Vol.2019-HCI-182 No.23
2019/3/19

航空写真と電子地図からの幅員情報付き歩道ネットワークの自動作成に関する検討

前田篤彦[†1]

概要：車椅子利用者等の身体が不自由な人を電子的にナビゲートするには，「歩行空間ネットワークデータ」の整備が必要である．歩行空間ネットワークデータとは，歩道や横断歩道をノードとリンクで表現し，各々のリンクに幅員や傾斜等の様々なバリアに関する情報を属性として持たせたものである．しかし，このようなデータは今のところ十分整備されておらず，網羅的な整備には通常の工程を踏むと莫大な費用がかかる．そこで本研究では，既存の地理データを活用し，上記データの要となる，歩道と横断歩道のネットワークデータを効率的に自動作成する方法を提案する．また，車椅子利用者が通れるかの判断に有用な歩道の幅員も同時に算出する．提案手法は大まかに 3 つの工程からなる．1) 歩道情報の抽出に関しては，専門家が航空写真をトレースするなどして作成した電子地図に一般的に歩道の領域が描かれているため，このデータから歩道中心線と幅員を求める．2) 横断歩道に関しては，それらの位置と向きを航空写真から抽出する．3) 最後に，抽出した歩道及び横断歩道の情報を適切に接続して歩道ネットワークデータを作成する．提案手法を用いて， JR 山手線内全域の幅員情報付き歩行者道路ネットワークデータを作成し，評価した結果も報告する．

キーワード：歩道抽出，横断歩道検出，道路ネットワーク，バリアフリーマップ，歩行空間ネットワーク

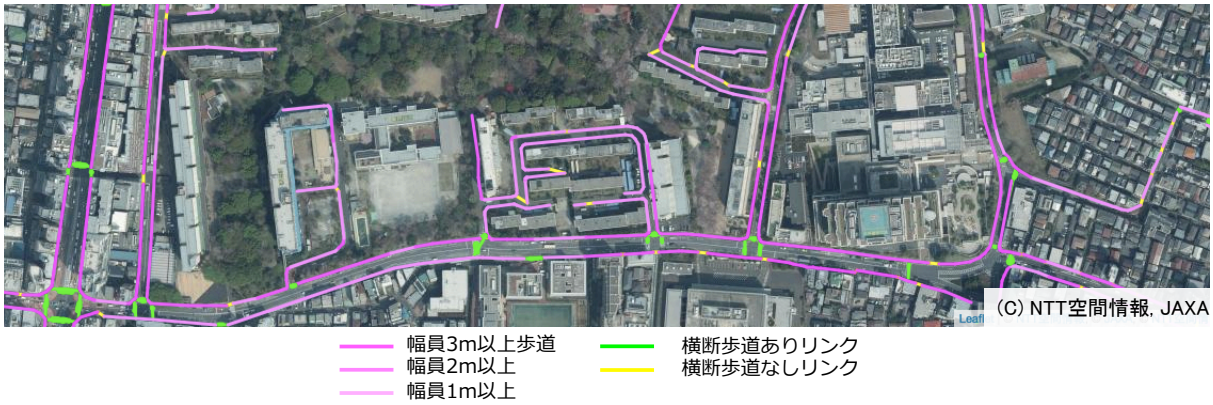

図 1　自動作成した幅員情報付き歩行者道路ネットワークの一例

1. はじめに

視覚障がい者や車椅子利用者等の身体が不自由な人を電子的にナビゲートするには,「歩行空間ネットワークデータ」[1]の整備が必要である．歩行空間ネットワークデータとは，歩道及び横断歩道の位置と接続関係をノードとリンクで表現し，各々のリンクに対して，幅員や傾斜等の様々なバリアに関する情報を属性として付与したデータセットを意味する．このようなデータが整備されていれば，例えば，車椅子利用者をナビゲートする際には，車椅子が通過できる幅があり，かつ車椅子では通過困難な段差や斜面が存在しない経路を抽出して示すことが可能になる．

