

ストロークネットワークを用いた道なり優先経路探索 システムの実現

新帯 里奈¹ 山本 大介¹ 高橋 直久¹

概要：我々はこれまでに道なり道路（ストロークと呼ぶ）の概念を厳密に定義しストロークに基づくストロークネットワークを構築し，その上での道なり優先経路探索手法を提案した．しかし，この手法はストローク数最少の経路を与えるが，ノード数が多いとストロークネットワークの構築に多大な時間を要するという問題がある．本稿では，道路ネットワークから交差点での成す角に従って求めたストロークのデータベース（SDB）から基準長以上の長さがあるストロークからなる道路ネットワーク（ストロークネットワークと呼ぶ）を求め，ストロークネットワークデータベース（SNDB）を構築する．SNDB から基準長の異なる3種類のストロークネットワークを抽出し，それらを合成して，場所により密度の異なる道路ネットワークを構築する．構築した道路ネットワーク上で最少ストローク数最短経路探索法（SPMS 法）で道なり優先の経路を求めるシステムを開発した．SNDB を用いることで，ストロークネットワークの構築時間が1秒以下に短縮できることが分かった．

1. はじめに

カーナビやルート検索のアプリケーションにおける経路探索システムでは，様々な条件で経路探索ができる．遠距離の道なり優先の経路を求めたい場合，長距離に渡って利用される道路は幹線道路のような長い道なり道路であることが多い．しかし，「道なり」という概念はあいまいであり，人によって道なりでないと感じてしまう問題がある．また，幹線道路上に出発地や目的地が無ければ幹線道路とつながる経路を求める必要がある．道なり優先の経路探索においては，道なり道路（ストロークと呼ぶ）を厳密に定義し，ストロークに基づいた経路探索を行う必要がある．このとき，全ての道路を用いてしまうと，多くの交差点が探索範囲に含まれてしまう問題がある．

我々はこれまでにストロークの概念を厳密に定義しストロークに基づくストロークネットワークを構築し，その上での道なり優先経路探索手法を提案した [1]．この手法は，出発点と到着点の2点を含む範囲内で長大なストロークからストロークネットワーク（SN1）を構築する．しかし，その2点が長大なストローク上に無ければ，その2点とSN1とをつなぐ道路ネットワークが必要となり，経路探索する際に2種類のネットワークに分けて行わなければならない．このアプローチとして，出発点や到着点の周辺のストロークの密度を上げる．出発点や到着点の周辺にお

ける短小なストロークも含めたストロークネットワーク（SN2,SN3）を構築する．3つのストロークネットワーク（SN1,SN2,SN3）を合成し，1つのストロークネットワークを構築する．合成したストロークネットワーク上でストローク数最少の経路探索をする．ストロークネットワークを用いることで交差点の数を減らし，ストローク数優先の経路を与える．しかし，ノード数が多くなると経路探索よりもストロークネットワークの構築に多大な時間を要するという問題がある．

経路探索において，文献 [2] ではできるだけ道なり経路を探索する手法を実現しているが，最も道なりに従う経路を求めてはいない．また，ノードを隣接する道路の数だけ分割して経路探索を行うため，経路探索に時間を多くに要する問題がある．

本稿では，道路ネットワークから交差点での成す角に従って求めたストロークのデータベース（SDB）から基準長以上のストロークを抽出し，それらのストロークからなるストロークネットワークを求め，ストロークネットワークデータベース（SNDB）を構築する．そして，SNDB から抽出したストロークネットワーク上で最少ストローク数最短経路探索法（SPMS 法）で道なり優先の経路を求めるシステムを提案する．これにより，多大な時間を要していたストロークネットワークの構築時間を短くし，ストロークネットワーク上での道なり優先の経路探索を可能にする．

¹ 名古屋工業大学大学院 工学研究科 情報工学専攻

2. 関連研究

経路探索に関して、多くの研究がある。道路の特徴をコストにして渋滞を回避する経路を探索する研究 [3]、ダイクストラの最短経路探索から高速化アルゴリズムを開発する研究 [4] などがある。文献 [2] では、右左折等のコストを考慮した道路網ネットワークのデータ構造を提案し、そのデータ構造を用いて出発点から任意の到着点までの最小コスト経路を探索する手法を提案している。全ての交差点の右左折リンクのコストを直進リンクのコストより大きく設定すると右左折の回数が少ない経路が得られる。しかし、この経路の右左折の回数が最少となるとは限らない。

