DOI: 10.14714/CP79.1212

PEER-REVIEWED ARTICLE

Pairwise Line Labeling of Geographic Boundaries: An Efficient and Practical Algorithm

Maxim Rylov
Heidelberg University
maxim.rylov@geog.uni-heidelberg.de

Andreas Reimer
Heidelberg University
andreas.reimer@geog.uni-heidelberg.de

We present an algorithm that labels linear features with two matched toponyms describing the left and the right side of a line, respectively. Such a pairwise line labeling strategy is commonly used in manually produced maps to differentiate administrative or other geographic divisions. Our approach solves two basic tasks of the automated map labeling problem, namely candidate-position generation and position evaluation for a given scale. The quality of the name placement is evaluated by comparison to a set of established cartographic principles and guidelines for linear features. We give some results of our experiments based on real datasets. The implementation of our algorithm shows that it is simple and robust, and the resulting sample maps demonstrate its practical efficiency.

KEYWORDS: automated label placement; automated cartography; quality evaluation; computational geometry; GIS mapping

INTRODUCTION

Over the past few decades there have been many attempts to automate label placement in the field of cartography. Label placement algorithms have matured from being able to solve only the simplest problems (Yoeli 1972; Hirsch 1982; Basoglu 1982) into complex and sophisticated tools (see the excellent bibliography of papers on this topic produced by Wolff and Strijk [2009]). Examples used in map production include the *Maplex Label Engine* (ESRI 2009) and *Label-EZ* (**maptext.com/labelez**). The main goal of labeling algorithms is to relieve a human cartographer of two manual tasks, namely:

- The editing of the map, i.e. confirming the names are correct.
- The positioning of names on the map using predefined typefaces.

As a consequence, automated type placement reduces map production time and cost. Although commercial and open source labeling packages have been available for some time, there is still a great need to manually resolve conflicts and use non-automated labeling techniques in order to achieve a professional level of functionality and legibility on the final map. In addition, labeling packages are often difficult to parameterize to match production standards (Revell et al. 2011; Regnauld et al. 2013).

In cartography, all map objects to be labeled can be divided into three categories (Imhof 1962; Imhof 1975; Wood 2000; Brewer 2005): punctiform (e.g., settlements, mountain peaks), linear (e.g., roads, rivers, boundaries) and areal (e.g., countries, lakes, islands) designations. Each type of designation has its own requirements and involves its own challenges. Compelling attempts to automate map lettering were made by Yoeli (1972), Christensen et al. (1995), van Kreveld et al. (1999) for point features, by Barrault and Lecordix (1995), Edmondson et al. (1996), Chirié (2000), Wolff et al. (2001) for lines and by van Roessel (1989), Barrault (2001), Rylov and Reimer (2014b) for areas. In this article we propose a method for the pairwise labeling of a special type of linear feature: those that demarcate area boundaries. There are several situations in which a boundary needs to be labeled twice, differently on each side of the linear feature: e.g. international borders, municipal divisions, grid-zones or military zonings where different rules of engagement apply. In manual

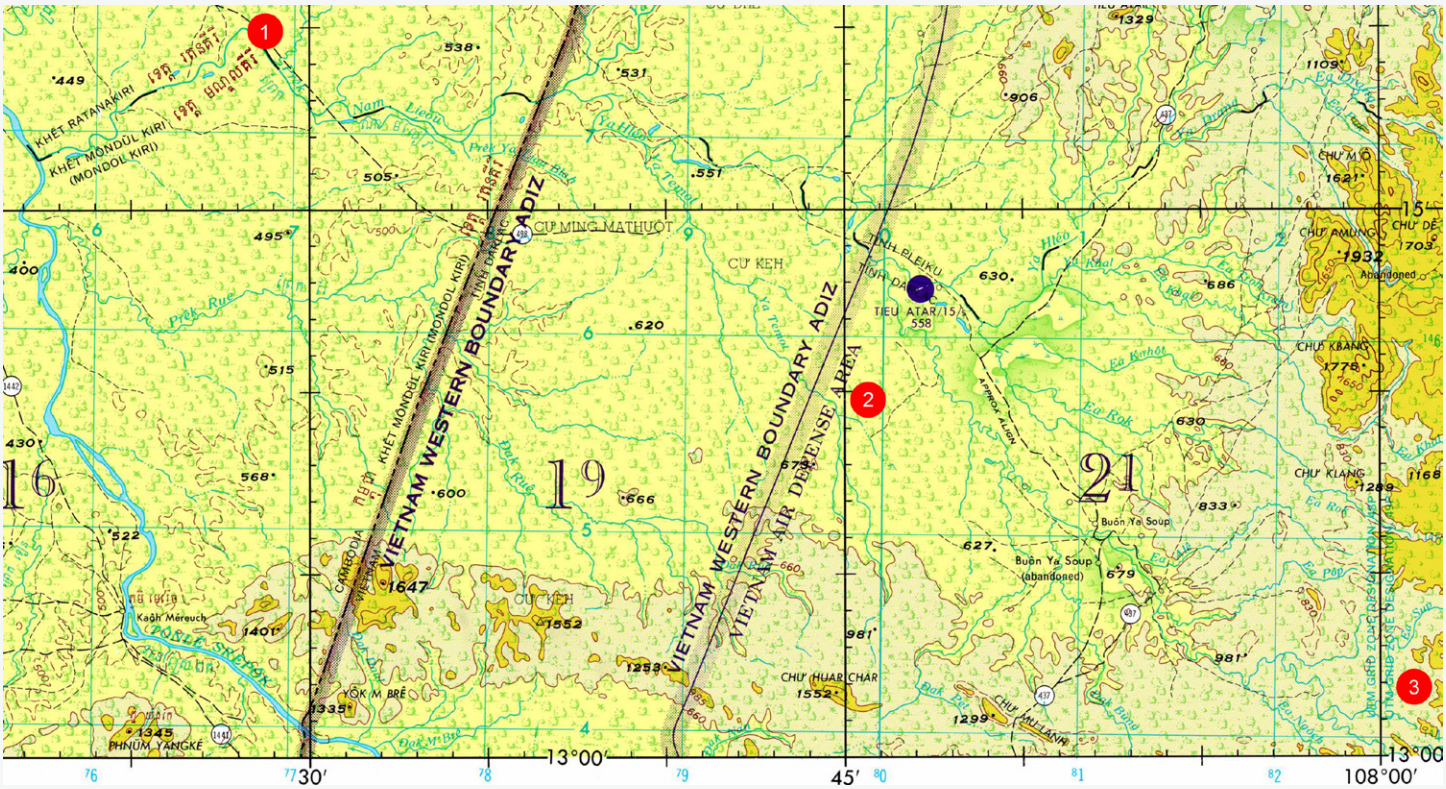


Figure 1. *An example of manual pairwise line labeling: provincial boundaries (bilingual) [1], zones for changing rules of engagement [2] and UTM grid zone changes [3]. Source: Defense Mapping Agency (1973).*

cartography two different techniques are used to letter a boundary line in pairwise manner: the boundary can be labeled either with text placed along a straight line (Figure 1) or curved following the direction of the polyline to be annotated (Figure 2). Curved lettering is often the preferred choice, aesthetics-wise. This paper presents an algorithm that is able to position labels in a way which is visually similar to the approach used in Figures 1 and 3b, when the label is not curved.

This method can be used on large-scale maps to label areas when the scale becomes too large to place the label inside the area. With pairwise line labeling, regions that lie on opposite sides of a boundary line can be identified without difficulty. The main visual advantage is that a map reader is informed about the exact nature of the line, not only its general type. This helps to easily distinguish boundaries from other linear objects and amplifies the precise graphic relation between the toponyms and the relevant map features. Another strong feature of such labeling is the consideration of two names as a unit or a single label. It means that the resulting map is free from partial designations, i.e. a label either on the left or on the right side of the object (Figure 3a).