しかし，今のところ十分な整備がされておらず，網羅的な整備には通常の工程を踏むと莫大な費用がかかる．また，この問題に対する取り組みとして，従来研究では，目視でも比較的判別しやすい点字ブロックの有無や路面の大きな凹凸等の情報をボランティアが発見して収集するものや，歩行者が持つスマートフォンのセンサー等を活用して歩道上の段差の情報を取得することに焦点を当てたものが多く，ナビゲーションの要となる歩行者道路ネットワークデータの効率的な作成方法に関するものは少なかった．また，歩道の幅員情報に関してもボランティアによる目視等では正確に判別しにくいという問題があった．

そこで本論文では，幅員情報を備えた歩行者道路ネットワークを効率的に自動作成する手法を提案する．歩行者道路とは歩道及び横断歩道のことである．提案手法は大まかに 3 つの工程からなり，歩道に関しては，専門家が航空写真を正確にトレースするなどして作成した電子地図データに一般的に歩道の領域も描かれているため，この地図から歩道中心線と幅員を求める手法を提案する．横断歩道に関しては，一般的に地図制作会社は地図に描いておらず情報も持っていないため，航空写真からそれらの位置と向きを自動抽出する方法を提案する．最後に，抽出した歩道と横断歩道の情報を適切に接続して歩行者道路ネットワークデータを作成するアルゴリズムを提案する．提案手法を実装

†1 NTT 未来ねっと研究所
NTT Corporation, NTT Network Innovation Laboratories

し， JR 山手線内ほぼ全域の幅員情報付き歩行者道路ネットワークを作成し（図 1），評価した結果も報告する．本手法を活用して作成した歩行者道路ネットワークには，幅員情報に限らず，従来研究で効率的な収集が可能になった様々なバリア関連情報を属性情報として付与することも可能であり，多様な条件での経路探索に活用できる．

2. 関連研究

有用な歩行空間ネットワークデータを作成するためには，歩道及び横断歩道の配置や接続状況，幅員，路面の段差・傾斜，階段・点字ブロック・手すり・エレベータ等の有無などを収集し，これらの接続関係を整理する必要があり，国土交通省によりガイドラインが作成されている[1]．これまで歩行空間に関するバリア関連情報を効率的に収集するために様々な手法が研究されてきた．

最も一般的な方法は，人が目視によりバリア情報を収集するというものである．効率化のために複数のボランティアが効率的に情報共有し，正しい情報を反映できる仕組み[2][3][4]や，人が現地に出向く労力をなくすためにリモートセンシング画像で判断する仕組み等が考えられている[5][6][7]．視認によりバリア情報を検出する手法の長所は，機械的で自動化された方法では発見困難な多様な種類のバリアの有無を検出しやすいことである．一方，短所としては，歩道の幅員や傾斜角等の定量化が必要なデータの収集が困難なことである．また，データ収集者としてボランティアを想定する場合には，参加へのインセンティブも考慮する必要がある．

一方，特定のバリア情報にフォーカスし，センサー等で自動収集する方法も提案されている．文献[8][9]は，車両移動時のセンサーデータから車道のバリア情報を検出する試みである．車椅子を使って，歩道のバリア情報を検出する方法も数多く提案されている[10][11][12][13][14][15]．さらに，車椅子だけでなく歩行時のセンサーデータから歩道上のバリア検出を試みた研究もある[16][17][18]．車椅子だけでなく歩行者のセンサーデータからでも検出できれば，より多くのボランティアがバリア検出に参加できる可能性が高まる．しかし，センサーでバリアを自動検出する手法は，現状，路面上の段差や傾斜の検出に限られている．

上記はいずれも，ナビゲーションで利用する歩行空間ネットワークデータのリンク情報に付与すべき属性情報の収集・整備に関するものであるが，ナビゲーションの要となる歩行者道路ネットワークデータを効率的に作成する方法の提案は少ない．現在のところ，地図制作会社でも大規模な歩行者道路ネットワークデータは整備できていない．車道ネットワークに関しては，通常，現地測量の結果や正斜投影により歪みが補正されたオルソ航空写真のトレース等から地図を作成すると共に，道路ネットワークデータも作成する方法が取られる．最近は，高解像度航空写真から自動作成する方法[19][20]や車の移動軌跡データから自動作成する方法も試みられている[21][22]．移動軌跡データから自動作成する方法は情報の更新頻度を向上させられる可能性が期待できるものの，対象となる全ての道路に関する複数回の移動軌跡データを人々のプライバシー等も考慮してどのようにして集めるかが問題となる．