文献 [5] では、道路網を道路の役割を基準に階層化して表現し、階層化により生成されたブロック単位で最短経路を探索する手法を提案している。道路網の階層は上位層の道路が幹線道路や高速道路などの広く長い道路で長距離の移動に適した道路、下位層は上位層が表現する領域を上位層の道路で区切り、それから得られる多角形領域内の道路網としている。出発点を含む下位層から2点を含む上位層、到着点を含む下位層へ順に最短経路探索を行うことで、探索するノードを減らし、長距離の移動ほど上位層の道路が含まれる経路が得られる。しかし、出発点・到着点のいずれかを含む下位層と2点を含む上位層に階層の差があるとその分さらに時間がかかってしまう。本研究は、上位層にあるような道路からなるストロークネットワークと、下位層にある道路からなるストロークネットワークを重ね合わせ1つのストロークネットワークにし、その上でストロークに基づいて道なり優先である経路を求めることを目的とするため、これらとは大きく異なる。

3. 提案システムの概要

3.1 提案システムの特徴

本研究では、道路ネットワークから交差点での成す角に従ってストロークを求め、図1に示すような基準長以上の長さがあるストロークからなる道路ネットワーク（ストロークネットワークと呼ぶ）を構築する。また、ストロークネットワークについては、ネットワーク内に存在するストロークの割合を密度とする。基準長が長いストロークネットワーク（SN）は、基準長の短いSNに比べて、道路の密度が低くなる。

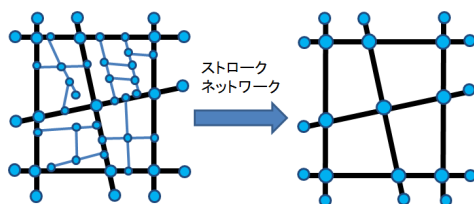


図1 ストロークネットワーク

本研究で提案するシステムの特徴を以下に挙げる。

特徴1【SN 作成機能】 指定された長さ L 以上のストロークをストロークデータベース (SDB)[6] から取得してストロークネットワーク SN(L) を作成する。これにより、交差点数の少ない道路ネットワークを扱えるようにする。

特徴2【SN 合成機能】 基準長の異なる2つのストロークネットワーク SN($L1$) と SN($L2$) を合成する。これにより、場所により道路密度の異なる道路ネットワークを構築できるようになり、経路探索が SN($L1$) と SN($L2$) でそれぞれに行うのではなく SN 合成機能により作成した SN だけでまとめて行えるようになる。

特徴3【SN 探索機能】 SN 合成機能により作成した SN 上で最少ストローク数最短経路探索法 (SPMS 法)[7] によりストローク数が最小かつ最短の経路を探索する。これにより、目的に応じて道路密度を調整した SN を用いて効率的に道なり道路を探索できるようになる。

3.2 提案システムで使用するデータ

提案システムにおいて使用するデータは、道路データとストロークデータである。

3.2.1 道路データ構造

道路のデータ構造を図2に示す。道路データには、ポイント、ノード、アークの3つのデータがある。ポイントは、道路上の点を表し、位置座標（緯度、経度）からなる。道路の形状は図2のように道路上の隣接ポイントを順番につないでできる折れ線で近似的に表現される。ノードとは、道路ネットワークの交差点である。アークは始点ノードから終点ノードに至る道路であり、道路上のポイントの系列で表す。リンクはアークとノードをまとめたものであり、始点ノードと終点ノードの組をエッジと呼ぶ。ノードとエッジによるグラフ表現を図3に示す。

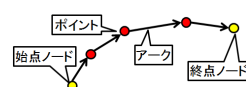


図2 実道路データ構造



図3 トポロジー表現

3.2.2 道路データベース

提案システムでは、道路データに対し、リンクごとに対応するノードとアークをまとめて、表1に示す形式のリンクテーブルを作成する。このとき、リンクは表2に示す9種類の道路クラスに分類し、アークは、線を表現するデータ型 LineString の形式 [8] で表現する。

3.2.3 ストロークの定義

道なりかどうかは形状や人の感覚によって異なり、何をストロークの定義にするかによって違いが出る [9]。道路の形状は前述のようにアークで表現されるが、本稿では、

表 1 リンクテーブル

カラム名	データ型	説明
ID	int	リンクを一意的に識別するリンク ID
clazz	int	道路のクラス
source	int	リンクの始点ノードの ID
target	int	リンクの終点ノードの ID
x1	double	リンクの始点ノードの経度
y1	double	リンクの始点ノードの緯度
x2	double	リンクの終点ノードの経度
y2	double	リンクの終点ノードの緯度
km	double	ノード間の道路の長さ (単位は km)
arc	LineString	道路の形状を表す

表 2 道路クラスについて

clazz	kmh	説明
11	120	高速道路
12	30	高速道路の出入り口
13	90	国道
14	30	国道への連絡路
15	70	主要地方道
16	30	主要地方道への連絡路
21	60	一般地方道
31	40	2 車線以上の一般道
41	30	2 車線未満の一般道