To the best of our knowledge, there are no preceding published works regarding automated pairwise line labeling. However, it is worth noting that some existing commercial label engines have the ability to label administrative boundaries. For instance, the *Maplex Label Engine* produces labeling of administrative units for each side of a linear feature independently (Figure 3a). This kind of labeling can be performed using any line labeling algorithm (e.g., Barrault and Lecordix 1995; Edmondson et al. 1996; Chirié 2000; Wolff et al. 2001). Note that such

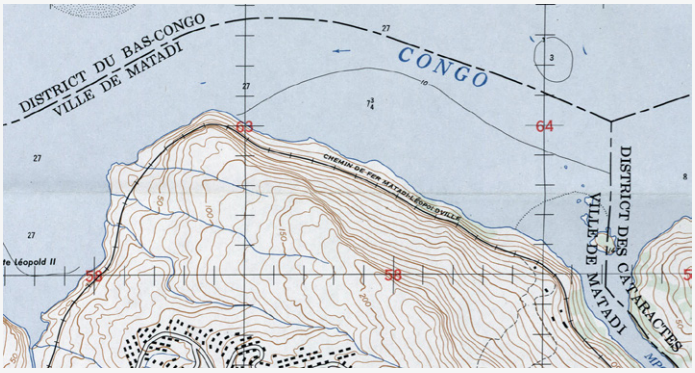
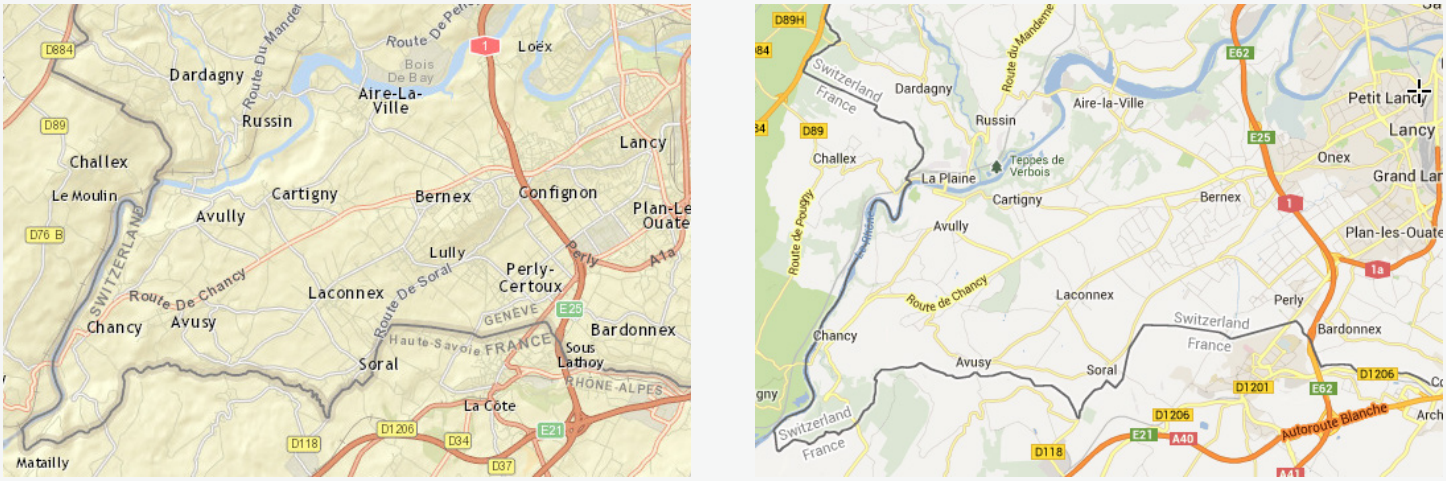


Figure 2. *An example of curved labels annotating administrative boundaries. Source: National Imagery and Mapping Agency (1962).*



(a) (b)

Figure 3. *Examples of different labeling of administrative boundaries. (a) Non-pairwise line labeling of boundaries with mixed hierarchies (© Esri). (b) Pairwise line labeling, closed source and unknown parameterization (© 2014 Google).*

label placement is not widely used in traditional cartography and violates cartographic labeling principles found in the literature and extant topographic maps. For example, this approach often creates ambiguities between the labels which annotate the boundaries of different subdivision levels (Figure 3a, "GENEVE" & "FRANCE"). The next example in Figure 3b illustrates map labeling on Google Maps, where the labels of national borders are coupled and positioned in regions with less curvature and where the text is less sloped. The two presented approaches follow different cartographic precepts, if at all. We interpret both approaches as arising from technical and theoretical limitations. The description and implementation of both algorithms is, of course, not known and closed source. Moreover, we have found no free/open source label engine or research publication aimed at pairwise label placement.

We start describing our method in the following section with a formalization of the criteria representing the cartographic guidelines for pairwise line labeling. Next, we introduce a general form of our scoring function (van Dijk et al. 2002). Then, we continue with a description of the first part of our model, which consists of an algorithm for generating of a set of potential label positions for each linear feature. Subsequently, we describe the components of the quality function in detail. The proposed quality measures take into account

- the curvature of the polyline,
- the offset of label from the polyline,
- the orientation of the lettering, and
- an even distribution of the labels along the polyline.

In general, a quality evaluation, or an objective function, can be employed by any *combinatorial optimization* algorithm (Christensen et al. 1995; Rabello et al. 2014) for finding a feasible near-optimal solution of the automated label placement problem. Note that the characteristics, like the position and the quality assessments, of the output label candidates can be used as input to a much more comprehensive and sophisticated general map labeling algorithm (Edmondson et al. 1996; Kakoulis and Tollis 1998). For example, such an algorithm could consider figure-ground relationship (Rylov and Reimer 2014c) or resolve any ambiguities between neighboring labels (Rylov and Reimer 2014a). In our results section, we show some significant map samples based on real-world datasets. These sample maps are labeled using our implementation of the proposed method. Finally, we conclude with a brief analysis of the present work and give some insights for possible improvements and future research.

CARTOGRAPHIC MODEL

The basic idea of our labeling model is depicted in Figure 4b. Succinctly, the necessary input of our algorithm is a polyline that describes a boundary, and two toponyms that define adjacent areal features. The output is a set of coupled labels that represent either side of the polyline to be annotated.

APPROACH METHODOLOGY

Automated text placement, or lettering, is one of the most difficult and complicated problems to be solved in automated cartography and geographic information systems. When it comes to solving a complex problem, usually the problem is decomposed into smaller and simpler sub-problems. In our approach we use the same technique. Thus, the map labeling problem in general can be divided into three substantially independent subtasks (Edmondson et al. 1996):

- *Candidate-position generation*: A method that generates a set of label candidates for each map feature, using its spatial characteristics and taking into account its type (e.g., point, line or area). The generated potential label positions are normally considered as the search space for the position selection procedure.

- *Position evaluation*: A process of computing a score for each label candidate. This score is calculated using a quality function, which measures how well a label is positioned with respect to the object it tags as well as to other labels and features on the map (van Dijk et al. 2002). In general, the quality function should take into account and reflect the formal cartographic precepts applied to each type of label (see Rylov and Reimer [2014a], for point features).

- *Position selection*: A process of choosing only one label position from each set of candidates, such that the total label quality measured with the quality evaluation function is globally maximized thereby achieving a superior level of cartographic quality in the resulting map (Christensen et al. 1995). Note that the *selection* is an *NP-hard* problem in general (Formann and Wagner 1991; Marks and Shieber 1991).

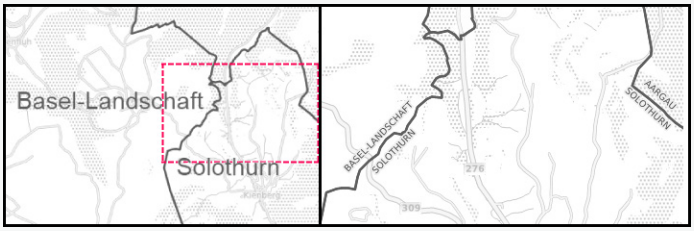


Figure 4. *Labeling of administrative regions in Switzerland: (a) Placement of the names inside the areas. The dashed rectangle corresponds to map (b). (b) Positioning of the names along the boundary line.*

Our method deals with the two first subtasks of automated map lettering for the case of pairwise line labeling. Once these subtasks are solved, the position selection procedure can be applied. The position selection is canonically treated as a general optimization problem via strategies such as *exhaustive search methods* (Yoeli 1972; Hirsch 1982), *simulated annealing* (Edmondson et al. 1996), *genetic algorithms* (Verner et al. 1997), *gradient based optimization* (Christensen et al. 1995) or *tabu search* (Yamamoto et al. 2002). We provide the solution of these subtasks in the next three subsections. But first, we define and enumerate requirements for pairwise line labeling according to corresponding cartographic guidelines.

LINEAR FEATURE LABELING REQUIREMENTS

We have selected and operationalized the relevant rules for pairwise line labeling from the extant cartographic literature on positioning names on maps (Imhof 1962; Imhof 1975; Yoeli 1972; Wood 2000; Brewer 2005). The list of design guidelines adapted to our problem is as follows:

G1. A label must be placed along the linear feature it tags.

G2. A label should conform to the curvature of the polyline.

G3. Avoid complicated and extreme curvatures of the polyline. Straight or almost straight parts of the polyline should be preferred.

G4. A label must be placed close to the polyline, but not too close.

G5. The name must not be spread out, but may be repeated at specified intervals along the linear feature.