3. 提案手法

3.1 全体構成

提案手法の特徴は，従来から安定的に更新され続けており，既に全国を網羅している既存の地理データを活用し，これらに対して多様な処理を積み重ねることで，実用的な効率と精度でデータ生成できるようにしていることである．図 2 に処理全体の流れと出力データを示す．提案手法では，既存の航空写真と電子地図データを入力データとする．航空写真は正斜投影により高層建築物で路面が隠れていることが少ないオルソ画像が望ましい．電子地図データは，専門家が航空写真を正確にトレースするなどして作成されており，一般的には歩道の領域が緯度経度座標によるポリゴンデータとして含まれているものの，横断歩道は描かれていない．いずれも地図会社から入手可能である．提案手法は大まかに 3 つの工程から構成される．1 つ目の工程では，歩道ポリゴンデータから歩道中心線と幅員を抽出する．2 つ目の工程では，航空写真から横断歩道の位置と向きを抽出する（地図も処理領域のマスクに使用）．最後の工程では，1 つ目及び 2 つ目の工程で抽出した歩道中心線と横断歩道の情報を適切に接続し，歩行者道路ネットワークデータを作成する．

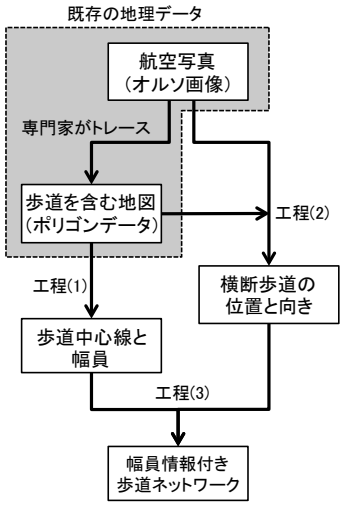

図 2　処理全体の流れと出力データ

3.2 工程（1）：歩道中心線と幅員の抽出

歩道ポリゴンデータから歩道中心線を抽出する工程でベースとなる手法は，指紋画像認識の前処理等で使われる細線化アルゴリズムである．ただし，通常の細線化処理を適用するだけでは，幅員が広めの歩道では，求めた歩道中心

線に余計な分岐が発生してしまう．また，典型的な歩道の形状を図 3 に示すが，歩道の端は，アール（カーブ半径）を持たせてある，いわゆる巻き込み部分となっているため，歩道の端に近づくにつれ，歩道中心線が反った形状になってしまい，ナビゲーションに利用するには好ましくない．そこで，本提案手法では，細線化によって得られる歩道中心線に対して余計な分岐を除去し，反りを修正する処理を導入する．図 4 は，歩道中心線と幅員の抽出工程のフローである．以下で処理の詳細を述べる．

図 3　歩道領域を含む地図（左）と航空写真（右）

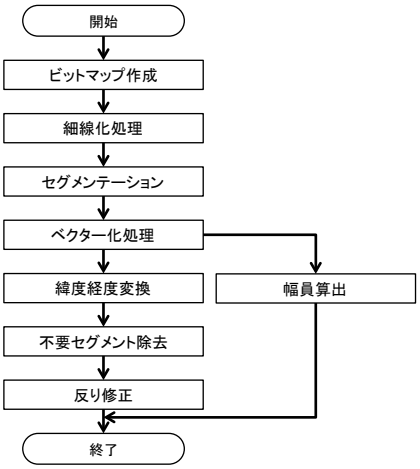

図 4　歩道中心線と幅員の抽出フロー

（ビットマップ作成）

電子地図に含まれている歩道ポリゴンデータは，緯度及び経度のリストとなっている．緯度経度を順に接続して形成されるポリゴンが歩道の領域を表す．最初に，このデータを 2 値のビットマップ画像に変換する．緯度経度は小数点で表されるため，歩道ポリゴンデータをビットマップに変換する前に，所望の精度に応じて，歩道ポリゴンデータの緯度経度に十分に大きな値を掛けて整数化する（例えば，緯度「35.692694242」，経度「139.702499333」に，10000000 を掛けて「356926942」「1397024993」という値にそれぞれ変換する）．そして，緯度の最大値と最小値との差分を縦幅とし，経度の最大値と最小値との差分を横幅とするビットマップを作成し，このビットマップの全体に歩道ポリゴンデータをモノクロで描画する．

