations, generalized building is rotated by the angle of $-\varphi$. The building is rotated so that its edges are parallel to the axes of x and y. In this position, all steps of the simplification process are carried out. Finally, the simplified building is rotated to the starting position by the angle of φ.

3 Simplification algorithm

3.1 Building rotation

An accuracy of determining the angle of rotation φ significantly affects an effectiveness of the algorithm. The most common method of detecting the angle of rotation φ formed by x axis and the longer edge of the rectangle, is based on construction of the minimum bounding box (rectangle enclosing all points with the minimum area), see Fig. 1. The procedure runs over a non-convex polygon and makes the process more difficult. Commonly available algorithms achieve quadratic time complexity $O(N^2)$ for this operation. Using rotating calipers we can perform this step in linear time. However, this procedure is usable for convex polygons. The first step brings transformation of the non-convex polygon on the convex hull.

The presented solution described in [5] solves the problem using two calipers orthogonal to each other in linear time. An idea of construction is based on the repeated rotation of the rectangle. This rectangle is gradually improved and becomes an approximation of the smallest area enclosing rectangle in the next step. At least one edge of the smallest area enclosing rectangle must be collinear with at least one segment of the convex hull. Let us denote φ_j, $j \in \langle 1, 4 \rangle$, four angles formed by the four smallest area enclosing box edges and four edges of the convex hull in points of contact

V_j. Let V_j' represents a point, that is a successor of the point V_j, and M_j represents a vertex of the smallest area enclosing box. Vertices of the smallest area (and thus edges) are clockwise oriented.

We find the minimum angle $\varphi_{min} = \min(\varphi_j)$ and rotate the rectangle by an angle φ_{min}. Another edge of the rectangle becomes collinear with some segment of the convex hull. Three points of contacts will not change. However one point V_j, represented by the start point of the collinear segments, changes to its successor V_j'. We calculate an area S of the rectangle, compare it with a minimum area S_{min} initialized during the first step to ∞. If $S < S_{min}$, we store $S_{min} = S$. Repeat those steps until $\sum \varphi_{min} < \frac{\pi}{2}$ leads to result $\sum \varphi_{min} = \varphi$.

3.2 Detection of vertices

The presented algorithm is based on the idea of hierarchical detection of vertices. Vertices are detected on the basis of their geometric significance. New vertices are gradually added to the building, among them edges of the building are geometrically reconstructed.

We search for such pairs or four points, that are located closest to the vertices of the smallest area enclosing rectangle, constructed over every edge [2]. Those points will represent new vertices of the building in the next step. All points P_i, which are located "between" two vertices, are matched all in all to the new edge. Each detected edge of the buildings will be replaced by several "U" and "L" segments. "L" segment is formed by the pair of new perpendicular edges, "U" segment is formed by three new perpendicular edges. Those edges will be processed using recursive approach in the same way.

Recursive approach of founding of vertices. As mentioned above, we will consider a non convex rectangular polygon in the plane to be a building. Each segment e of the building will be parallel to one edge of the smallest area enclosing rectangle. If some detected edge of the building is *parallel* and has the *same direction* (ie it is same oriented) to the other edge of the smallest area enclosing rectangle than the first edge (formed by vertices M_1, M_2), we formally rearrange ordering of the smallest area enclosing box vertices M_j. Let M_1 represents a point with coordinates $[\min(x_i), \max(y_j)]$. As mentioned above, vertices M_1, M_2, M_3, M_4 and edges e_1, e_2, e_3, e_4 of the enclosing rectangle are *clockwise ordered.*

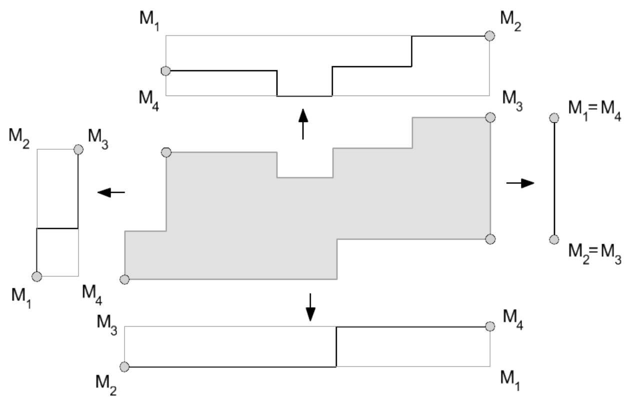


Figure 2: Smallest area enclosing rectangle for edges with rearrangement of the vertices.