G6. Avoid placing names near end points of the polyline.

G7. Horizontally aligned labels are preferred to vertical ones.

G8. The two parts of a label should be centered relative to each other.

G9. The name should not cross the linear feature.

The term "label" in the list actually means a "pair of labels," in other words, one label to annotate the left side of the polyline and another label for the right side.

These guidelines are used as the criteria for *candidate-position generation* as well as for the *position evaluation* task in the following up subsections. Note G2 is a general guideline which refers to different methods of lettering depicted in Figures 1 and 2. Our approach deals only with text that is straight.

SCORING LABELING QUALITY

Once a potential position of a label is computed, it is numerically scored using a quality evaluation function. A quality function achieves two main goals: to evaluate the label positions according to the cartographic precepts and to compare various labeling algorithms. Normally, a quality function is defined as a weighted sum of single metrics (van Dijk et al. 2002; Zhang and Harrie 2006) and has the general form:

$$\begin{aligned} Q(L,F) = & \sum_{l \in L} \Big(w_1 \, f_{\text{priority}}(l) + w_2 \, f_{\text{aesthetics}}(l) \\ & + w_3 \, f_{\text{association}}(l, L, F) + w_4 \, f_{\text{label-visibility}}(l, L, F) \Big) \\ & + \sum_{f \in F} \Big(w_5 \, f_{\text{feat-visibility}}(f, L, F) \Big) \end{aligned} \qquad (1)$$

where L is a set of labels, F is a set of non-textual features on the map, $w_1, \ldots, w_5$ are the weights and $f_*(l)$ are the quality metrics, measuring how well the demands of cartographic guidelines are met in the positioning of labels. The value of a quality function such as Equation (1) is usually normalized to the range [0,1]. For a detailed description and the meaning of each partial metric $f_*(l)$ we refer to the work by van Dijk et al. (2002). In addition, a review paper by Kern and Brewer (2008) contains a comparison table that shows how the four criteria $f_{\text{aesthetics}}$, $f_{\text{association}}$, $f_{\text{label-visibility}}$, and $f_{\text{feat-visibility}}$ have been used in various proposed techniques and algorithms presented in the literature.

In Equation (1) the measure $f_{\text{aesthetics}}(l)$ evaluates the quality of the position and the shape of a label with respect to the geometry of the feature it annotates, and $f_{\text{association}}(l, L, F)$ describes the clarity of the association between a feature and its label. In our approach, we construct a quality evaluation function called $H(l)$, which substitutes $f_{\text{aesthetics}}(l)$ and partially substitutes $f_{\text{association}}(l, L, F)$. With this new function we will numerically score potential label positions that are output by the algorithm presented in the next section. The components of the measure are metrics designed to meet the requirements of some of the cartographic guidelines specified in the previous section. Let us define $H(l)$ for scoring l by analogy with Equation (1) as:

$$\begin{aligned} H(l) = & \; m_1 \, g_{\text{PosDev}}(l) + m_2 \, g_{\text{BaseOffset}}(l) \\ & + m_3 \, g_{\text{GoodnessOfFit}}(l) + m_4 \, g_{\text{HorizAlign}}(l) \end{aligned} \qquad (2)$$

where $m_1, \ldots, m_4$ are the weight factors and $g_*(l)$ the functions are:

- $g_{\text{PosDev}}(l)$, measuring the deviation of a label position from an even distribution of labels along the polyline;
- $g_{\text{BaseOffset}}(l)$, evaluating how far a label's baseline is from the feature's centerline (Figure 5);
- $g_{\text{GoodnessOfFit}}(l)$, representing a measure for quantifying how well the centerline approximates the polyline in a given region; and
- $g_{\text{HorizAlign}}(l)$, evaluating the deviation of the orientation of the label from a horizontal alignment.

Note that the weights $m_1, \ldots, m_4$ should sum to 1.0 and the return value of each partial metric should fall into the range [0,1]. We design our metrics to yield higher values for label positions that are closer to the ideal position. To examine other research attempts which deal with developing quality measures to quantify label positions for linear features, see Barrault and Lecordix (1995), Edmondson et al. (1996), and Chirié (2000).

ALGORITHM FOR CANDIDATE-POSITION GENERATION

In this section we introduce an algorithm that produces candidate positions along the input polyline (Figure 5). The algorithm produces a set of imaginary line segments (see *centerline* in Figure 5) which locally approximate the original polyline in a certain region. The width of the region (grey area) equals the maximum width of the two names. In addition to the centerline, an offset of the label (see *d* in Figure 5) from the polyline is computed. The centerline and the offset define two baselines. The *baseline* is the line upon (or under) which the characters of the name are drawn. Note that the candidate-position generation algorithm complies with the guidelines G1, G2, G4, G5, and G6.

Let us define some useful terms and measures before giving a detailed description of the algorithm. The input of our algorithm consists of a polyline $P = (p_1, \ldots, p_n)$ specified by a sequence of points $p_i = (x_i, y_i)$, where $i = 1, \ldots, n$ (Figure 6), and two names n_l and n_r that describe the left and the right side of the polyline P. We denote the total length of P by L. Let w_l and w_r be the widths (in map units) of n_l and n_r respectively. In order to satisfy requirement G5, we introduce a parameter S that defines the distance between names repeated along the polyline (Figure 6). We define the width of a label as $w_{max} = \max(w_l, w_r)$. The algorithm is composed of four phases that are detailed below.

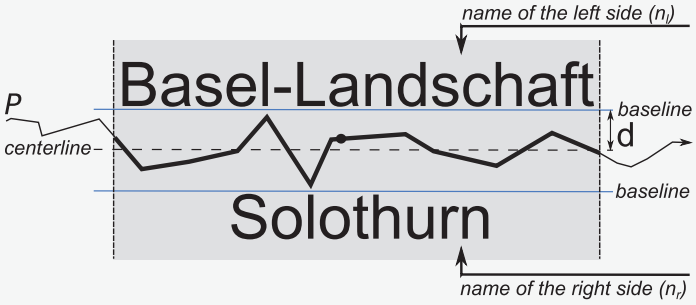


Figure 5. *The nomenclature used in describing the candidate-position generation algorithm.*

PHASE I

In the first phase, we generate a set of candidate locations along the polyline P (guideline G5). We denote a point that represents the anchor point of a candidate position by q_j, where $j = 1, \ldots, m$ and $m = \lceil (L - S) / S \rceil$, the number of such points. The point q_j lies on P and its distance from the starting point of P is defined by $S'(j) = (1/2 + j)\, S$.

Let q_j, $j = 1, \ldots, m$, be the points at which we will construct the centerline for placing a label. We consider q_j as preliminary locations, different from the resultant ones. The explanation of the difference between them is provided below. In order to increase the size of the search space, we move each point q_j along the polyline in both directions until the distance from q_j along P reaches a certain value, the maximum position deviation D_{max}. This approach gives us a set of positions (see blue areas in Figure 6), $\overline{q}_{jk}$ where $k \in [-N, N]$ with the center at q_j, $N \in \mathbb{Z}$. We denote this set by V_j. N is the half of the number of preliminary locations in V_j and defined as $N = \lceil D_{max} / D_{step} \rceil$, where D_{step} is the distance between two points $\overline{q}_{jk}$ and $\overline{q}_{jk+1}$. The total number of preliminary locations is calculated as $N_{total} = 2Nm = 2N\lceil (L - S) / S \rceil$. Assume that each point $\overline{q}_{jk}$ specifies a rough position for a label placement. Therefore, the maximum number of labels for one linear feature is equal to m, as only one label from each set V_j can be chosen. The adjustment of D_{max} and D_{step} should be done by the user, which controls the size of the search space. The parameters S, D_{max}, and D_{step} are measured in map units.