（細線化処理）

次に，歩道ビットマップを細線化する処理を行ない，歩道の領域から，当該領域の幅方向の中心線を抽出する．細線化については様々なアルゴリズムが提案されているが，ここでは Zhang と Suen による手法[23]を適用する．具体的には，生成された歩道ビットマップをラスタスキャンし，注目画素を中心に 3×3 の画像値のパターンを観測し，注目画素が以下の条件(1)，(2)，(3)を満たすとき，この注目画素の値を白画素の値にする．

(1) 境界上にある黒画素であること．
(2) 白画素に変更しても連結性が保存されること．
(3) 端点でないこと．

図 5 の左は歩道ポリゴンデータの境界の一例を示し，右はそれに対して細線化処理の結果を重ねた例である．

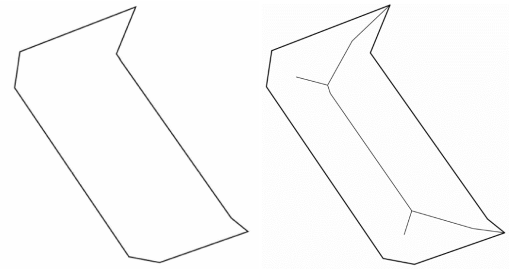

図 5　歩道ポリゴンデータ（左）と細線化の例（右）

（セグメンテーション）

次に，後処理のベクトル化のために，細線化画像のセグメンテーションを行なう．セグメンテーションとは，細線化画像における中心線を，当該中心線の端点，分岐点をもとに分割して複数の線を抽出する処理である．細線化画像をラスタスキャンし，注目するピクセルを P1 とし，この P1 の値が黒であるとき，P1 に隣接する 8 つのピクセルを図 6 の左のように P2～P9 にナンバリングする．これら P2～P9 のピクセルにおける値が黒であるピクセルの数 C_1 をカウントする．この結果が，C_1=1 であれば，ピクセル P1 を端点に設定する．また，C_1=1 以外であれば，ナンバリングしたピクセルの値を P2，P3，P4，P5，P6，P7，P8，P9，P2 の順に調べ，これら調べた計 9 つの値からなる配列を作成し，この配列における，白の次に黒になる箇所の数 C_2 をカウントし，C_2≥3 であれば，ピクセル P1 を分岐点とする．すべての端点及び分岐点を見つけた後は，これらの点間で線が存在する組を輪郭追跡により探し，2 点の組のリストを作成する．以後，これら 2 点の組でつながれた線をセグメントと呼ぶこととする．図 6 の右は細線化処理結果にセグメンテーションを施した結果である．E が端点，B が分岐点であり，seg_id がセグメントの ID である．

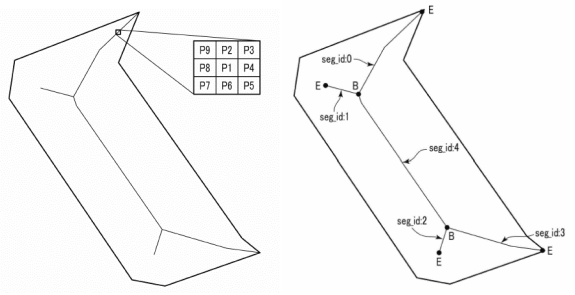

図 6　ピクセルへのナンバリング（左）とセグメンテーションの例（右）

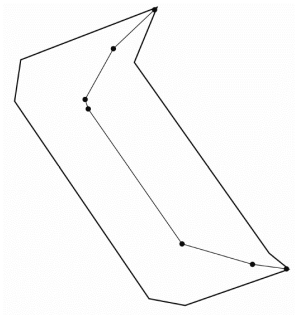

図 7　不要セグメントを削除した例

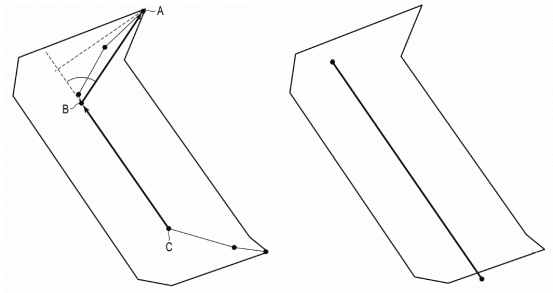

図 8　反りの判定（左）と反り修正結果（右）

（ベクター化・緯度経度への変換）

次に，ビットマップデータである個々のセグメントに対してベクター化処理を実施する．ここでは文献[24]の 194 ページに記載の 2 分割法を用いる．最後に，歩道ポリゴンをビットマップに変換したときの情報を用いて，各々の点の座標を緯度経度に戻す．

