

曲線データの圧縮に関するアルゴリズムと実験的評価

安田 祐一*

今井 桂子†

* 中央大学大学院 理工学研究科 情報工学専攻

† 中央大学 理工学部 情報工学科

概要: コンピュータグラフィックス, 地理情報処理において曲線を表すのにデータの点列の集合 (折れ線) が用いられる. 与えられた点列を別のより粗い折れ線で近似することがデータ圧縮のためにおこなわれている. これまでも様々な圧縮の手法が提案されている ([1, 2, 4]) が, 本稿ではそれらとは異なる許容される誤差基準を用いた手法を提案し, 比較, 考察をおこなった. さらに, より実用性を高めるために圧縮率を指定する手法や曲線の密集度を考慮した手法を提案する. また, 東京 23 区内地下鉄を曲線の集合とし, そのデータを圧縮する計算機実験によって, アルゴリズムの評価をおこなった.

キーワード: 曲線データの圧縮, 折れ線, 許容誤差.

Algorithms for Compressing Piecewise Linear Data with Computational Results

Yuichi YASUDA*

Keiko IMAI †

* Information and System Engineering Course,

Graduate School of Science and Engineering, Chuo University

† Department of Information System and Engineering, Chuo University

Abstract : In computer graphics, cartography and geographic information systems, a curve is represented as a set of points, and consists of many small segments, namely piecewise linear data. It is important to compute approximate linear data for a given curve in order to compress the data. Various methods for approximating curves have been proposed [1, 2, 4]. In this paper, we propose new algorithms for the problem and computational results for Tokyo subway map data are also shown.

keyword : compressing data, piecewise line, error tolerance.

1 はじめに

一般にコンピュータグラフィックスや地理情報処理で曲線を取り扱う場合には, データの点列の集合で与えられる. また, それを実際に画面上に表示するには, それらの点列を線分で結ぶことにより得られる. 画面の精度以上に細かいデータは, ディスプレイに表示することができない. また, 一般にこの様な複雑な曲線のデータは膨大な量となるため, データ量の圧縮や地図の大きさの縮小のため, 曲線 (点列を線分で結ぶことによってできる折れ線) をある誤差基準のもとでできるだけ少数の線分の折れ線で近似する必要がある.

点列を $p_0, p_1, \dots, p_n$ としたとき, それらを線分で結んだ曲線を $P = (p_0, p_1, \dots, p_n)$ と書く. 一般に曲線データの圧縮として, 以下の 2 つの場合が考えられる.

1. オリジナルのデータ点の集合 P から直接点を間引く. その結果の部分点列の集合を結びショートカットをおこなうことにより圧縮された曲線 Q を得る.
2. 何らかの近似関数で補間する. この場合データ点列は P の部分集合とはならない.

本稿では, 1 の手法について, まず既存のアルゴリズムを実装し, 計算機実験をおこなった. 次に, 本研究で提案した誤差基準を用いたアルゴリズムを実装し, 既存の結果との比較, 考察をおこなった. これまでは, 対象となる曲線が交わらない場合が考えられてきたが, 本稿では交差する場合の解法も提案する. また, より実用性を高めるために [1] で提案されたあらかじめ定めた圧縮率を用いる手法を提案アルゴリズムに適用させた実験や点列の密集度を考慮する圧縮法を提案し, 実験を

おこなった。

入力データとして、東京都 23 区内の地下鉄の路線図を曲線の集合と見なし、そのデータを圧縮する実験もおこなったのでそれを報告する。

2 圧縮の定義と許容誤差

ここでは、圧縮の定義と許容誤差、また圧縮された結果を評価をする基準について述べる。まず、曲線データの圧縮を次のように定義する。

定義 1

p_i ($i = 0, 1, \dots, n$) を平面上の点とし、曲線 P を点列 $P = (p_0, p_1, \dots, p_n)$ で表現する。つまり曲線 P は、この集合に含まれる点列を、始点 p_0 から終点 p_n まで順に結んだ、ひと続きの折れ線と考える。この曲線の圧縮をおこなうとは、 $P \supseteq Q$ で以下の条件を満たす点列 $Q = (q_0, q_1, \dots, q_m)$ ($m \leq n$) を得ることである。図 1 は、圧縮の例である。