Depending on which edge of the smallest area rectangle has the same orientation as the edge of the building, a set of vertices $\{M_1, M_2, M_3, M_4\}$ is rearranged to another set: $\{M_2, M_3, M_4, M_1\}$ for the same orientation as a second edge, or $\{M_3, M_4, M_1, M_2\}$ for the same orientation as a third edge, or $\{M_4, M_1, M_2, M_3\}$ for the same orientation as a fourth edge. For example, in the second case, the first vertex M_1 of rearranged rectangle corresponds with the vertex M_3 of non rearranged rectangle, and so on, see Fig. 2.

This problem is easily solvable with the recursive approach changing an ordering of formal parameters M_j. Let N_j, $N \in P$, represent four *vertices* of the building closest to points M_j. Then point N_1 is the closest to the point M_1, etc.

Orientation of edges and vertices. Algorithm keeps an information about the orientation of each edge of the building and the edge of the smallest area enclosing rectangle at any moment. Let g,$g = 1, 2, 3, 4$, represents an orientation of the building segment e to corresponding edge of the enclosing rectangle. As mentioned above, edges of the enclosing rectangle are *clockwise* arranged. Orientation g of the segment e depends on the orientation of its first point N denoted as $g(N)$. This orientation is subsequently set for all points P_i lying within the segment e. Let $g^{(r)}$ represents an orientation of the edge in the depth of recursion r, $g^{(r-1)}$ represents an orientation of the edge in the last recursion step $r - 1$. The relationship $g^{(r)}(N)$ to $g^{(r-1)}(N)$

$$g^{(r)}(N) = g^{(r-1)}(N) + j. \tag{1}$$

An orientation of the segment in any recursion depth r to corresponding edge of the smallest area enclosing rectangle, constructed over all points P (first recursion depth, $r = 1$)

$$g^{(1)}(N) = g^{(r)}(N)mod(4) - (r - 1)mod(4). \tag{2}$$

If

$$g^{(1)}(N) \begin{cases} = 0, & g^{(1)}(N) = 4, \\ \neq 0, & g^{(1)}(N) = g^{(1)}(N). \end{cases}$$

Formula (2) is calculated for each detected edge. For the sample of values $g^{(r)}$ in recursion depths $r = 1, 2, 3, 4$, see Tab. 1.

Table 1: Values $g^{(r)}$ for recursion depths $r = 1, 2, 3, 4$.

$g^{(1)}(N)$	$g^{(2)}(N)$		$g^{(3)}(N)$			$g^{(4)}(N)$			
1	2	6	3	7	11	4	8	12	16
2	3	7	4	8	12	5	9	13	
3	4	8	5	9		6	10	14	
4	5		6	10		7	11	15	

The proposed algorithm performs recursive splitting of the edge depending on its form complexity. The splitting criterion σ is calculated for each detected edge of the building. It is based on the calculation of standard deviation σ, that minimize the sum of the squares of the points distances from the regression line; the variance of the residuals is the minimum possible. The regression line is oriented in accordance with one edge of the smallest area enclosing rectangle. In this case, there is no need to determine the angle of the regression line. The regression line also passes through the center of gravity of the set of points. For a construction of the regression line, it is necessary to know the value of $g^{(1)}$. Depending on the value of $g^{(1)}$ the splitting criterion could be rewritten as

$$g^{(1)} = \begin{cases} 1, 3 & \sigma = \sqrt{\frac{\sum_{i=1}^{n}(y-y_T)^2}{n}}, \\ 2, 4 & \sigma = \sqrt{\frac{\sum_{i=1}^{n}(x-x_T)^2}{n}}, \end{cases} \tag{3}$$

where n represents number of points of the simplified segment e and $T = [x_T, y_T]$ is the center of gravity. In the first case, σ is the only function of the variable y, in the second case only function of the variable x. For each edge the coordinates x_T, y_T of the center of gravity usable to its further reconstruction, are stored. An interesting opportunity for further research brings adding the dependency on the map scale to (3).

3.3 Building reconstruction.

The reconstruction of the simplified building is carried out from parameters of regression lines. Edges are stored in the stack, therefore two following edges poped from the stack are always perpendicular. The reconstruction process runs in a very simple way.