簡単のため、文献 [6] と同様にエッジによりリンクを表現して道なりの判定に用いる。以下のルールに従って、連続したリンクを同一のストロークと判定する。

ルール 1 図 4 の (a) のように注目するリンク L の端点に接続するリンクがリンク L_i の 1 つのみの場合、リンク L_i とリンク L は同じストロークに属する。

ルール 2 図 4 の (b) のように注目するリンク L の端点に接続するリンクが複数の場合、注目するリンクと同じ道路のクラスで、成す角が 45 度以下であり、かつ最小となるリンク L_i をリンク L の系列の対象とする。

ルール 3 大きな交差点内において交差点内リンクが存在する場合、交差点内リンクを挟むリンクを全て接続するものとみなし、条件 2 を満たすリンクをリンク L の系列の対象とする。

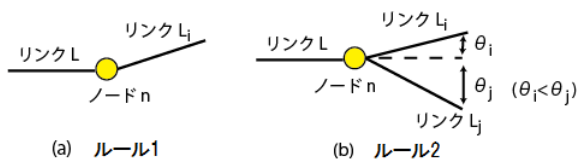


図 4 ストロークルール

3.2.4 ストロークデータ構造

ストロークのデータ構造は上記のストロークの定義を満

たすリンクの系列として表される。各リンクは必ず一つのストロークのみに属する。ストロークデータは、実道路データから構成される。ストロークの例を図 5 に示す。

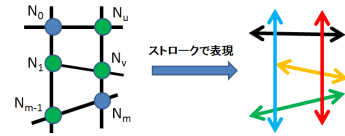


図 5 ストローク例

3.2.5 ストロークデータベース (SDB)

ストロークデータベースにはストロークテーブルが存在する。ストロークテーブルのデータ構造を表 3 に示す。

表 3 ストロークテーブル

カラム名	データ型	説明
stroke_ID	int	ストロークの ID
stroke_line	MultiLineString	ストロークの形状
stroke_length	double	ストロークの長さ

3.3 提案システムの構成

提案システムの構成を図 6 に示す。提案システムでは、ユーザが指定した地図と SN の範囲を描画し、SN の範囲内で SN を構築する。経路探索における探索範囲を SN の範囲内に属するノードとし、探索範囲内で最少ストローク最短経路探索法 (SPMS 法) により経路探索をする。SPMS 法はストロークネットワークデータベース (SNDB) より各リンクのストローク ID を取得し、ストローク ID を用いて経路探索をし、道なり優先経路を実現する。提案システムは、ユーザが出発点と到着点、そして SN を構築する範囲と基準長を指定すると、地図サーバから地図画像をストロークネットワークデータベース (SNDB) から範囲内の SN データを取得し、SPMS 手法により経路を探索して地図画面に描画する。さらに、経路長、ストローク数、計算時間、SN 構築時間を地図外に表示する。

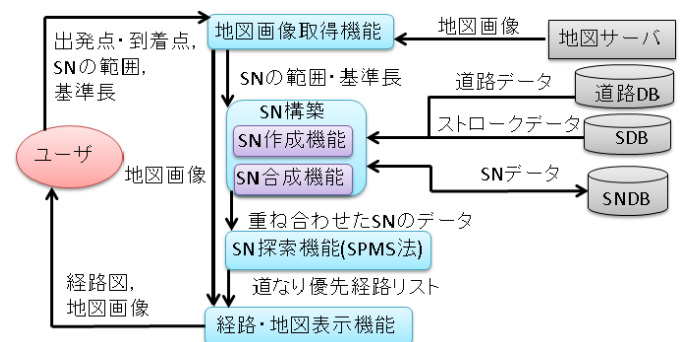


図 6 システム構成図

4. 提案システムの実現法

4.1 ストロークネットワーク (SN)

ストロークネットワークは、SDB から抽出した、ストロークの交点または端点をノード、ノード間のストロークの断片をリンクとする無向グラフである。出発点と到着点の2点を含む範囲内で長さ α 以上のストロークを抽出してストロークネットワークを1つ構築する。しかし、これだけでは出発点や到着点を通過するストロークが含まれるとは限らない。このため、出発点や到着点の周辺のストロークの密度を上げる。出発点や到着点の周辺で、出発点または到着点を通るストロークの長さ β ($\alpha \geq \beta$) 以上であるストロークを抽出してストロークネットワーク (SN2, SN3) を構築する。SN1, SN2, SN3 内の上乗点または到着点を重ね、1つのネットワークを構築する。