1. $q_0 = p_0, q_m = p_n$ である。
2. $q_i = p_j$ ($0 \leq i < m, 0 \leq j < n$) のとき、ある k ($j < k \leq n$) に対し $q_{i+1} = p_k$ を満たす。

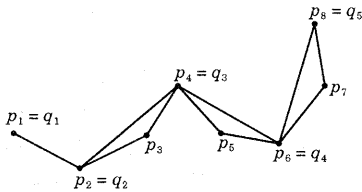


図 1. 曲線データ圧縮の例

2.1 誤差の定義

曲線データの圧縮をおこなう場合、用いる誤差基準などにより圧縮結果そのものが非常に異なってくる。ここでは、問題を統一的に扱うために、大きく分けて次のような 2 つの定義を採用する。

- 基準となる角度を決める。それ以上曲線が曲がっていればショートカットをしない。また、その基準となる角度を許容角度誤差と呼ぶ。
- 基準となる距離を決める。それ以上誤差がでるようならばショートカットをしない。また、その基準となる距離を許容距離誤差と呼ぶ。

2.2 性能の評価

ここでは、圧縮の各種アルゴリズムの精度を評価する方法を挙げる。

評価法 1. オリジナルのデータから圧縮した結果の折れ線に至る距離の総和を見る。

評価法 2. オリジナルデータの折れ線と圧縮した結果のデータの折れ線が囲む総面積を見る。

いろいろな評価の方式があるが、上に挙げた評価方法は、圧縮結果がオリジナルの図形とどのくらいかけ離れているかの視覚的な感じを的確にとらえている。

3 長方形被覆を用いたアルゴリズム

本稿で提案するアルゴリズムの性能を評価するために、これまでに提案されている点列の長方形被覆問題について説明する [3]。

幅 w の長方形 $R_1, \dots, R_r$ が点列 $P_1, \dots, P_n$ を被覆するとは、 $q = 1, \dots, r$ に対して長方形 R_q が点 $P_{i(q-1)}, \dots, P_{i(q)}$ ($i(q-1) \geq i(q)$) を覆うことをいう。ただし、 $i(0) = 1, i(r) = n$ で、長方形の長さは十分長いとする。このとき問題は、与えられた点列を被覆する長方形の列で、長方形の数が最少のものを求めることである。図 2 は、長方形被覆の例である。このとき、長方形の幅 w は許容距離誤差を指定していることになる。この問題に対しては、次の素朴な貪欲算法が最適解を与えることが容易に示される。

1. $i := 1$
2. $i^* := \max\{j \mid i < j \leq n, \text{点 } P_i, \dots, P_j \text{ を覆う幅 } w \text{ の長方形が存在}\}$;
3. 点 $P_i, \dots, P_{i^*}$ を幅 w で覆う;
 $i = n$ なら停止; そうでなければ $i := i^*$ として 2 へ戻る。

この算法で問題となるのは、ステップ 2 であるが、この計算が $O(n \log n)$ でおこなえることが [3] で述べられている。

本研究の圧縮の手法は、折れ線から点を間引く手法であるから長方形で覆われた点列の添字の最も小さい頂点と最も大きい頂点を結ぶことによって圧縮をおこなうことにする。

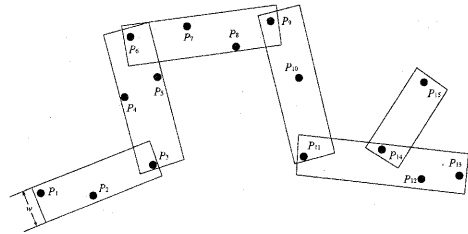


図 2. 点列の長方形被覆の例

おこなった。

入力データとして、東京都 23 区内の地下鉄の路線図を曲線の集合と見なし、そのデータを圧縮する実験もおこなったのでそれを報告する。

2 圧縮の定義と許容誤差

ここでは、圧縮の定義と許容誤差、また圧縮された結果を評価をする基準について述べる。まず、曲線データの圧縮を次のように定義する。

定義 1

p_i ($i = 0, 1, \dots, n$) を平面上の点とし、曲線 P を点列 $P = (p_0, p_1, \dots, p_n)$ で表現する。つまり曲線 P は、この集合に含まれる点列を、始点 p_0 から終点 p_n まで順に結んだ、ひと続きの折れ線と考える。この曲線の圧縮をおこなうとは、 $P \supseteq Q$ で以下の条件を満たす点列 $Q = (q_0, q_1, \dots, q_m)$ ($m \leq n$) を得ることである。図 1 は、圧縮の例である。