（不要セグメントの削除）

全長に対して幅員が広めの歩道では，求めた歩道中心線の端に余計な分岐が生じ，不要なセグメントが作成されてしまう．そこで，これらを除去する処理を行なう．ここでは，緯度経度の座標列として示される線のうち，同一座標の分岐点から延びる複数の線であって，分岐点を一端として他方が端点である線を抽出し，それらの中に所定の長さ未満の線があれば，それらの線を分岐がなくまるまで削除する．さらに不要セグメントを削除した結果，残ったセグメントが 1 つのパスとして表せる場合には，データ形式を 1 セグメントに変換する．図 7 は，不要セグメント削除処理の結果の例である．

（反り修正）

一般的に歩道の端は，アールを持たせてある巻き込み部分となっているため，歩道中心線のベクターデータは，歩道の端に近づくにつれ，図 7 のように曲がった形状になる．そこで，この曲がった形状を整形する反り修正処理を行なう．本手法の概要は，反りを有する線を，基準となる線分と，当該線分に直線的に連なる線分とに変換するというものである．図 8 の左は，反り修正処理の手順を示している．最初に歩道中心線の端点を見つけ，この端点を図 8（左）に示す A とし，これが歩道中心線の先頭のポイントだとすると，A から隣接するポイントまでの距離をヒュベニの公式等で算出する．この長さが反り修正除外基準の長さ L_{first} 以上であったならその端点側には反りがないものとして，残りの端点に対する処理に移行する．L_{first} 未満なら，ポイントを末尾方向に辿っていき，隣接するポイント間の長さが反り修正の基準となる線分の長さ L_{base} 以上となる線分を探し出す．この L_{base} 以上となる線分を図 8（左）に示す BC とすると，ベクトル $\boldsymbol{CB}$ とベクトル $\boldsymbol{BA}$ の $\cos\theta$ を求め，$\boldsymbol{CB}$ の B 側を $\cos\theta|\boldsymbol{BA}|$ の長さ分だけ延長し，点 B の直前までのポイントに対応するポイントをすべて削除する．また，端点 A が，歩道中心線の末尾のポイントである場合は，ポイントを逆方向に辿り，同様の処理を実施する．以上の処理の結果，反りが修正された歩道ベクターデータが生成される．図 8 の右は，反り修正結果の一例である．

反り修正において基準となる線分の長さ L_{base} は，歩道中心線全体の長さによって変わる可能性がある．具体的には，全体の長さが短ければ，L_{base} も短くなる可能性がある．そのため，本処理を実施する前に，歩道中心線の全長を計測し，その長さに応じて複数の基準を使い分けてもよい．

以上の処理をすべての歩道ポリゴンデータに対して行なう．図 9・10 では，不要セグメント除去及び反り修正処理の有無を比較している．図 9 はそれらの処理を複数の横断歩道に対して実施しなかった場合であり，図 10 は実施した場合となっている（幅員も抽出もしているが，この方法に関しては後述）．

図 9　地図に重畳した，不要セグメント除去及び反り修正なしの歩道中心線（青は幅員 1m 未満，ピンクは 1m 以上で濃いほど広い）

図 10　不要セグメント除去及び反り修正した歩道中心線

（幅員算出）

歩道の幅員を算出するために，ベクター化して求めた歩道中心線の各ポイントから，歩道ポリゴンの境界までの最小距離を求める．この方法としては，歩道中心線を形成する各ポイントにおける中心線の傾きを算出し，それに対する垂線を歩道ポリゴンの境界まで伸ばして幅員を求める方法が第一に考えられる．しかし，ベクター化処理の粒度等の影響によっては垂線の方向が必ずしも最小距離とはならないケースが発生するため，ここでは歩道中心線の各ポイントから最小距離となる方向を総当たりで確実に探索する方法をとる．