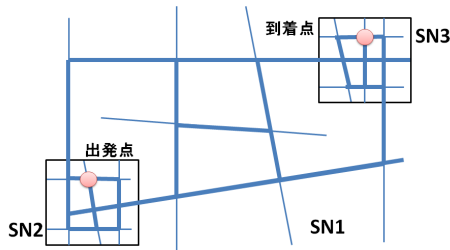


図7 重ね合わせたストロークネットワーク

4.1.1 ストロークネットワークデータ構造

ストロークネットワークのデータ構造を図8に示す。ストロークネットワークデータには、SN ノードと SN リンクの2つのデータがある。SN ノードは、ストロークの交点または端点である。SN リンクは、SN ノード間のストロークの断片である。

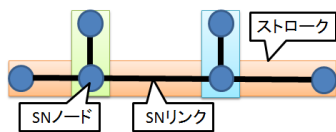


図8 SN ノードと SN リンク

4.1.2 ストロークネットワークデータベース (SNDB)

ストロークネットワークデータベースにはストロークネットワークテーブルと SN リンクテーブルが存在する。ストロークネットワークテーブルのデータ構造を表4に、SN リンクテーブルのデータ構造を表5に示す。

表4 ストロークネットワークテーブル

カラム名	データ型	説明
SN_ID	int	SN の ID
Base_length	double	基準長
SN_shape	MultilineString	SN の形状

表5 SN リンクテーブル

カラム名	データ型	説明
SN_linkID	int	SN リンクの ID
SN_ID	int	SN の ID
stroke_id	int	SN リンクのストローク ID
source_nodeID	int	SN リンクの始点ノード ID
source_x	double	始点ノードの経度
source_y	double	始点ノードの緯度
target_nodeID	int	SN リンクの終点ノード ID
target_x	double	終点ノードの経度
target_y	double	終点ノードの緯度
link_length	double	SN リンクの長さ
SN_shape	MultilineString	SN リンクの形状

4.1.3 ストロークネットワークの合成

複数の SN を構成するストロークの中には共通のストロークが含まれることがあり、そのストローク内の SN リンクが異なることがある。共通するストローク内の SN リンクの重なり方に着目すると、両端点が同一なもの（パターン1）、一端のみが同一なもの（パターン2）、また、ある SN リンクが他の SN リンクを覆いかぶさるもの（パターン3）があり、図9に示す。重なりを解消するために、重なりがある SN リンクを分解し、重なりをもたない SN リンクを生成する必要がある。本研究では密度の高いストロークネットワーク (SN2, SN3) を基にし、そのなかに密度の低いストロークネットワーク (SN1) を加え、重なる SN1 の SN リンクを重なりを持たない SN リンクへ変換させる。これにより、SN リンクが重なりが発生するのは SN2, SN3 の境界線上となる。

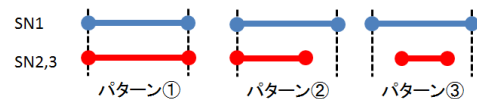


図9 SN リンクの重なり

差分 SN リンク生成手法 差分 SN リンク生成手法では、重なる SN1 の SN リンクから差分 SN リンクを生成し、重なる SN1 の SN リンクを差分 SN リンクと置き換えることで、重なりを持たない SN リンクへ変換することが目的である。

パターン1では、まったく同一の SN リンクが2つあることになるので、差分 SN リンクは生成せず、SN1 の SN リンクを削除する。

パターン2では、2つの SN リンクで共有していない端点をつないだリンクを生成し、ストローク ID をその SN リンクのストローク ID、長さを2つの SN リンクの長さの差とする。このとき、2つの SN リンクで共有していない端点をつないだリンクを差分 SN リンクとし、SN1 の SN リンクと置き換える。

パターン3では、範囲を SN1 の SN リンクの長さ

し基準長を SN2, SN3 の基準長とした SN を SNDN より取得し, SN1 の SN リンクの端点から SN2,3 の SN リンクの端点までに存在するリンクの系列を 1 つの差リンクにする. このとき, 長さはこのリンクの系列の長さの合計であり, ストローク ID はこのリンクの系列のストローク ID とする. そして, SN 2, SN3 の SN リンクの両側に存在する差リンク 2 つをそれぞれ差分 SN リンクとし, SN1 の SN リンクと置き換える.

アルゴリズム 密度の高いストロークネットワーク (SN2,SN3) と密度の低いストロークネットワーク (SN1) で重なりを持たない SN リンクと上記の手法より得た差分 SN リンクからなるストロークネットワーク (SN4) を構築する. SN4 の SN リンクを SN4.L, その SN リンクのフラグを $Flug_k$ として, 以下に疑似コードを示す.