1. $q_0 = p_0, q_m = p_n$ である。
2. $q_i = p_j$ ($0 \leq i < m, 0 \leq j < n$) のとき、ある k ($j < k \leq n$) に対し $q_{i+1} = p_k$ を満たす。

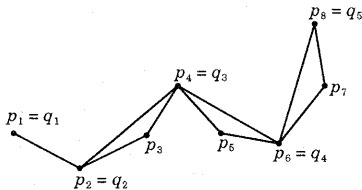


図 1. 曲線データ圧縮の例

2.1 誤差の定義

曲線データの圧縮をおこなう場合、用いる誤差基準などにより圧縮結果そのものが非常に異なってくる。ここでは、問題を統一的に扱うために、大きく分けて次のような 2 つの定義を採用する。

- 基準となる角度を決める。それ以上曲線が曲がっていればショートカットをしない。また、その基準となる角度を許容角度誤差と呼ぶ。
- 基準となる距離を決める。それ以上誤差がでるようならばショートカットをしない。また、その基準となる距離を許容距離誤差と呼ぶ。

2.2 性能の評価

ここでは、圧縮の各種アルゴリズムの精度を評価する方法を挙げる。

評価法 1. オリジナルのデータから圧縮した結果の折れ線に至る距離の総和を見る。

評価法 2. オリジナルデータの折れ線と圧縮した結果のデータの折れ線が囲む総面積を見る。

いろいろな評価の方式があるが、上に挙げた評価方法は、圧縮結果がオリジナルの図形とどのくらいかけ離れているかの視覚的な感じを的確にとらえている。

3 長方形被覆を用いたアルゴリズム

本稿で提案するアルゴリズムの性能を評価するために、これまでに提案されている点列の長方形被覆問題について説明する [3]。

幅 w の長方形 $R_1, \dots, R_r$ が点列 $P_1, \dots, P_n$ を被覆するとは、 $q = 1, \dots, r$ に対して長方形 R_q が点 $P_{i(q-1)}, \dots, P_{i(q)}$ ($i(q-1) \geq i(q)$) を覆うことをいう。ただし、 $i(0) = 1, i(r) = n$ で、長方形の長さは十分長いとする。このとき問題は、与えられた点列を被覆する長方形の列で、長方形の数が最少のものを求めることである。図 2 は、長方形被覆の例である。このとき、長方形の幅 w は許容距離誤差を指定していることになる。この問題に対しては、次の素朴な貪欲算法が最適解を与えることが容易に示される。

1. $i := 1$
2. $i^* := \max\{j \mid i < j \leq n, \text{点 } P_i, \dots, P_j \text{ を覆う幅 } w \text{ の長方形が存在}\}$;
3. 点 $P_i, \dots, P_{i^*}$ を幅 w で覆う;
 $i = n$ なら停止; そうでなければ $i := i^*$ として 2 へ戻る。

この算法で問題となるのは、ステップ 2 であるが、この計算が $O(n \log n)$ でおこなえることが [3] で述べられている。

本研究の圧縮の手法は、折れ線から点を間引く手法であるから長方形で覆われた点列の添字の最も小さい頂点と最も大きい頂点を結ぶことによって圧縮をおこなうことにする。

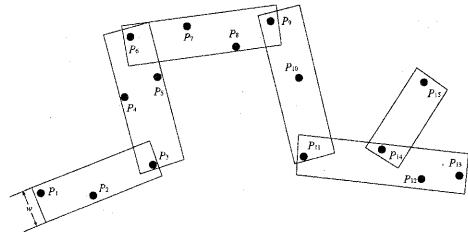


図 2. 点列の長方形被覆の例

トカットができなくなるまで繰り返すことによってショートカット候補を得ることができる。

次に、 ε -Voronoi 図というものを求める。この ε -Voronoi 図とは、折れ線 C からの距離 ε 内にある全ての頂点からなる領域を考える。すなわち、領域 $C \oplus B_\varepsilon$ を考える。 B_ε は半径 ε の球で、 $\oplus$ は標準的な Minkowski 和である。 ε -Voronoi 図は以下の式で表される。

$$d_\varepsilon = \begin{cases} d(p, q) & d(p, q) \leq \varepsilon \\ \infty & \text{otherwise} \end{cases}$$