図 11 は，この方法を図式化したものである．具体的には，歩道中心線内のあるポイント P_s に対し，当該ポイントの横方向のビットマップの画素分の配列を作成し，この配列において，対応する画素に歩道が描かれている場合のみ，上記ポイントから各画素までの距離の二乗 H を計算し，この H を対応する要素（画素）の値として代入する．次に，この配列で数値 H が代入された各要素から，ビットマップの垂直方向（すなわち上下 2 方向）を走査し，この垂直方向における上記要素から歩道の境界（歩道と歩道外（背景）とを隔てる境界）までの間で最短距離となるピクセルを検出し，この最短距離の二乗 V を計算する．次に，各要素について計算された個々の H と V とを足した結果から最小値となるポイント P_w を見つけ，緯度経度座標に戻した後，P と P_w 間の距離をヒュベニの公式等により求め，これを 2 倍したものを P_s における幅員とする．

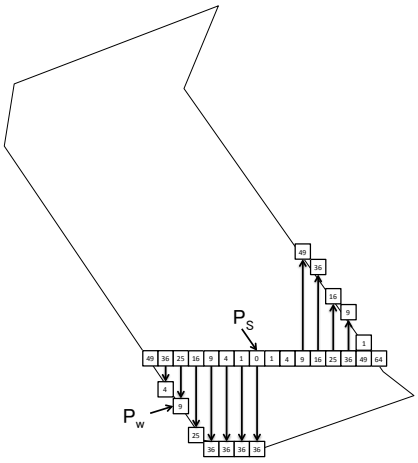

図 11　幅員算出過程

3.3 工程（2）:横断歩道の位置と向きの検出

電子地図には一般的に描かれていない横断歩道の情報を抽出するためにここでは航空写真を用いるが，そのためには横断歩道の縞模様が概ね 45cm 間隔であることを利用する．具体的には，図 12 に示すように，個々の横断歩道の領域より小さい一定サイズのウインドウを使って航空写真を走査する．このウインドウ内で，様々な角度がついた，実際の長さが 90cm に相当する線分を複数準備し，各線分上の一端を A，他端を C，その中間点を B とし，以下の評価値 D_1 と D_2 を求める．

$D_1 = |V_C - V_A|$

$D_2 = |(V_C - V_B) - (V_B - V_A)|$

ここで，A,B,C の明度をそれぞれ V_A, V_B, V_C とする．そして図 13 に示すように，これらの二つの値のそれぞれの平均 D_{1av} と D_{2av} を線分の角度別に求めると，横断歩道上では次の条件を満たすことになる．

（条件 1）D_{1av} の最小値の角度と，D_{2av} の最小値の角度とが同じである．

（条件 2）D_{2av} の最小値の角度と，D_{2av} の最大値の角度との差分が 90 度である．

（条件 3）D_{1av} の最小値の角度と D_{2av} の最大値の角度が同じである．

ウインドウを走査し，上記条件に当てはまる箇所の座標と D_{1av} の最小値の角度をすべて出力する．次に，これらの出力結果を車道幅程度のウインドウサイズで走査し，位置と角度が近いもの同士をクラスタリングする．クラスタリングしたあとは，個々のクラスタの中心座標も求めておく．

最後に横断歩道の方向（人が進行する方向）を求める．そのためには 2 種類の方法を使い分ける．具体的に，第一の手段として同一クラスタに分類された歩道検出二次元座標に対して主成分分析を行ない，第 2 主成分の分散と第 1 主成分の分散の比が一定以上であるならば，第 1 主成分の方向を横断歩道の方向とする．逆に一定未満であるなら，副次的な手段として D_{2av} の最大値すなわち横断歩道の白線に対する垂線を方向とする．図 14 は 2 種類の方法による横断歩道の方向の検出例である．D_{2av} だけから方向を求めない理由は，図 14 の右上の横断歩道のように，白線に対する垂線の向きが横断歩道の向きを必ずしも反映していないことがあるためである．