Step1 SN2,3 を SN4 へ

for each SNlink L_i in SN2,SN3

SN4.L $\leftarrow L_i$

SN4.Flug $_k=0$

Step2 SN1 を SN4 へ

for each SNlink L_j in SN1

if SN1. l_j が SN2,SN3 の境界線上に存在する

SN4.L $\leftarrow L_j$

SN4.Flug $_k=1$

else then

SN4.L $\leftarrow L_i$

SN4.Flug $_i=0$

Step3 重なり補正

for each SNlink L_k in SN4

if SN4.Flug $_k>0$

link=差分 SN リンク生成手法

SN4.L $\leftarrow$ link

delete(SN4.L $_k$)

4.2 最少ストローク最短経路探索法 (SPMS 法)

道なり経路を実現するために, 本研究では SNDB を用いてストローク数が最少の中で最短距離である経路を探索させる. ストロークを考慮した経路探索をする上では, ある交差点を通過する場合, どのノードから通過するかにより経路長とストローク数に違いが出る. このため, ノードを接続するリンクごとに分解し, 分解した子ノードから道なり経路探索を行う.

4.2.1 構造体

ノード, 子ノード, リンクの関係を図 10 に示す. また, ノードと子ノードとリンクのそれぞれに構造体を持たせる. ノードとリンクには道路データベースに格納された一意に識別する ID を 1 つ持つ. また子ノードは親ノードのノード ID を用いて一意に識別した ID を持つ.

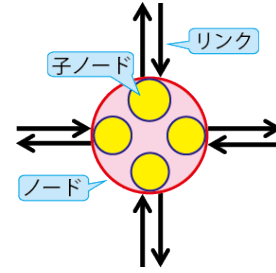


図 10 ノードと子ノード, リンクの関係図

ノード 各ノードには接続するリンクの数だけ子ノードがあり, その子ノードの子ノード ID を持つ. また, 各子ノードは親ノードであるノードが持つ子ノード ID より取り出すことができることとする. 各子ノードが持つ構造体を表 6 に示す. 初期状態では経路長とストローク数は無限大の値とし, 着目ノードの接続リンクのストローク ID と, 直前ノードの接続リンクのリンク ID とそのストローク ID は未登録とする. 着目ノードの接続リンクのストローク ID は SNDB より取得する.

表 6 子ノードの構造体

カラム名	データ型	説明
ChildNode_ID	int	着目ノードの子ノード ID
TotalLength	double	着目ノードの経路長
TotalStroke	int	着目ノードのストローク数
After_linkID	int	着目ノードの接続リンクのリンク ID
After_strokeID	int	着目ノードの接続リンクのストローク ID
Bofore_nodeID	int	直前ノードのノード ID
Before_linkID	int	直前ノードの接続リンクのリンク ID
Before_strokeID	int	直前ノードの接続リンクのストローク ID

リンク 各リンクが持つ構造体を表 7 に示す. ストローク ID は SNDB より取得する.

表 7 リンクの構造体

カラム名	データ型	説明
Link_ID	int	リンクのリンク ID
Length	double	リンクの距離数
Stroke_ID	int	リンクのストローク ID
startNode_ID	int	リンクの始点ノードのノード ID
nextNode_ID	int	リンクの終点ノードのノード ID

4.2.2 アルゴリズム

ストローク数計算手法 ストローク数計算では、直前ノードの子ノード n_u に接続するリンク l_i のストローク ID と、着目ノードの子ノード n_v と直前ノードの子ノード n_u に接続するリンク l_{uv} のストローク ID を比較し、比較した結果よりストローク数を計算する。比較した 2 つのリンクのストローク ID が同一であれば、着目ノードの子ノードがもつストローク数は直前ノードの子ノードが持つストローク数となる。また異なれば、直前ノードの子ノードがもつストローク数に 1 加えたものとなる。直前ノードの子ノード n_u とその接続するリンク l_i 、着目ノードの子ノード n_v とその接続するリンク l_{uv} の関係図を図 11 に示す。