図 5 (b) は、(a) が入力されたときの ε -Voronoi 図の例である。

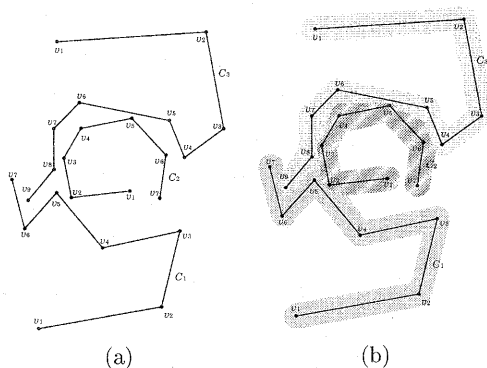


図 5. ε -Voronoi 図の例

最後に、前のステップで求めたショートカットの候補の中から ε -Voronoi 領域に収まりきっていない線分を消去し、最短パスを計算する。最短パスの計算は 4.1 節と同様である。

5 圧縮する割合を指定するアルゴリズム

ここでは、桧山、花田、今井 [1] が提案した圧縮する割合を指定する GEM (Global Eliminating Methods) 法を本研究で提案したアルゴリズムに適用させた結果を述べる。

これまでの圧縮法は、許容誤差を事前に決めなければならない。しかし、扱うデータの大きさや単位は簡単には分からないことが多い。一方、データの個数はあらかじめ分かっていることが多いので、この方式のほうが便利である。

まず、[1] の GEM 法のアルゴリズムについて説明する。GEM 法とは、まず $i = 1$ から $n - 1$ までの隣り合う 3 点 p_{i-1}, p_i, p_{i+1} ($i = 1, 2, \dots, n - 1$) に対し、点 p_i から線分 $\overline{p_{i-1}p_{i+1}}$ に至る距離 d_i を求める。ただし、線分 $\overline{p_{i-1}p_{i+1}}$ に直交し $p_{i-1}p_{i+1}$ のそれぞれの通る 2 本の直線に p_i がはさまれて

いない場合は、 $d_i = \min(\overline{p_i p_{i+1}}, \overline{p_i p_{i-1}})$ とする。 $i = 1$ から $n - 1$ までのすべての d_i の中で最も小さい点を間引き、指定した点数になるまで繰り返す手法である。

本稿では、これを角度誤差を用いて以下の様なアルゴリズムを提案する。隣り合う 3 点 p_{i-1}, p_i, p_{i+1} ($i = 1, 2, \dots, n - 1$) のなす角 $\angle p_{i-1}p_i p_{i+1} = a_i$ を求める。 $i = 1$ から $n - 1$ までのすべての a_i の中で最も大きい点を間引くとする。以下残った点でこの操作を繰り返す。ここでは、4 章で述べた基準線を用いた改良をおこなわないとする。 m 点残す ($n - m$ 点間引く) アルゴリズムは以下ようになる。

```

k = 0;
repeat
    max  $a_i = \angle p_{i-1}p_i p_{i+1}$  なる  $i$  を求める;
     $p_i$  を間引く;
    k = k + 1;
until k > n - m;

```

6 密集度を考慮した圧縮アルゴリズム

この節では、より実用性を高めるため点列の密集度を考慮に入れた手法を提案する。一般に曲線データの形状が複雑になったり、複数の曲線が集中して集まってしまうとデータ量が大きくなってしまふ。そこで、バケットを用いて点列の密集度を求め、それに応じて許容誤差の値、つまり圧縮率を変えることによって、データをユーザの必要に応じた情報量に圧縮する手法について述べる。

バケットを用いる手法とは、入力されたデータを任意の領域に分割し、その領域に含まれる点数を数え、それを密集度とし、それに応じて各バケットの各頂点に任意の許容誤差を決めることにより、密集度に応じた曲線の圧縮をおこなうことができる。図 6 は、バケットの例である。ここでは、点列の密集度と長方形被覆を合わせた手法を用いる。アルゴリズムを以下に示す。