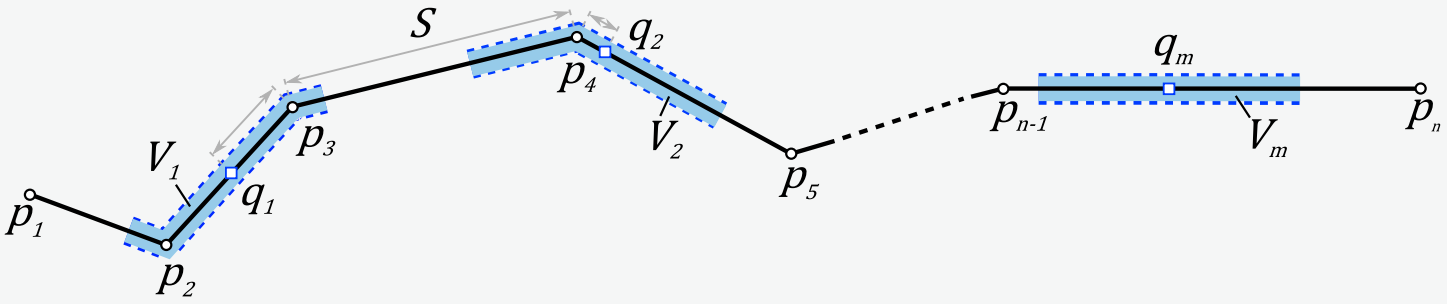


Figure 6. *The input polyline P (solid black line) with nodes p_i. Points q_j are the centers of sets of potential locations V_j. S is the interval between q_j and q_{j+1}.*

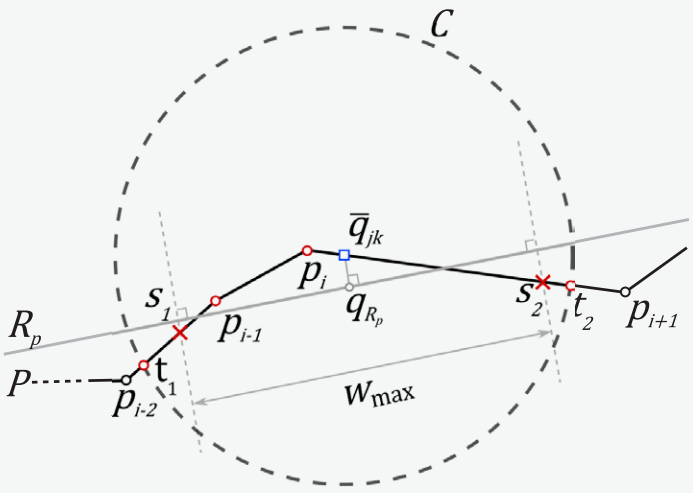


Figure 7. *Best-fitting straight line R_p for a set of points of P with its center in $\overline{q}_{jk}$.*

As the allowed position deviation D_{max} increases, the distribution of labels along P becomes less regular. It can be seen in Figure 6 that the method for candidate-position generation also complies with G6 (avoidance of end points) automatically.

PHASE II

In this phase, we try to find a centerline which approximates a part of P centered at $\overline{q}_{jk}$. Each part of this kind consists of points whose distance from $\overline{q}_{jk}$ along P is at most $w_{max}/2$. Such a centerline, or the best-fitting straight line denoted as R_p, can be found by employing the method of least squares (Chatterjee and Hadi 2006). This method requires a set of points as the input. An approach to finding this set of points on P and consequently the line R_p is described in the following steps.

1. Let C be a circle with the center at $\overline{q}_{jk}$ (Figure 7) and a radius equal to $r_c = Kw_{max}$, where K is a control parameter in the range [0.5,1]. As the actual shape of P is unknown, the circle radius is grown by increasing K until a satisfactory solution is found, i.e. step 3 has been passed and a centerline was found. Next, we want to find points of intersection between P and C. Due to the possible sinuosity of P, there could be many such points. Therefore, we consider only those two points of intersection whose distance from $\overline{q}_{jk}$ along P is the shortest. These two points we denote as t_1 and t_2. Note that there are two special cases when it is not possible to find these points: when P fully lies inside the circle C, and when $\overline{q}_{jk}$ is too close to one of the ends of P.

   The distance between t_1 and t_2 should be large enough to accommodate the label. Therefore, we check whether the distance between t_1 and t_2 is less than w_{max} before moving on to the next step. A refinement step can also be applied by trying several circles with different radii, as the curvature of a polyline can vary greatly from a straight line to a very bent curve.

2. Construct the best-fitting straight line R_p from a set of points. This set consists of all vertices of P that lie inside the circle C. In Figure 7 these points are: t_1, p_{i-1}, p_i, and t_2. R_p is a preliminary line.

3. We check whether P reverts too far back on itself for label placement, i.e. whether it represents a bulge in the segment under consideration. For this, we construct the perpendicular to R_p through the point $\overline{q}_{jk}$ and check whether the points t_1 and t_2 are on the same side of the perpendicular. If the points t_1 and t_2 happen to be on the same side of the perpendicular, we consider R_p to be invalid. In this case we skip $\overline{q}_{jk}$ and move to the next point $\overline{q}_{jk+1}$ and repeat steps 1–3.

PHASE III

Every time the circle C is grown beyond $K = 0.5$, the Euclidean distance between t_1 and t_2 can be greater than w_{max}. In this case we assume that R_p is not optimal and consider it as a first approximation. Therefore, we describe the procedure that refines the result of Phase II.

1. Construct a perpendicular from the point $\overline{q}_{jk}$ to R_p. Find a point q_{Rp} that is the intersection of the perpendicular and R_p.

2. Find two perpendiculars to R_p that are equidistant from the point q_{Rp}. The distance between q_{Rp} and each of them is $w_{max}/2$.

3. Find the points of intersection between P and the perpendiculars from step 2. Denote these points as s_1 and s_2 respectively (Figure 7).

4. Find the best-fitting straight line R from a new set of points s_1, p_{i-1}, p_i, s_2 (Figure 8).

5. Construct a perpendicular from the point $\overline{q}_{jk}$ to R. We denote the point of intersection as q_R. This point defines the center of a label that will be placed along the centerline R.

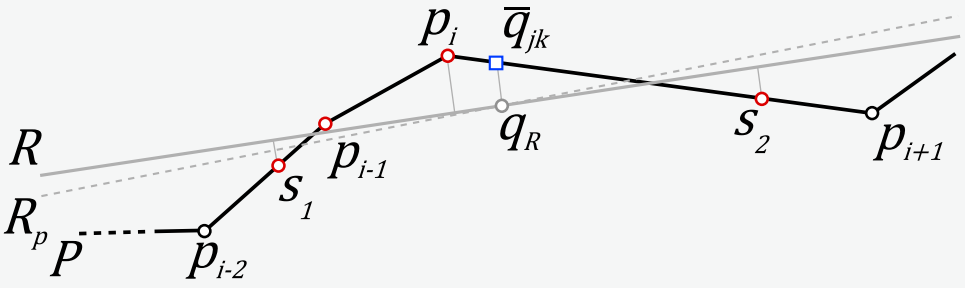


Figure 8. *Refinement of the preliminary centerline R_p.*

Note that Phase III should be omitted if $K = 0.5$.

PHASE IV

The final phase computes the offsets for the baselines upon which, or under which, the labels will be placed (see Figure 5). This phase has three steps.

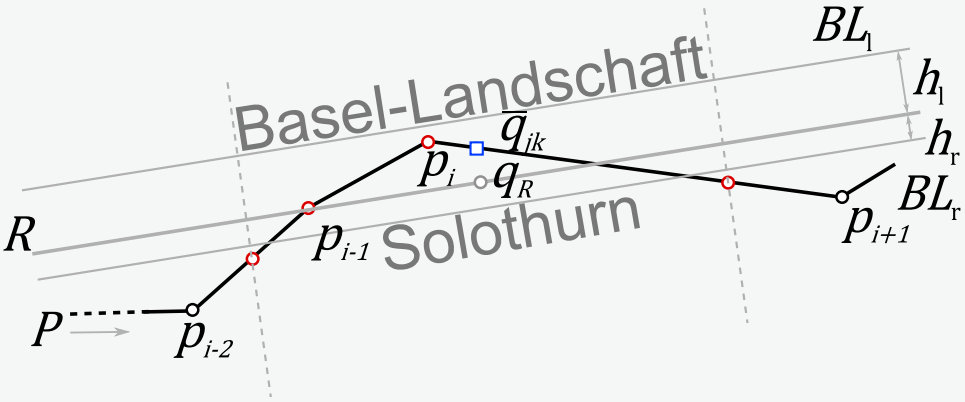


Figure 9. *Computation of baseline offsets h_l, h_r from the centerline R.*

1. Compute the Euclidean distance between R and each point in the set of points that we have employed for constructing R.
   Put the values of the distances into two separate lists. The first list contains the points that lie on the left side of R and the second list for the points on the right side.

2. Compute the maximum value of all entries in each list. These values denoted as h_l and h_r are the offsets of the baselines BL_l and BL_r from the centerline R (Figure 9). Each offset defines the Euclidean distance of the respective baseline to R.

3. Increase each offset from the centerline by adding a typeface-dependent value to each offset. This approach helps to avoid overlapping of the polyline with the descenders or the ascenders of the characters of the label, e.g. if $h_l = 0$ or $h_r = 0$.

To comply with the condition of G8, both names should be centered with respect to q_R.