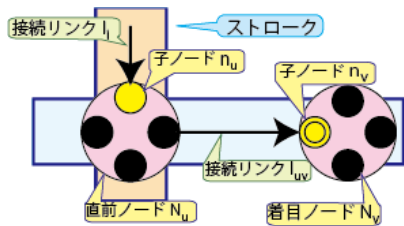


図 11 子ノードと子ノードの接続リンク

ストローク数最適解算手法 ストローク数最適解算手法では、あるノードの子ノードの中で以下のコスト比較条件を満たす子ノードを探索することが目的である。

- ストローク数が最少
- ストローク数が同数なら、経路長が最小

このときコストはストローク数と経路長の 2 つであり、コスト比較の際にはストローク数のみを用いて、ストローク数が同数となる場合のみ経路長を用いる。着目するリンクに隣接する 2 つのノードに着目し、着目ノードの子ノードの内の着目するリンクのリンク ID を持つ子ノードと直前ノードの複数の子ノードからストローク数計算手法に従って着目ノードの子ノードのストローク数と経路長を計算し、コスト比較を行う。コスト比較条件を満たせばそのストローク数と経路長を着目ノードの子ノードに上書き登録する。これを直前ノードがもつ子ノード全てに対して行い、着目ノードの子ノードに最適解となるストローク数とその経路長とその時の直前ノードや直前ノードの接続リンクのリンク ID・ストローク ID を登録させる。着目リンクを l_{uv} とし、直前ノードを N_u 、その子ノードを n_u 、着目ノードの子ノードを n_v として、以下にストローク数最適解算手法 (N_u, n_v) の擬似コードを示す。また、着目リンク l_{uv} と直前ノード N_u とその子ノード、着目ノードの子ノード n_v の関係図を図 12 に示す。

- ストローク数最適解算手法 (N_u, n_v)
for N_u . 子ノード数

ストローク数=ストローク数計算 (N_u, n_u, n_v)

```

if( $n_v$ .TotalStroke > ストローク数) or
  {( $n_v$ .TotalStroke=ストローク数) and
  ( $n_v$ .TotalLength> $N_u.n_u$ .TotalLength+ $l_{uv}$ .Length)}
then  $n_v$ .TotalStroke=ストローク数
 $n_v$ .TotalLength= $N_u.n_u$ .TotalLength+ $l_{uv}$ .Length
 $n_v$ .Before_nodeID= $N_u$ .Node_ID
 $n_v$ .Before_linkID= $N_u.n_u$ .After_linkID
 $n_v$ .Before_strokeID= $N_u.n_u$ .After_strokeID

```

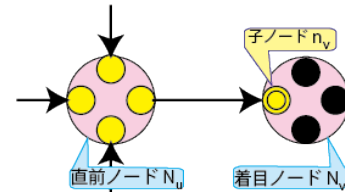


図 12 ストローク数最適解算手法

道なり経路探索のアルゴリズム SPMS 法のアルゴリズムは上記 2 つの手法を用いて出発点ノードの子ノードと隣接するノードの子ノードから順に、ストローク数最少の中で最短距離となる経路を確定させ、全てのノードに対して経路探索を行う。全てのノードからストローク数最少の中で最短距離である子ノードを持つノードに着目し、そのノードに隣接するノードの子ノードへ 2 つの手法よりストローク数最少の中で最短距離となるストローク数と経路長、直前ノードを登録する。これを全ての子ノードにストローク数最少の中で最短距離となるストローク数と経路長、直前ノードが登録されるまで繰り返し行う。探索経路は到着点ノードの子ノードの中で、ストローク数最少の中で最短距離が登録された子ノードから、その子ノードに登録された直前ノードと直前ノードの接続リンクのリンク ID より直前ノードの子ノードを辿ることで求めることができる。N はノードの集合、 N_u はノード、 n_u は N_u の子ノード、 n_v は N_u に隣接するノードの子ノード、 N_s は出発点ノード、TC は終了条件 (全てのノードの子ノードに対するストローク数最適解を算出すること) として以下の擬似コードに示す。

Step1 初期化

```

for each node  $N_u$  in N
  if  $N_u = N_s$ 
    then for each Childnode  $n_u$  in  $N_u$ 
       $N_u.n_u$ .TotalStroke=1
       $N_u.n_u$ .TotalLength=0
    else for each Childnode  $n_u$  in  $N_u$ 
       $N_u.n_u$ .TotalStroke=無限大
       $N_u.n_u$ .TotalLength=無限大

```

Step2 本計算

While(TC=False) Do

N_u =ストローク数最少の内, 最短距離数の子ノードを持つノード

for N_u に隣接するノードの子ノード n_v
ストローク数最適解算手法 (n_u, n_v)

5. プロトタイプシステム

提案システムを実現するために, 以下のリソースを用いてプロトタイプシステムを実装した.

- JAVA 言語 [10]
- PostgreSQL[11]

5.1 プロトタイプシステムの概要

構成は, 提案システムそのものをプロトタイプシステムで実現したため, 提案システムと同一である. 道路ネットワークと地図画像には地図サーバの地図データを用いる. 基準長と緯度経度座標を用いて SNDB から範囲内に存在するストロークネットワークの道路データを取得する. 取得したストロークネットワークの道路データを用いて指定したストロークネットワークの範囲内で SPMS 法より指定した 2 点間の経路探索を行う. SPMS 法における経路長, ストローク数, 計算時間および経路図を表示する.