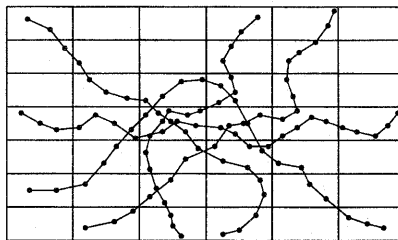


図 6. バケットを用いる手法の例

1. 与えられた点列を幅 w , 高さ h のバケットに分割する;
2. 各バケットの頂点数を数え, 各頂点にそれに応じた許容誤差を与える;
(各頂点の許容誤差) $= \{g_1, g_2, \dots, g_n\}$;
3. すべての点列に以下を繰り返す;
4. $i := 1$
5. $i^* := \max\{j | i < j \leq n, \text{点 } P_i, \dots, P_j \text{ を覆う幅 } g = \min\{g_t | i < t \leq j\} \text{ の長方形が存在}\}$;
6. 点 $P_i, \dots, P_{i^*}$ を幅 g で覆う;
 $i = n$ なら停止; そうでなければ $i := i^*$ として 5 へ戻る;

7 計算機実験

本稿で述べた手法について, それぞれ東京 23 区内の地下鉄路線図を圧縮する計算機実験をおこなった. 入力データは, 国土地理院の数値地図 2500 (平成 9) から地下鉄路線図を抽出した. 路線の数は 11 本, 点の総数は 3428 点となっている. まず, 4 節で述べた 2 つの結果と 3 節の長方形被覆の結果を表 1 にまとめる.

許容角度誤差を用いたアルゴリズムの結果については, 用いている許容誤差が異なるので, 安易に長方形被覆の結果と比較することはできないが, それほど大差ない結果をより少ない計算回数で求めることができる. しかし, うまく許容角度誤差を設定することができなければ, 精度の悪い圧縮結果になってしまうので, 注意しなければならない. 図 8, は入力地下鉄路線図 (図 7) を許容角度誤差 ($145^\circ, 175^\circ$) で圧縮した結果である. 次に, ϵ -Voronoi 図を用いたアルゴリズムの結果について述べる. 長方形被覆と比べると, 圧縮後の頂点数が多いのが分かる. 原因としては, ϵ -Voronoi 領域の中でだけショートカットを許すということと, 一度折れ線を切断し圧縮後につなげる手法をとっているため余計な頂点が増えると考えられる.

次に, 圧縮割合を指定する手法について述べる. GEM 法と提案手法を表 2 にまとめる. 性能の評価として, 2.2 節で述べた評価法の他に圧縮後のそれぞれの線分から一番遠かった点までの距離の平均を求め, 評価基準として用いた. これは, 圧縮結果の距離誤差がどの程度あるかを評価するためである. 結果を比較すると距離誤差の平均, 面積の誤差ともに本研究で提案した手法の方が多少よい結果を得た. これは, GEM 法は [1] によるとジグザグな曲線を入力して点数を半分程度残す場合に

よい結果を得られるとなっている. 今回用いた入力データは, 比較的なだらかな曲線である. このことから, なだらかな曲線が入力されるときは角度誤差を応用した手法を用いた方が有効であると考えられる.

最後に, 密集度を考慮した手法について述べる. 密集度が 500 以上の領域の許容誤差を変化させ実験をおこなった. 図 10 は密集度 500 以上の領域 (網目の領域) の許容距離誤差を 5 とし, その他の領域を 50 とした実験結果を拡大したものである. 図 9 と比べると正しい結果が得られたことがわかる. 曲線の形状によってデータの量に増減があるので密集度に応じた誤差設定をおこなえば, ユーザにとって必要に応じた結果を得ることができる.

8 おわりに

本研究では, 数値地図データから路線図を曲線データとして抽出し, 曲線データの圧縮アルゴリズムを提案した. また, 東京都 23 区内の地下鉄路線図に対する実験をおこない, 提案した手法の実用性を示した. 地下鉄路線図のデータには多少かたまりがあるが, 今回使用した数値地図 2500 への適用としては, 良い結果が得られた. このことから今回提案した手法は, 一般の地図データに対しても適用できると思われる.

今回提案した手法と既存の手法の実験結果を比べてみても, 大きな差のない結果を得ることができたので, 今回の実験は成功したと思われる. 実験で使用した様々なパラメタ (許容誤差, 圧縮率) を入力する曲線データによって使い分けることによってより良い結果をえることができる.