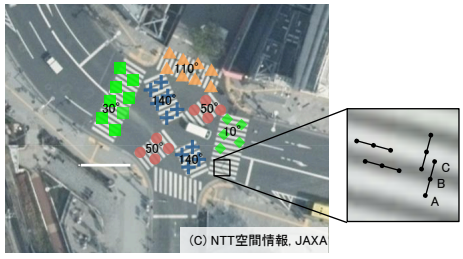

図 12　横断歩道の検出過程
（同一角度の縞模様が検出された箇所には同一記号を描画）

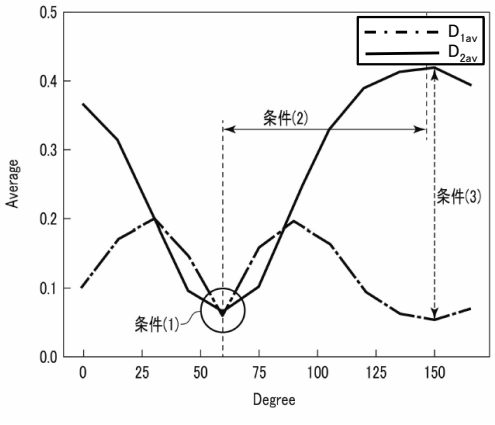

図 13　横断歩道上における評価値の分布

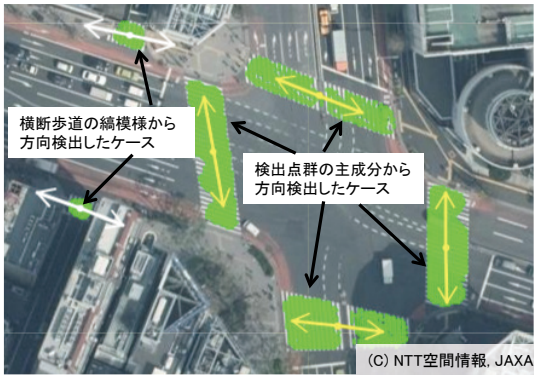

図 14　横断歩道の方向検出

なお航空写真を走査する際，電子地図データの車道ポリゴンデータに基づき，ディジタル航空写真における車道を示す領域のみを走査対象とする．これにより走査を高速化し，車道ではない領域での横断歩道の誤検出を防ぐことができる．

3.4 工程（3）：歩道中心線と横断歩道の接続

ここまでの工程で，歩道中心線と横断歩道のデータが準備できる．最後の工程では，複数の歩道中心線と横断歩道の情報を適切に接続し，歩行者道路ネットワークデータを作成する．ここでは接続の条件を以下とする．

- 実際に横断歩道によって連結されている歩道同士はデータ上でも無条件に接続する．
- 横断歩道によって連結されていなくても，ある歩道中心線の端点が他の歩道の端点もしくはコーナー点まで一定距離以内（障がい者が渡っても危険が少ない場所と距離）であれば接続する．

すなわち，横断歩道のデータを起点として連結箇所を探索するだけでなく，歩道を起点としても連結箇所を探索する必要がある．なお，横断歩道がない箇所での接続条件を端点もしくはコーナー点に限定している理由は，これらの場所が車道の分岐点に相当し，車の運転手が注意して走行する箇所と考えられるからである．図 15 は本工程のフローである．以下で処理の詳細を述べる．

（端点・コーナー点検出）

最初に，歩道中心線の端点 E 及びコーナー点 C を検出する．端点 E は，個々の歩道中心線を構成するセグメントの先頭と末尾のうち，他のセグメントと重複していない点である．一方，コーナー点 C に関しては，まず歩道中心線を形成する個々の線分の長さをヒュベニの公式等により求める．次に，この求めた長さが一定長以上となる二つの線分からみてなるべく中間の点をコーナー点Cとして検出する．

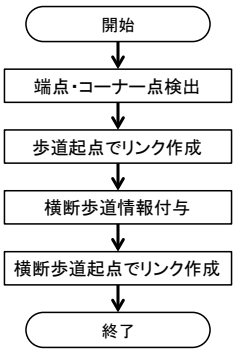

図 15　歩道中心線と横断歩道の連結フロー

（歩道起点でリンク生成）

次に，端点 E を起点として，リンクを生成する．個々の端点に対して，その他の端点又はコーナー点の中から，一定距離以内にある点を抽出し，それらの座標を結んだ線分をリンク情報として保存する．