5.2 プロトタイプシステムの動作

プロトタイプシステムの操作画面を図 13 に示す. 左側に地図画像を, 右側の上部に入力インターフェースを, 右側の下部に出力部を配置している. 地図画像内で SN 範囲にする長方形の左上の角・右下の角の順に左クリックをすると地図内に枠が表示され, 出発点・到着点の順に右クリックをすると地図内にマークが表示される. 出力部には出発点と到着点の世界測地系の緯度経度が表示される. 枠とマークが表示された状態でボタン「SPMS 法最短経路」を押すと, SPMS 法による経路探索を実行し, 地図画像内に探索経路を, 出力部に経路長とストローク数および計算時間が表示される. 探索範囲は枠の範囲内である.

6. 提案システムの評価

6.1 評価実験の目的

作成したプロトタイプシステムにおいて, SNDB の有無における SN 構築時間と経路探索時間を求める. SN 構築時間より, SNDB を用いることで SN の構築にかかる時間がどれだけ短縮されたかを検証する. 経路探索時間より, SNDB を用いて構築した SN と SNDB を用いずに構築した SN での経路探索に要する時間を比較評価する.

6.2 評価実験の方法

(1) 出発点・到着点の選定

名古屋市内で直線距離が以下となる任意の 2 点を選定

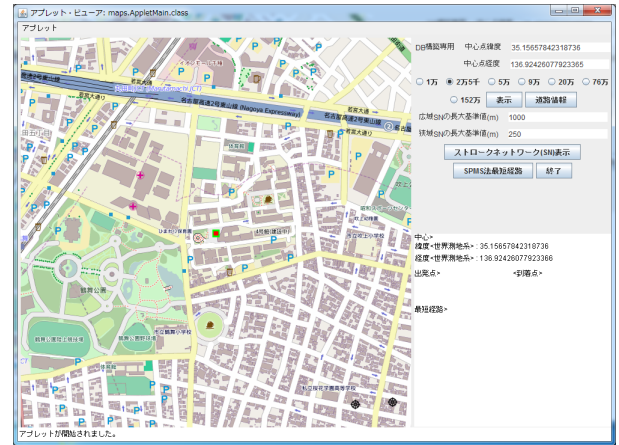


図 13 プロトタイプシステムの外観

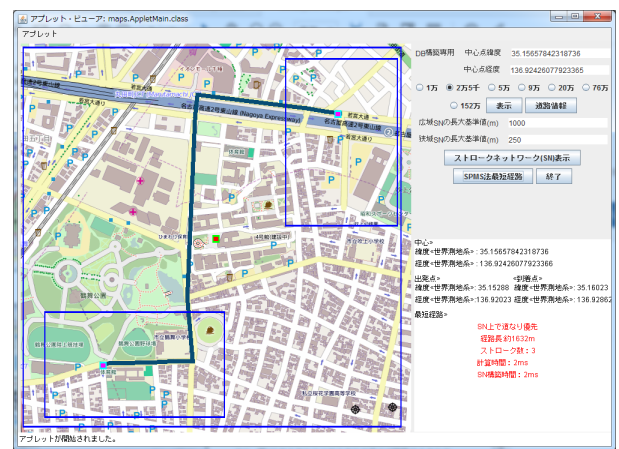


図 14 SPMS 法の経路探索

して出発点 S と到着点 G に選定する.

直線距離: 0.5km, 1.0km, 1.5km, 2.0km, 2.8km, 3.5km, 4.0km, 4.5km, 5.0km

(2) ストロークネットワーク

下記 SN1, SN2, SN3 を合成してストロークネットワーク SN を構築する.

SN1 低密度ストロークネットワーク

基準長: 1.0km

範囲: S と G を結ぶ直線を 1.15 倍した延長線の端点を含む最小正方形

SN2 高密度ストロークネットワーク

基準長: 250m

範囲: 出発点を中心とし, SN1 の範囲の $1/9$ の大きさの正方形.

SN3 高密度ストロークネットワーク

基準長: 250m

範囲: 到着点を中心とし, SN1 の範囲の $1/9$ の大きさの正方形.

(3) ストロークネットワーク SN の構築法

方法 1 提案手法 (SNDB あり): SNDB を用いて 4.1.3 節の手法により SN を構築する.