文献

- [1] 桧山澄子, 花田孝郎, 今井仁司: 実用的な曲線データの間引き法, 日本応用数学会論文誌, (1993), Vol.3, No.2, pp.85-104.
- [2] 今井浩: 直線・折れ線近似に関する組み合せの幾何的解法, 第8回数理計画シンポジウム論文集, (1987), pp.69-78.
- [3] H. Inai and M. Iri: Computational-Geometric Methods for Polygonal Approximations of a Curve, *Computer Vision, Graphics, and Image Processing*, (1986), Vol.36, pp.31-41.
- [4] N. Mustafa and E. Koutsofios: Hardware-Assisted View-Dependent Map Simplification, *Proceedings of the 17th Annual Symposium on Computational Geometry*, (2001), pp.50-59.

表 1. 既存の手法と提案手法のまとめ

角度誤差を用いた手法	許容角度誤差	頂点数	評価法 1 (総距離)	評価法 2 (総面積)
	(145°, 175°)	883	3666	138526
	(120°, 150°)	359	12885	3145380
	(100°, 130°)	219	16320	4924532
ϵ -Voronoi 図を用いた手法 括弧内は長方形被覆の結果	許容距離誤差	頂点数	評価法 1 (総距離)	評価法 2 (総面積)
	5	905 (897)	3343 (3728)	128549 (139602)
	50	374 (365)	11876 (12895)	2893532 (3145465)
	100	235 (219)	15052 (16318)	4794483 (4924644)

表 2. 提案手法と GEM 法の比較

圧縮割合	70%	80%	90%	95%	
角度誤差の応用	4.02	8.25	50.35	137.88	距離誤差の平均 面積の誤差
	118234	145351	2864337	5683296	
GEM 法	4.23	8.46	52.12	140.09	距離誤差の平均 面積の誤差
	118466	146832	2908241	5684327	

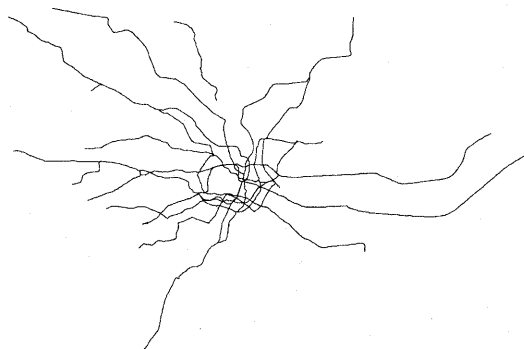


図 7. 入力する地下鉄路線図

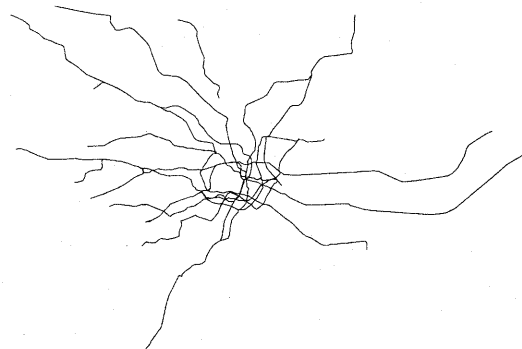


図 8. 許容角度誤差 (145°, 175°)

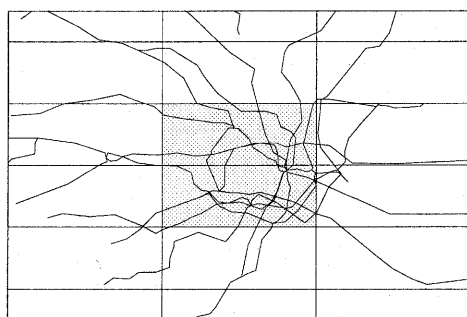


図 9. 許容誤差 50 とした結果

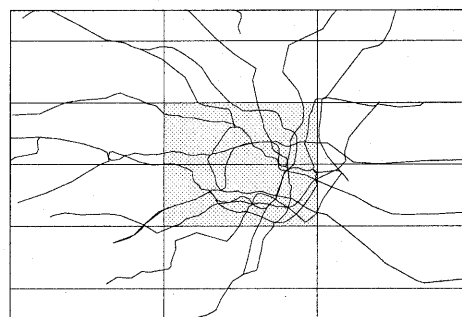


図 10. 密集度 500 以上の領域の誤差 5