1. Initialize $i = 1$. Get edge e from the top of the stack S.

2. Store coordinates x_T, y_T of the edge e as $x_T^{old} = x_T$, $y_T^{old} = y$.
3. Using (2) determine its orientation $g^{(1)}$. Let B_i is a new edge point of the simplified building. If $i > 1$

$$g^{(1)}(N) = \begin{cases} 1,3: & B_i = [x_T^{old}, y_T]. \\ 2,4: & B_i = [x_T, y_T^{old}]. \end{cases}$$

4. Add B_i to the building.
5. Go to (1) until we reach end of the stack.

4 Building simplification process

The simplification process represents an application of the recursive approach. It consists of several steps, that are repeated until $\sigma > \sigma_{max}$. The value σ_{max} could be set by a user and indicates a simplification ratio.

4.1 Processing edges

A procedure performing the recursive splitting of each edge has two parameters. The first one, σ_{max}, represents the simplification tolerance, and the second one, enables or disables splitting of the edge formed by two points. The splitting procedure can be described as follows:

While stack S not empty:

1. Pop an edge e from the stack S.
2. Get $g^{(1)}$ and calculate σ for the edge e.
3. If e represents initial edge, split e into four new edges e_1, e_2, e_3, e_4 and push them into stack S.
4. Else if $\sigma > \sigma_{max}$: Replace e by several "U" and "L" segments formed by set of two or three new edges. Push those edges into stack S.
5. Else replace the edge e with the regression line.

The first step of the simplification algorithm replaces the polygon with the rectangle formed by four edges. These four edges are subsequently pushed into a stack. An edge on the top is removed from the stack and splitted into several "U" and "L" segments, if its form is considered as too complex. So, one edge is replaced by more new edges. However edges of the simple form are not splitted but sent to the output. Due to the recursion it is quite difficult to determine a time complexity of the proposed algorithm.

4.2 Recursive splitting of the initial edge

Splitting of the initial edge e formed by the polygon, into four new edges e_1, e_2, e_3, e_4 is simple. During this step four points N_j closest to four vertices M_j of the smallest area enclosed rectangle, constructed over this edge, are found. Each point N_j corresponds to point P_i , N_j is pointer to P_i. The algorithm tests, if these points $P_i \approx N_j$ are clockwise ordered in polygon. If there is no such ordered set of points, the splitting procedure is stopped and building is replaced by an rectangle. This situation is becoming very rarely.

The algorithm keeps an information about the orientation of the vertices $g^{(r)}$ in each recursion depth r calculated from (2). As mentioned above, an orientation $g^{(r)}$ is subsequently set for all points P_i lying within the segment e. During this step it is not necessary to determine the criterion σ, an initial edge is divided into four new edges in any case. All points belonging to one edge have the same orientation. Presented method has the advantage, that orientation of each edge can be determined from the orientation of its first point. Each vertex of an initial edge becomes a start point of one new edge, and at the same time an end point of one new edge adjacent to previous edge. Four new edges e_1, e_2, e_3, e_4 are subsequently added to the stack.

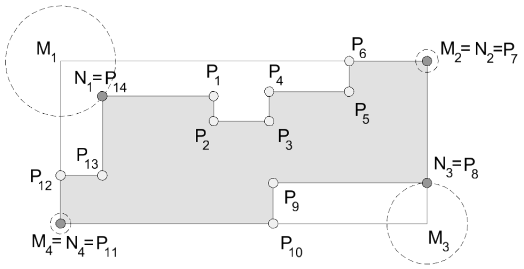


Figure 3: Recursive splitting of initial edge: finding closest vertices.

4.3 Recursive splitting of other edges

The splitting procedure for other edges of the building is based on the same idea as the splitting procedure for an initial edge. However, it is accompanied by several steps. Recursive splitting of such edge is not always done, but only in cases, when the form of the edge is more complex. For an assessment of the shape complexity a criterion based on a standard deviation σ is used. The splitting criterion σ is calculated repeatedly for each edge of the building, and assessing distances of the points to the regression line. As mentioned above, until $\sigma > \sigma_{max}$, such edge is recursively splitted into sets of new edges. Having regard to the brevity, only the most important steps and options of the splitting procedure will be presented.