OUTPUT OF PHASES I–IV

After applying the four phases to each point $\overline{q}_{jk}$, a candidate label position l can be defined with the following properties expressed as functions of l that will be used later in the quality measures:

- The center point, denoted as $q_R(l)$.
- Two offsets $h_l(l)$ and $h_r(l)$ from the centerline that represent the baselines for placing the characters.
- The tilt of the label, denoted as $\alpha(l)$.
- The deviation of a label from an even distribution, denoted as $\delta(l)$.
- The coefficient of determination (explained below), denoted as $\gamma(l)$. The value of this function is computed from the same set of points (see red points in Figure 9) that we used for computing the centerline R.

The output of the presented method meets the requirements of six cartographic guidelines listed above, namely guidelines G1, G2 in Phases II–III, partially G3 in Phase III (see Step 4), G4 in Phase IV. The requirements of guidelines G5 and G6 are met automatically in the approach for computing of q_j (Phase I). In the following sections we also use G4 and G5 for scoring label candidates.

In the following subsections we provide a more detailed description of each metric in Equation (2).

POSITION DEVIATION METRIC

In order to follow guidelines G5 and G6, the labels should be placed along P. We have already given the procedure that generates the candidate label positions with their centers near the points q_j. If an input polyline is more curved, as it is often the case when a border is following a natural feature (e.g. rivers, mountain ranges, etc.), it is not always possible to make a label placement at a certain position q_j. Therefore, our method allows increasing the number of candidate positions around q_j. These potential label placements are anchored at $\overline{q}_{jk}$. It might be that two labels specified by two locations $\overline{q}_{jk}$and $\overline{q}_{jk+1}$, from two different sets V_j and V_{j+1}, are too close to each other (Figure 6). Thus, we need a metric to quantify the deviation of label candidate positions from an even distribution, i.e. the deviation of each point in V_j from the center point q_j. A function for this metric can have the following form:

$$g_{\text{PosDev}}(l) = 1 - \delta(l)/D_{\max} \qquad (3)$$

where $\delta(l)$ returns the length of the part of the polyline that is bounded by the points q_j and $\overline{q}_{jk}$. Figure 10a depicts an example of the function for $g_{\text{PosDev}}(l)$. It is clear that the metric in Equation (3) gives the highest score when $\delta(l)$ returns 0.0; the worst case is when $\delta(l)$ returns $D_{\max}$.

BASELINE OFFSET METRIC

One of the output values of the candidate-position generation method are the offsets from the centerline R, which we call *baseline offsets*. Since the values of $h_l(l)$ and $h_r(l)$ represent the maximum distance between the centerline R and the points of P, it is clear that placing the labels on the lines BL_l and BL_r (Figure 9) respectively can lead to overlapping of P with descenders or ascenders of the label characters. To avoid this problem, we propose an additional offset to the values obtained from $h_l(l)$ and $h_r(l)$. The underlying idea is simple. We need to translate the baseline some distance in a direction perpendicular to the centerline. This additional offset we denote as ε. Note that the value of ε should be chosen by taking into account the font size of the label and the thickness (stroke width) of the boundary line.

Let us define a quality metric:

$$g_{\text{BaseOffset}}(l) = 0.5u(h_l(l) + \varepsilon) + 0.5u(h_r(l) + \varepsilon) \qquad (4)$$

where function u has the form:

$$u(\beta) = \begin{cases} 0, \beta < B_{\min} \\ 1 - \dfrac{\beta - B_{\min}}{B_{\max} - B_{\min}}, \beta \in [B_{\min}, B_{\max}] \\ 0, \beta > B_{\max} \end{cases} \qquad (5)$$

where $B_{\min}$, $B_{\max}$ are minimum and maximum allowed offset values. Function (5) (Figure 10b) yields a value of 0.0 when the distance β between the baseline of a label and the centerline is less than $B_{\min}$ or greater than $B_{\max}$, and a value of 1.0 when $\beta = B_{\min}$. It means that labels whose distance from the centerline is in the range $[B_{\min}, B_{\max}]$ are all acceptable. Note that the closer the label is to the centerline

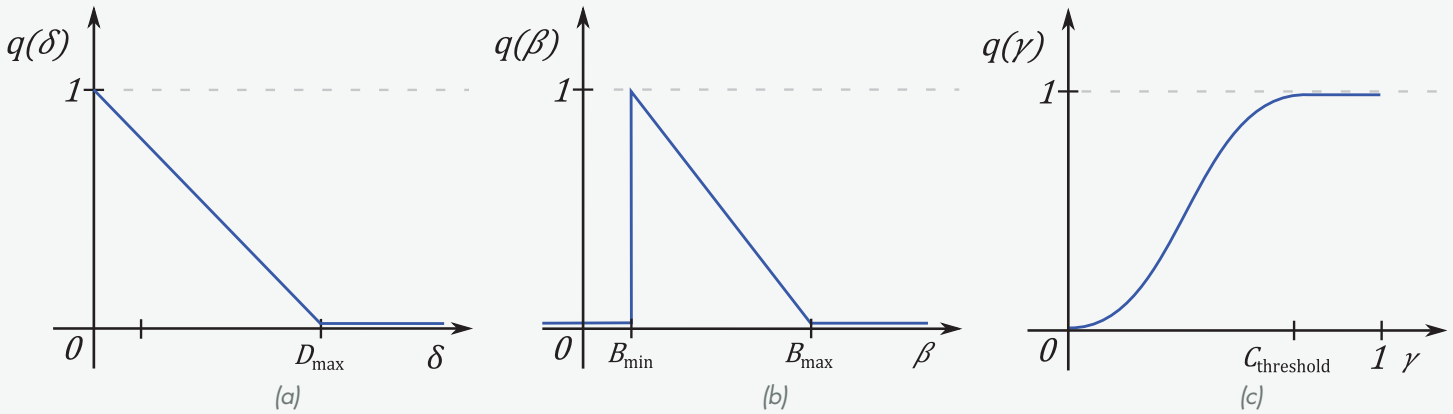


Figure 10. *Quality functions used in the metrics. (a) Position deviation metric. (b) Baseline offset metric. (c) Goodness of fit metric.*

the better. Parameter B_{max} defines the upper limit above which the label-feature association becomes unclear. The metric (4) scores how well the requirement of G4 is met.

GOODNESS OF FIT METRIC

We used the method of least squares for generating candidate-positions. Hence, we can calculate the coefficient of determination that equals the square of the correlation coefficient between the observed (polyline points) and modeled (centerline) data values for the case of a simple regression model. The coefficient of determination is a statistical characteristic that provides us with some information about the goodness of fit of a model. In our case it measures how well the centerline R locally approximates the polyline P (Figure 8). The coefficient of determination has values in the range [0, 1], where a value of 1.0 indicates that the centerline fits the polyline perfectly, for instance when all points used for the construction of R (see Phase II step 2 or Phase III step 4) lie on one line segment of P. Let us construct a metric that employs the coefficient of determination. For this purpose we use an appropriate fading function, and also define a threshold to the value of the coefficient of determination, denoted as $C_{threshold}$. Figure 10c depicts an example of such a fading function, in which the highest value is obtained if the goodness of fit is equal to the threshold value or higher. The quality of the fit to the polyline deteriorates as the goodness of fit metric approaches 0. As a fading function we chose the following:

$$g_{GoodnessOfFit}(l) = \begin{cases} 1, \gamma(l) > C_{threshold} \\ 1 - \left(\cos\left(\frac{\pi\gamma(l)}{C_{threshold}}\right) + 1\right)/2, \gamma(l) \leq C_{threshold} \end{cases} \quad (6)$$

where $\gamma(l)$ returns the coefficient of determination. Note that metric (6) corresponds to G2.

HORIZONTAL ALIGNMENT METRIC

This metric considers cartographic guideline G7, which says that "horizontally aligned labels are preferred to vertical ones." In other words, the text should be as near to "reader normal" as possible (Wood 2000). Therefore, we can determine the corresponding metric as follows:

$$g_{HorizAlign}(l) = 1 - \frac{\alpha(l)}{90} \quad (7)$$

where $\alpha(l)$ returns the angle between the horizontal axis and the centerline which defines the tilt of the label l. The return value is measured in degrees. Metric (7) is designed to yield a value of 1.0 when $\alpha(l)$ has a value of 0.0. This is the case when a label is horizontal.

PARAMETERIZATION

On first sight, our approach is dependent on ten input parameters. The main reason for this large number is that we present the method as it is actually implemented, to support our goal of reproducibility and adoption by others. Some of the parameters are simply pre-processing thresholds, which could be structurally left out. Others are adaptions of the *sliding model* (see van Kreveld et al. 1999; Strijk and van Kreveld 2002) to polylines, in order to generate more fine-grained candidate positions. The parameters and their functions are summarized in Table 1.