方法2 比較手法 (SNDB なし) : SDB からストロークを抽出し 4.1 節の手法により SN ノードと SN リンクを生成して SN を構築する。

(4) 計算時間の比較

方式1 と方法2 について、以下の2つの時間を測定して比較する。

(a) SN を構築する。

(b) SN 上で SPMS 法により出発点から到着点までの経路を探索する。

6.3 結果と考察

SN 構築時間を図15に、経路探索時間を図16に示す。SN 構築時間は、SNDB を用いることで密度の異なる SN をそれぞれストロークから構築する必要がなくなり、1秒以下となることがわかった。しかし、SN 構築の際に基準長を固定し範囲を正方形としたため、探索するのに重要となるノードが含まれた SN であるとは言えないという問題がある。また、経路探索時間はどちらも構築した SN 上で行っているため、SN ノード数に差がなく、経路探索時間がほとんど同じになるということがわかった。

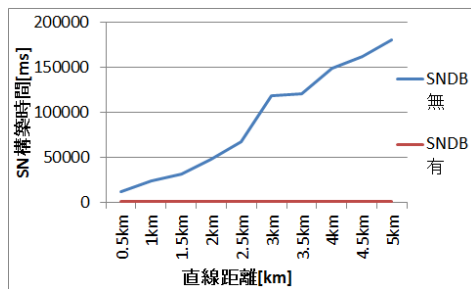


図 15 SN 構築時間

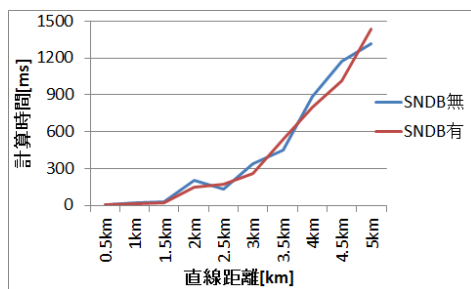


図 16 経路探索時間

7. おわりに

本論文では、SNDB から抽出した SN 上でストロークに基づいた経路探索手法で道なり優先の経路を求めるシステムを提案し、その実現法について述べた。また、提案システムを実装したプロトタイプシステムを用いて評価実験を行った。実験結果より SNDB を用いることで密度の異なる SN をそれぞれストロークから構築する必要がなくなり、

SN の構築時間がその分短くなり、約 1 秒以下の時間で SN を構築できることがわかった。SNDB の SN データには探索するのに重要となるノードが含まれた SN であるとは言えないという問題がある。

今後の課題として、優れた SN 構築手法を開発・評価することが挙げられる。提案システムでは SN の範囲内に存在する全てのノードを探索完了するまで経路探索を行うため、より効率的な経路探索ができるように SN を構築する範囲と基準長の検討、DB へのアクセス方法や経路探索法についても改良する手法を考える必要がある。また、ストロークをリンクから判定したため道路の形状から判定しているとは限らないという問題があり、アークから判定するルールについても検討する必要がある。

謝辞

本研究は JSPS 科研費 26330136, 25700009 および、科学技術振興機構 CREST の助成を受けたものです。

参考文献

- [1] 新帯, 山本, 高橋, “ストロークネットワークを用いた道なり優先経路探索法の実現”, “平成 27 年度電気・電子・情報関係学会東海支部連合大会, D5-5, 2015
- [2] 大西, 加藤, “交差点内コストを考慮した道路網における経路探索の手法とそのマルチメディア型経路案内システムへの応用”, 情報処理学会論文誌, 33(7), 970-979, 1992
- [3] 原, 塚原, 狩野, “多目的遺伝的アルゴリズムを用いたカーナビゲーションのための予測経路探索”, 情報処理学会高度交通システム研究会, ITS-24(5), pp.31-38, 2006
- [4] P.Sanders, D.Schultes, Engineering Fast Route Planning Algorithms, Experimental Algorithms (WEA), pp.23-36, 2007
- [5] 丹羽, 吉田, 福村, “道路網の階層的表現にもとづく経路探索アルゴリズムと地図情報システムへの応用”, 情報処理学会論文誌 31(5), 659-666, 1990
- [6] M.Murase, D.Yamamoto, N.Takahashi, On-demand Generalization of Guide Maps with Road Networks and Category-based Web Search Results, W2GIS 2015, pp.53-70, 2015
- [7] 新帯, 山本, 高橋, “道路ネットワークにおけるストロークに基づく経路探索法の実現と評価”, DICOM2015 シンポジウム論文集, 2015.
- [8] PostGIS 入門第 8 章
<http://workshops.boundlessgeo.com/postgis-intro-jp/geometries.html>
- [9] Robert C.Thomson, Dianne E.Richardson, The ‘Good Continuation’ Principle of Perceptual Organization applied to the Generalization of Road Networks, the 19th International Cartographic Conference, Ottawa, Canada, pp.1215-1223, 1999
- [10] Java
入手先 <<http://www.oracle.com/technetwork/java/index.html/>> (参照 2014-1-30)
- [11] PostgreSQL
入手先 <<http://www.postgresql.org/>> (参照 2014-1-30)