A splitted edge is intersected by the regression line at least in one point (except the case of collinearity). Such edge is decomposed to several new edges, whose end points are intersections of this edge and the regression line. Those edges are called *temporary edges*. The idea of the splitting procedure is very simple. Each temporary edge is processed separately. After that, all temporary edges are joined and replaced by several edges of a simpler form. A splitting procedure for an edge e can be described as follows:

1. Intersection of the regression line and the splitted edge e.
2. Create set of k temporary edges $\{t\}$. For every edge t_k and recursion depth r split the temporary edge:
    (a) Find a position of the temporary edge t_k to the regression line (left or right).
    (b) Construct the smallest area enclosing rectangle over t_k.
    (c) Find a pair vertices N_1, N_2 closest to a pair of points M_1, M_2 (left position) or a pair of vertices N_4, N_3 closest to a pair of points M_4, M_3 (right position), see the Chapter 4.3.1.
    (d) Set orientation $g^{(r)}$ for all points P_i lying within the temporary edge t_k.
3. Create new edges over all detected vertices of all temporary edges.
4. Add those edges to the stack S.

Each edge e is tested, whether it intersects the regression line, a list of its intersections is stored for each edge. Intersection can be easily calculated from parametric equations, this step can be done in linear time.

A temporary edge t is constructed from all points located between two adjacent intersections. However none of the pairs of intersections is added to the temporary edge. In some cases (eg temporary edge formed by only one point P_i), it is necessary to correct subsequently a form of the temporary edge. For example, such triangle, represented by point P_i, intersection of the segment P_i, P_{i+1} with the regression line and an orthogonal projection of P_i to the regression line, is replaced by the rectangle with the same area.

4.3.1 Splitting of temporary edges

This procedure represents an important step of the algorithm and significantly affect its effectiveness. During the recursive subdivision of each edge the smallest area enclosing rectangle over this edge is constructed. As mentioned above, every splitted edge of the building has the same orientation as one edge of the smallest area enclosing rectangle.

Unlike the case of finding the smallest area enclosing rectangle for an initial edge, only one pair of points N_1, N_2 closest to one pair of vertices M_1, M_2, is searched.

First we determine the location of the temporary edge t with respect to the regression line, that passes through the center of gravity $[x_T, y_T]$. A temporary edge can be found left or right to the regression line or it may be collinear. In the third case any change of the temporary edge is not carried out. It is apparent, that location of the temporary edge to the regression line depends on the location of its first point $P_{(1)} = [x_{(1)}, y_{(1)}]$. The next temporary edge will be with regard to the regression line in the opposite position, their positions are alternately changing.

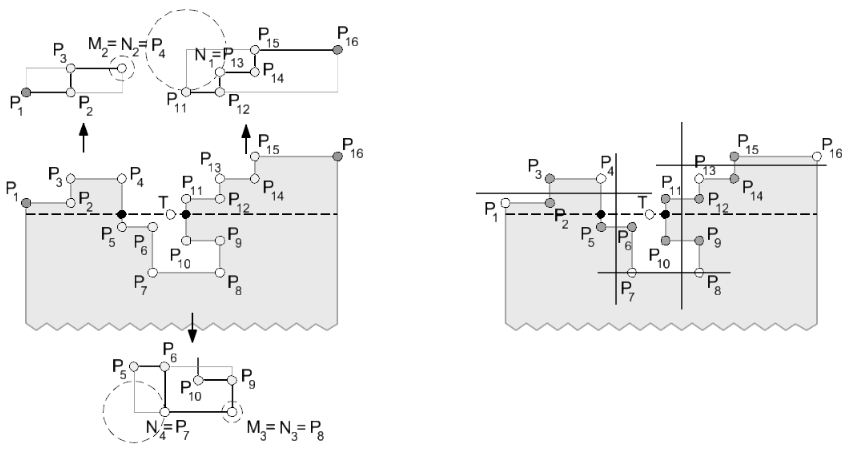


Figure 4: Recursive splitting of the edge with construction of smallest area enclosing rectangles and a replacement of splitted edges by regression lines.

"U" and "L" segments. It should be noted, that points of the building closest to vertices of the smallest area rectangles, represent possibly vertices of such building. As mentioned above, each edge of the building will be replaced by several "U" and "L" segments. The first and the last temporary edge will be replaced by a *pair* of edges. Other temporary edges will be replaced by a set of *three* edges. This step is independent on position of the temporary edge to the splitted edge. Another important variable represents an index of the temporary edge. We already know, that the first and last temporary edges are splitted in another way.

Orientation of vertices and edges. All found vertices are given an orientation $g^{(r)}(N)$ depending on the position of the temporary edge to the regression line. However it is important to maintain an orientation of first and last newly created edges and a corresponding parental edge. Otherwise, it would not be guaranteed, that adjacent edges have any intersection.