We can see that Q_T (see next section), D_{max}, and D_{step} are just variations dealing with the size of the search space for the solving algorithm (for which we generate the input; see Christensen et al. [1995]) and could structurally be omitted. The distance between label pairs can be best thought of as a scale-dependent function of the number of toponym repetitions the cartographer is aiming for, modified by the available drawing space in relation to the size of the objects to be labeled. Along with the allowed offsets from the polyline B_{min} and B_{max}, distance S could very conceivably be derived automatically for a given map series. As such, the only undefined parameters that require manual input are the four weights. We currently see no alternative to deriving them empirically or through trial and error. In the following experimental section, we provide two cartographically viable sets of weights.

Generally speaking, the number of parameters that are needed to be set by a user can be reduced in an implementation of the method in a GIS application.

Type	Role	Parameter	Meaning
geometric	scale dependent styling	S	distance between label pairs
	sliding model	D_{max}	maximum allowed deviation from label center
		D_{step}	interval between candidates
quality	pre-processing	Q_T	quality threshold
	scale dependent styling	B_{min}	minimum allowed offset from polyline
		B_{max}	maximum allowed offset from polyline
	general styling	m_1	position deviation weight
		m_2	baseline offset weight
		m_3	goodness of fit weight
		m_4	horizontal alignment weight

Table 1. *Overview of the parameters used in our approach.*

EXPERIMENTS

In this section we provide some results of the experiments that we carried out to test our method. We first describe our experimental methodology. Then we present performance results and labeling quality measurements. We finish this section with sample maps generated with our method.

DATASETS, IMPLEMENTATION, AND EXPERIMENTAL METHODOLOGY

We have implemented a version of the proposed algorithm on top of *MapSurfer.NET* (**mapsurfernet.com**), a platform for publishing spatial data to the Web, written in C#. The experiments were conducted on a computer equipped with an Intel® Core™ i5-2500 CPU @ 3.30 GHz and running *Windows 7 Professional* x64 with 8GB installed memory. The runtime execution environment of our test application was *.NET Framework 4.5* (x64).

We performed our experiments on a dataset from the *OpenStreetMap* project (OSM), one of the most promising crowd-sourced projects (Haklay and Weber 2008; Ramm et al. 2010). We chose Italy, a country with almost "complete" data for administrative divisions. The sample dataset was limited to a bounding box defined as: 41.836501°N to 41.948695°N, 12.436859°E to 12.626374°E. We extracted all municipal boundaries from the OSM dataset with tag value of "admin_level=9," which is used in the region of interest to define administrative subdivisions in Rome. Then, we added two additional attributes: *name_left* and *name_right*, which define the label content for the left and the right side of a polyline respectively.

The input parameters of our algorithm S, D_{max}, D_{step}, B_{min}, and B_{max} are measured in map units, which were pixels in our tests. Additionally, in our implementation, we used a quality threshold parameter Q_T. This parameter allowed us to control and eliminate candidate label positions that corresponded to poor and sloppy label placement. These potential label positions we considered unacceptable and omitted from the *position selection* procedure. Parameter Q_T takes values in the range [0,1], where a value of 1.0 corresponds to an ideal case. In the tests we used $D_{step} = 1$ and chose the value of the parameter K (see Phase II step 1) sequentially from the set [0.5, 0.55, 0.6, 0.65, 0.7, 0.75] until a label placement was found. Next, to evaluate each label position we used function (2) with two different sets of parameters, namely:

$$H_1(l) = 0.3\, g_{\text{PosDev}}(l) + 0.1\, g_{\text{BaseOffset}}(l) + 0.5\, g_{\text{GoodnessOfFit}}(l) + 0.1\, g_{\text{HorizAlign}}(l)$$

$$H_2(l) = 0.1\, g_{\text{PosDev}}(l) + 0.5\, g_{\text{BaseOffset}}(l) + 0.3\, g_{\text{GoodnessOfFit}}(l) + 0.1\, g_{\text{HorizAlign}}(l) \quad (8)$$

In function H_1 we give great weight to $g_{\text{GoodnessOfFit}}$ and nearly neglect the influence of $g_{\text{BaseOffset}}$ and $g_{\text{HorizAlign}}$. In function H_2 we give lowest priority to g_{PosDev} and $g_{\text{HorizAlign}}$, shift the importance of $g_{\text{GoodnessOfFit}}$ to second place, and give $g_{\text{BaseOffset}}$ the highest priority.

PERFORMANCE AND VISUALIZATION RESULTS

In the first set of experiments we used quality function H_1 and explored how the success rate (the number of potential label locations) decreases as the quality threshold Q_T and the position deviation $D_{\max}$ increase, as well as how much time the labeling took. We set the input parameters to $S = 400$, $B_{\min} = 2$, $B_{\max} = 8$. Then, taking into account the value of S and the length of each of the 46 polylines in the tested region, we calculated the maximum possible number of labels. This number was $m = 493$. It is worth noting that from each set V_j of the candidate positions $\overline{q}_{jk}$, we choose only one candidate. In Figure 11a we present the results of the experiment: the algorithm is able to place labels in 95% of the desired positions ($m = 493$) with $Q_T = 0.55$ and $D_{\max} = 80$. When the quality threshold is higher, namely $Q_T = 0.75$ and $D_{\max} = 1$, we observe 75% of the maximum possible number of labels. Therefore, we conclude that enlargement of the search space and a lowered quality threshold results in a higher rate of labeled positions. Furthermore, we also measured the algorithm's runtime, in order to determine the influence of the search space. Figure 11b illustrates a linear dependence. Our algorithm is able to find one label position in 7.602 milliseconds. Note that such performance makes the algorithm appropriate for usage in interactive and dynamic labeling (Been et al. 2006; Mote 2007).

In another test, we fixed the position deviation value $D_{\max} = 25$ and ran our algorithm several times by varying $B_{\max}$. The results of the tests are shown in Figure 12a, and illustrate the ability of the algorithm to increase the percentage of placements by increasing the maximum permissible distance between two coupled labels (n_l and n_r) on either side of the polyline. This possibility comes in handy in case of labeling extremely curved parts of a polyline.

Finally, we evaluated the dependence of the number of placed labels on the type of the quality function, running the same test for both functions H_1 and H_2. The results presented in Figure 12b show that the algorithm places more labels with function H_1 than with H_2. However, the number of labels is almost the same with higher values of $D_{\max}$.

In order to demonstrate that our algorithm is able to generate legible and cartographically plausible label placements, we prepared two sample maps (Figures 13 and 14). We utilized function H_1 for type placement in both maps. Figure 13 depicts a map which was labeled using

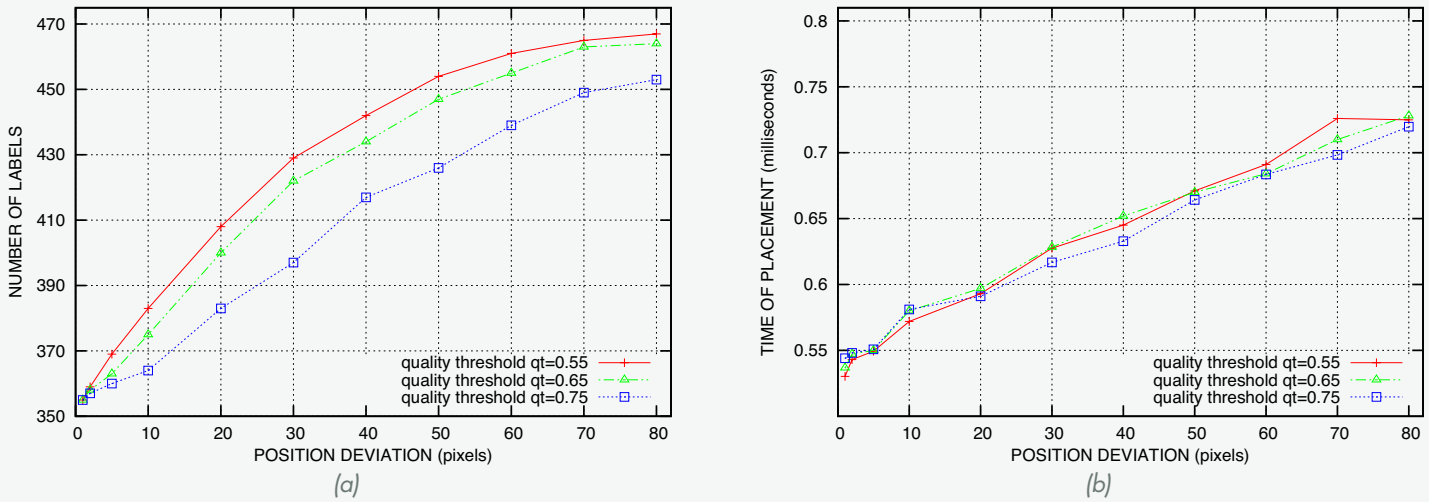


Figure 11. *Experimental results for different Q_T values. (a) Dependence of the number of label placements on $D_{\max}$. (b) Dependence of the runtime on $D_{\max}$; the Y-axis defines the time needed to find one label placement.*