Vertices of the building and their orientation. On the basis of an index of the temporary edge and a position of the temporary edge to the splitted edge, corresponding points M_j and N_j (vertices) are searched. There are several rules:

1. *First temporary edge t on the left of the regression line:* Search point N_2 closest to point M_2. Point N_2 becomes a vertex of "L" segment.

2. *First temporary edge t on the right of the regression line*: Search point N_3 closest to point M_3. Point N_3 becomes a vertex of "L" segment.

3. *Last temporary edge t on the left of the regression line:* Search point N_1 closest to point M_1. Point N_1 becomes a vertex of "L" segment.

4. *Last temporary edge t on the right of the regression line:* Search point N_4 closest to point N_4. Point N_4 becomes a vertex of "L" segment.

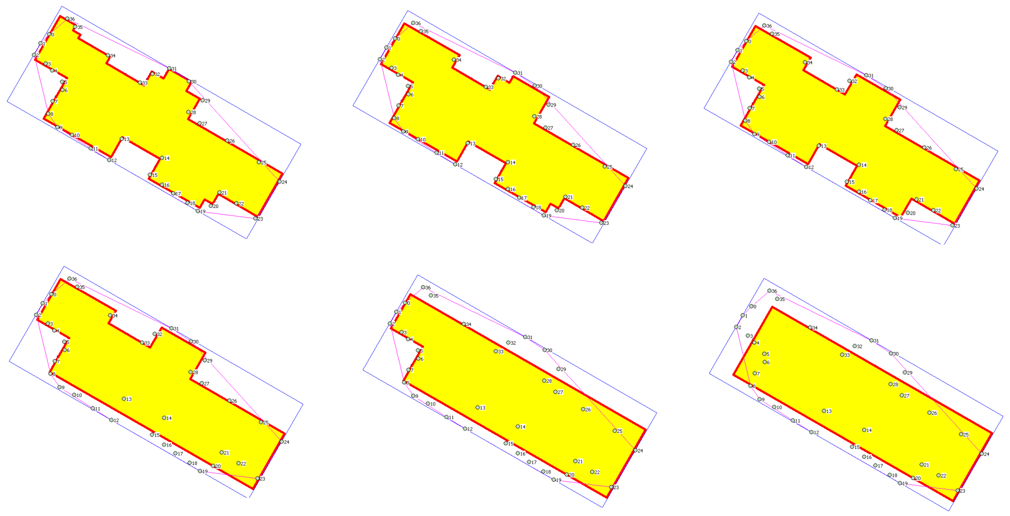


Figure 5: Simplification process of the buildings for various values of σ_{max} in six steps.

5. *Other temporary edge t on the left of the regression line:* Search two points N_1, N_2 closest to points M_1, M_2. Points N_1, N_2 become vertices of "U" segment.

6. *Other temporary edge t on the right of the regression line:* Search two points N_3, N_4 closest to points M_3, M_4. Points N_3, N_4 become vertices of "U" segment.

One parental edge may arise at least three new edges. Each detected vertex of the temporary edge becomes a start point of the new edge and an end point of the adjacent edge. During this step a set of new edges is formed, all edges are subsequently added to the stack. Each edge can be splitted in the next recursion step equally, if necessary.

5 Results and implementation

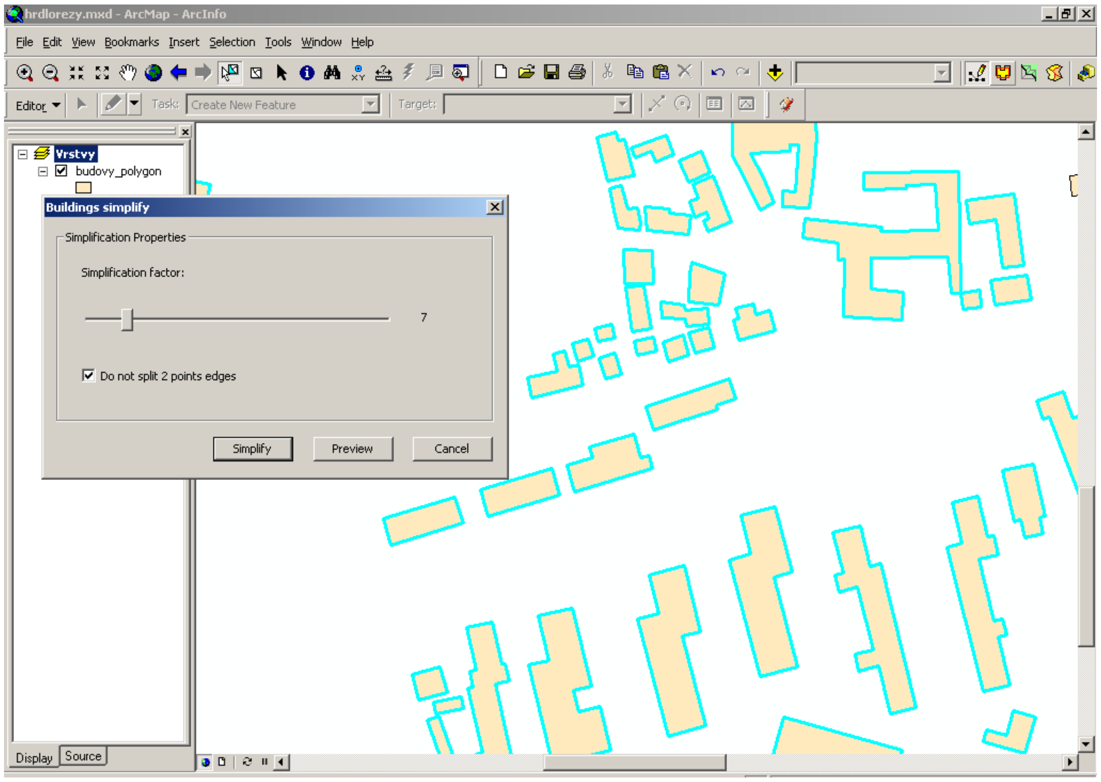
The algorithm was implemented in C++ (also Java version exists), graphical user interface was created using WinAPI, see Fig. 6. With the use of ArcObjects libraries, a new DLL application designed for ArcGIS 9.x was created. This tool is applicable to the polygonal shape files. Author does not use any publicly available library of geometric algorithms.

The presented algorithm gives appropriate cartographic results. It was able to simplify buildings with the high form complexity. In most cases, such buildings represent only geometric constructs. However, they generally verify abilities of the simplification algorithm. It also appeared, that the need for manual corrections of the simplification process is relatively rare. This algorithm is suitable for stand-alone buildings with interior angles formed by adjacent edges of $\pm\frac{\pi}{2}$. They represent buildings of "classical" forms. Modern architecture is typical of a greater form variability and more complex structures. Due to the fact, that original segments of the buildings are replaced by perpendicular segments, results may look artificially.

The accuracy of determining the angle of rotation φ affects the resulting quality of the simplification algorithm. For U-shaped and L-shaped buildings better results are achieved, L-shaped and Z-shaped buildings bring more problems. For these buildings the smallest area enclosing rectangle may not represent prevailing form of the building. As a result of the recursive splitting new vertices in such positions, which are absent in the original building, can be created. And so the resulting polygon meets geometric conditions, but it is not similar to the building.

6 Conclusion

This paper introduced an algorithm for automated cartographic generalization of the buildings. Presented algorithm is based on the recursive approach. It provides quite appropriate cartographic results and minimizes the needs of manual



Buildings simplify

Simplification Properties

Simplification factor: 7

☑ Do not split 2 points edges

Figure 6: ArcGIS 9.3 and Buildings Simplification Tool.

corrections. This algorithm is suitable for stand-alone buildings with interior angles of adjacent edges of $\pm\frac{\pi}{2}$. It was implemented in C++ and designed as a stand-alone function for ArcGIS 9.x. This tool is applicable to the polygonal shape files. In the next step, the author is going to modify the algorithm so that it is suitable for general buildings and also prevents self intersections.

References

[1] Sester M.: *Generalization based on least square adjustment*, International Archieves of Photogrammetry and Remote Sensing, 2000.

[2] Dutter M.: *Generalization of buildings derived from high resolution remote sensing data*, Wien, 2007

[3] Galanda M.: *Automated Polygon Generalization in a Multi Agent System*. Mathematisch-naturwissenschaftlichen Fakultät, Universität Zürich, 2003.

[4] Toussand G., *Solving Geometric Problems with the Rotating Calipers*, McGill University Montreal, 1983

[5] Rourke O. J., *Computational Geometry in C*, Cambridge University Press, 2005

[6] Berg M., Kreveld M., Overmars M., Schwarzkopf O., *Computational Geometry*, Springer, 2000