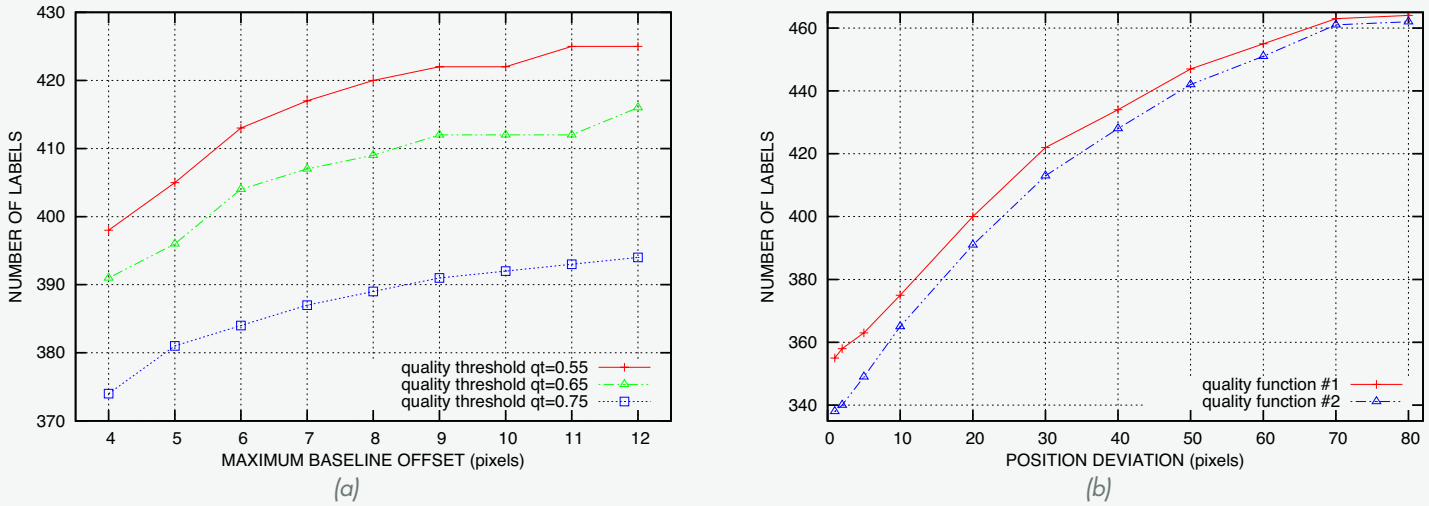


Figure 12. *Experimental results. (a) Dependence of the number of label placements on B_{max} when $B_{min} = 2$. Tested function is H_1. (b) Comparison of two quality functions H_1 and H_2 when $B_{min} = 2$, $B_{max} = 8$.*

Figure 13. *Labeling of municipal boundaries in Rome (7 labels). The input parameters are $S = 400$, $D_{max} = 1$, $B_{min} = 2$, $B_{max} = 4$, $Q_T = 0.75$. Data source: © OpenStreetMap contributors (2013, data licensed under ODbL).*

Figure 14. *Labeling of municipal boundaries in Rome (20 labels). The input parameters are $S = 400$, $D_{max} = 100$, $B_{min} = 2$, $B_{max} = 10$, $Q_T = 0.55$. Data © 2013 OpenStreetMap contributors (data licensed under ODbL).*

a small number of candidate positions and a high value of Q_T, while Figure 14 shows the same region with more candidate positions and a lower quality requirement. The algorithm placed 2.86 times more labels when using the second set of parameters.

The red marks in Figure 14 show that in some cases our algorithm places labels that overlap sinuous polylines, which contradicts guideline G9. This inability of the algorithm can be overcome by performing an additional post-processing step such as leaving out label pairs that intersect their polyline. To check whether a polyline and its labels intersect, we can utilize the algorithm by Bentley and Ottmann (1979) for reporting intersections between two sets of line segments in $O((n + k) \log n)$ time and $O(n)$ space, where n is the total number of line segments, and k is the number of itersections. Note that slightly faster but harder to implement algorithms for the same purpose exist, such as Mairson and Stolfi (1988) that requires $O(n \log n + k)$ time and $O(n)$ space. The polyline composes the first set of line segments, whereas eight line segments (eight for each pair of labels) bounding the label

comprise the second set. For the sake of the performance this check should be done only once after all potential labels are generated. Note that our implementation currently does not take this extra step.

Figure 15 illustrates a part of a map generated by *MapSurfer.NET*, which contains labels for points (e.g., settlements, motorway shields and peaks), curved lines (e.g., streets, rivers, boundaries) and areas (e.g., parks, lakes). This map demonstrates the possibility of using our algorithm as a part of a more general labeling algorithm (Edmondson et al. 1996).

A set of maps involving pairwise line labeling of boundaries are available online through a web map tile service (García et al. 2012) on *OpenMapSurfer* (**korona.geog.uni-heidelberg.de**). On this page the layers "OSM Roads" and "OSM Admin Boundaries" demonstrate the output of the algorithm on the OSM dataset for the whole globe.

CONCLUSIONS AND FUTURE WORK

In this paper we have introduced a new, efficient, and easily configurable algorithm for performing visually plausible and functional pairwise labeling of lines representing geographic boundaries. Our algorithm achieves two goals: it generates candidate positions and evaluates their quality according to cartographic guidelines for line labeling.

Figure 15. *A sample map containing administrative boundary labels together with other feature types such as roads, railways, districts, parks, etc. Data © 2013 OpenStreetMap contributors (data licensed under ODbL).*

The results of our experiments on a real-world dataset show that our algorithm is able to find candidates in 95% of desired positions with a certain set of input parameters. The runtime measurements confirm the high performance of the algorithm. Another advantage of the algorithm is that the generated candidate positions and the quality function can be used in a general map labeling algorithm such as that of Edmondson et al. (1996) that labels all feature types (e.g., points, lines and polygons) simultaneously. More precisely, the quality function can potentially be used as a component for a comprehensive quality function (van Dijk et al. 2002; Rylov and Reimer 2014a) which is employed by a combinatorial optimization algorithm (Christensen et al. 1995) to find the globally best and optimal label placement. We also believe that our algorithm can be easily reproduced and embedded in commercial or open source GIS toolkits (Steiniger and Hunter 2013).

It remains an open problem how to perform pairwise labeling of boundary lines using curved text as depicted in Figure 2, which is often more preferable. This task can be accomplished by exploiting a curve fitting procedure. Note that it will require a new method for candidate positions generation and the construction of another quality function. Moreover, both parts of the algorithm should be based on an adopted list of cartographic guidelines that need to be determined through a study (like in Reimer et al. [2015]) of formal principles commonly used in manual lettering. We think that some parts of our algorithm can be borrowed as a baseline for the construction of a new method.

In conclusion, we sincerely hope that our approach advances the development of more robust and efficient algorithms for labeling geographic boundaries.

REFERENCES

Barrault, M. 2001. "A Methodology for Placement and Evaluation of Area Map Labels." *Computers, Environment and Urban Systems* 25 (1): 33–52. doi: **10.1016/S0198-9715(00)00039-9**.

Barrault, M., and F. Lecordix. 1995. "An Automated System for Linear Feature Name Placement which Complies with Cartographic Quality Criteria." *Proceedings of the International Symposium on Computer-Assisted Cartography (AutoCarto 12),* 321–330.

Basoglu, U. 1982. "A new approach to automated name placement." *Proceedings of the International Symposium on Computer-Assisted Cartography (AutoCarto 5)*, 103–112.

Been, K., E. Daiches, and C. Yap. 2006. "Dynamic Map Labeling." *IEEE Transactions on Visualization and Computer Graphics* 12(5):773-780. doi: **10.1109/TVCG.2006.136**.

Bentley, J. L. and T. A. Ottmann. 1979. "Algorithms for Reporting and Counting Geometric Intersections." *IEEE Transactions on Computers* C–28 (9): 643–647. doi: **10.1109/TC.1979.1675432**.

Brewer, C. A. 2005. *Designing Better Maps: A Guide for GIS Users*. Redland: ESRI Press.

Chatterjee, S., and A. Hadi. S. 2006. "Simple Linear Regression." In *Regression Analysis by Example*, *4th Edition*. New Jersey: John Wiley & Sons.

Chirié, F. 2000. "Automated Name Placement With High Cartographic Quality: City Street Maps." *Cartography and Geographic Information Science* 27 (2): 101–110. doi: **10.1559/152304000783547902**.

Christensen, J., J. Marks, and S. M. Shieber. 1995. "An Empirical Study of Algorithms for Point-Feature Label Placement." *ACM Transactions on Graphics* 14 (3): 203–232. doi: **10.1145/212332.212334**.

Defense Mapping Agency. 1973. Series 1501 AIR, Sheet ND 48-12, Virachey, Cambodia; Vietnam, 1:250,000, Edition 3. Scan provided by University of Texas, Perry-Castañeda Library Map Collection. Accessed January 20, 2014. **http://www.lib.utexas.edu/maps/jog/vietnam/nd-4812_geo.pdf**.

van Dijk, S., M. van Kreveld, T. Strijk, and A. Wolff. 2002. "Towards an Evaluation of Quality for Names Placement Methods." *International Journal of Geographical Information Science* 16 (7): 641–661. doi: **10.1080/13658810210138742**.

Edmondson, S., J. Christensen, J. Marks, and S. Shieber. 1996. "A General Cartographic Labeling Algorithm." *Cartographica* 33 (4): 13–23. doi: **10.3138/U3N2-6363-130N-H870**.

ESRI. 2009. "Maplex for ArcGIS. An ESRI White Paper." **http://www.esri.com/library/whitepapers/pdfs/maplex-for-arcgis.pdf**.

Formann, M., and F. Wagner. 1991. "A Packing Problem with Applications to Lettering of Maps." *Proceedings of the 7th Annual Symposium on Computational Geometry*, 281–288. doi: **10.1145/109648.109680**.

García, R., J. P. de Castro, E. Verdú, M. J. Verdú, L. M. Regueras. 2012. Web Map Tile Services for Spatial Data Infrastructures: Management and Optimization. In *Cartography — A Tool for Spatial Analysis*, edited by C. Bateira, 25–48. doi: **10.5772/46129**.

Haklay, M., and P. Weber. 2008. "OpenStreetMap – User Generated Street Map." *Pervasive Computing, IEEE* 7 (4): 12–18. doi: **10.1109/MPRV.2008.80**.

Hirsch, S. A. 1982. "An Algorithm for Automatic Name Placement around Point Data." *American Cartographer* 9 (1): 5–17. doi: **10.1559/152304082783948367**.

Imhof, E. 1962. "Die Anordnung der Namen in der Karte." *Internationales Jahrbuch für Kartographie II*, 93–129. Zürich: Orell-Füssli Verlag.

———. 1975. "Positioning Names on Maps." *The American Cartographer* 2 (2): 128–144. doi: **10.1559/152304075784313304**.

Kakoulis, K. G., and I. G. Tollis. 2003. "A Unified Approach to Automatic Label Placement." *International Journal of Computational Geometry and Applications* 13 (1): 23–60. doi: **10.1142/S0218195903001062**.

Kern, J. R., and C. A. Brewer. 2008. "Automation and the Map Label Placement Problem: A Comparison of Two GIS Implementations of Label Placement." *Cartographic Perspectives* 60: 22–45. doi: **10.14714/CP60.230**.

van Kreveld, M., T. Strijk, and A. Wolff. 1999. "Point Labeling with Sliding Labels." *Computational Geometry: Theory and Applications* 13 (1): 21–47. doi: **10.1016/S0925-7721(99)00005-X**.

Mairson, H. G., and J. Stolfi. 1988. "Reporting and Counting Intersections Between Two Sets of Line Segments." In *Theoretical Foundations of Computer Graphics and CAD, Volume 40*, edited by R. A. Earnshaw. Berlin: Springer Berlin Heidelberg. 307–325. doi: **10.1007/978-3-642-83539-1_11**.

Marks, J., and S. Shieber. 1991. "The computational complexity of cartographic label placement." Technical Report TR-05-91, Harvard University, March.

Mote, K. 2007. "Fast Point-Feature Label Placement for Dynamic Visualizations." *Information Visualization* 6 (4): 249–260. doi: **10.1057/palgrave.ivs.9500163**.

National Imagery and Mapping Agency. 1962. Series Z901, Sheet S6/13 SW-4B, Matadi, 1:12 500, Edition 2 – AMS, Prepared by the Army Map Service (TV). Scan provided by University of Texas, Perry-Castañeda Library Map Collection. Accessed January 20, 2014. **http://www.lib.utexas.edu/maps/africa/txu-oclc-55853021-matadi-1959.jpg.**

Rabello, R. L., G. R. Mauri, G. M. Ribeiro, and L. A. N. Lorena. 2014. "A Clustering Search Metaheuristic for the Point-Feature Cartographic Label Placement Problem." *European Journal of Operational Research* 234: 802–808. doi: **10.1016/j.ejor.2013.10.021**.

Ramm, F., J. Topf, and S. Chilton. 2010. *OpenStreetMap: Using and Enhancing the Free Map of the World*. UIT Cambridge.

Regnauld, N, S. Lessware, C. Wesson, and P. Martin. 2013. "Deriving Products from a Multiple Resolution Database using Automated Generalisation at Ordnance Survey." *26th International Cartographic Conference*, August 25–30, Dresden, Germany.

Reimer, A., A. van Goethem, M. Rylov, M. van Kreveld, and B. Speckmann. 2015. "A Formal Approach to the Automated Labelling of Groups of Features." *Cartography and Geographic Information Science*. Forthcoming. doi: **10.1080/15230406.2015.1053986**.

Revell, P., N. Regnauld, and G. Bulbrooke. 2011. "OS Vectormap District: Automated Generalisation, Text Placement and Conflation in Support of Making Public Data Public." *25th International Cartographic Conference*, July 3–8, Paris, France.

van Roessel, W. 1989. "An Algorithm for Locating Candidate Labeling Boxes within a Polygon." *American Cartographer* 16 (3): 201–209. doi: **10.1559/152304089783814034**.

Rylov, M. A., and Reimer, A. W. 2014a. "A Comprehensive Multi-Criteria Model for High Cartographic Quality Point-Feature Label Placement." *Cartographica* 49 (1): 52–68. doi: **10.3138/carto.49.1.2137**.

———. 2014b "A Practical Algorithm for the External Annotation of Area Features." *The Cartographic Journal*. doi: **10.1179/1743277414Y.0000000091**.

———. 2014c "Improving Label Placement Quality by considering Basemap Detail with a Raster-Based Approach." *GeoInformatica* 19 (3): 463–486. doi: **10.1007/s10707-014-0214-6**.

Steiniger, S., and A. J. S Hunter. 2013. "The 2012 free and open source GIS software map – A guide to facilitate research, development and adoption." *Computers, Environment and Urban Systems* 39: 136–150. doi: **10.1016/j.compenvurbsys.2012.10.003**.

Strijk, T. W., and M. J. van Kreveld. 2002. "Practical Extensions of Point Labeling in the Slider Model." *GeoInformatica* 6 (2): 181–197. doi: **10.1023/A:1015202410664**.

Verner, O. V., R. L. Wainwright, and D. Schoenefeld. 1997. "Placing Text Labels on Maps and Diagrams using Genetic Algorithms with Masking." *INFORMS Journal on Computing* 9 (3): 266–275. doi: **10.1287/ijoc.9.3.266**.

Wolff, A., L. Knipping, M. van Kreveld, T. Strijk, and P. K. Agarwal. 2001. "A Simple and Efficient Algorithm for High-Quality Line Labeling." *Innovations in GIS VII: GeoComputation*, 11: 147–159.

Wolff, A., T. Strijk. 2009. The Map-Labeling Bibliography. Available at **http://i11www.iti.uni-karlsruhe.de/~awolff/map-labeling/bibliography/maplab_author.html**.

Wood, C. H. 2000. "Descriptive and Illustrated Guide for Type Placement on Small Scale Maps." *The Cartographic Journal* 37 (1): 5–18. doi: **10.1179/caj.2000.37.1.5**.

Yamamoto, M., G. Camara, and L. A. N. Lorena. 2002. "Tabu Search Heuristic for Point-Feature Cartographic Label Placement." *Geoinformatica* 6 (1): 77–90. doi: **10.1023/A:1013720231747**.

Yoeli, P. 1972. "The Logic of Automated Map Lettering." *The Cartographic Journal* 9 (2): 99–108. doi: **10.1179/caj.1972.9.2.99**.

Zhang, Q. and L. Harrie. 2006. "Placing Text and Icon Labels Simultaneously: A Real-Time Method." *Cartography and Geographic Information Science* 33 (1): 53–64. doi: **10.1559/152304006777323127**.