（横断歩道情報付与）

次に，上記リンク情報に横断歩道情報を付与する処理を行なう．歩道起点リンク情報を形成する 2 点間の中心座標を求め，この中心座標から一定距離以内にある横断歩道の中心座標を探索し，この中心座標が存在すれば，リンク情報に「横断歩道あり」という属性情報（横断歩道情報）を付与し，中心座標が存在しなければ，「横断歩道なし」という属性情報を付与する．ただし，ここで一定距離以内にあるとされた横断歩道の情報を，後述する横断歩道起点でのリンク作成に重複して利用されないよう使用済みのフラグをつけておく．図 16 は端点・コーナー点を検出した直後の例で，横断歩道の検出座標も緑の点群で示している．図 17 は歩道起点でリンクを作成し，作成したリンクに必要に応じて横断歩道情報を付与した例であり，この段階でまだ利用されていない横断歩道の検出座標は残されている．

（横断歩道起点でリンク作成）

最後に，残された横断歩道の情報を起点としてリンク情報を生成する．この処理のために考えられる最も一般的な方法は，横断歩道の方向を表す線分の延長線上で歩道中心線との交点を求めることであろう．しかし，図 18 の例のように，歩道の領域に対して細線化処理を施した後では横断歩道が歩道中心線と交差しないことがある．そこで，実際

の横断歩道の向きになるべく近似させるかたちで横断歩道を起点とするリンク情報を作成する．具体的には以下のように処理する．

まず，歩道中心線上の各点の間に，一定間隔（例えば 1~2m 間隔）で新たな点の座標を挿入（補間）するポイント補間処理を行なう．具体的には，歩道中心線ベクターデータにおける全ての歩道中心線上の各点の間についてベクトルを生成し，ヒュベニの公式等により各ベクトルの距離を求めて単位ベクトルを生成し，この単位ベクトルを一定倍することで挿入点の座標を取得する．

このあと，個々の横断歩道から最短距離にある歩道中心線のポイントを探索し，P_{min1} とする．ここでの探索対象は，補間したポイントも含む．さらに，上記の歩道中心線を除外した上で再度同じ処理を行ない，検出したポイントを P_{min2} とする．図 19 の緑の点線で示すように，これら P_{min1} と P_{min2} を結んだ線分がリンク情報の元となるが，このままでは，実際の横断歩道と歩道の連結箇所から大きく離れるため，接続位置を補正する処理を行う．

そのためには，先に求めた横断歩道の方向を示すベクトルと横断歩道の中心 P_C から P_{min1} 及び P_{min2} それぞれへのベクトルとの角度を算出し，これらの角度がなるべく小さくなるよう歩道中心線上のポイントを選び直す（図 19 の補正後のリンク）．具体的には，上記の角度が最小となるポイントをまず取得し，その角度との差が一定以下で，且つ P_C との距離が最も短くなるポイントに選び直す．最後に，上記ポイント補間処理で付与したうち作成したリンクに使われなかったポイントの削除処理を行なう．

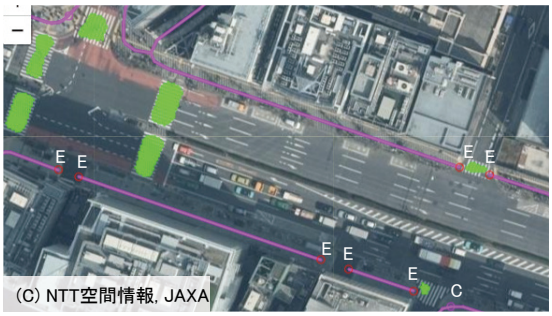

図 16 端点(E)・コーナー点(C)を検出した例（緑は横断歩道）

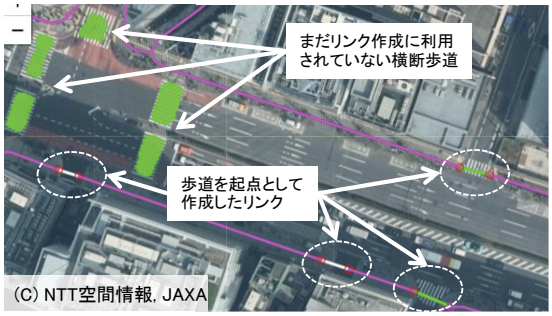

図 17 歩道起点リンク作成後，横断歩道属性を付与した例（白線は横断歩道なし，緑線はあり）

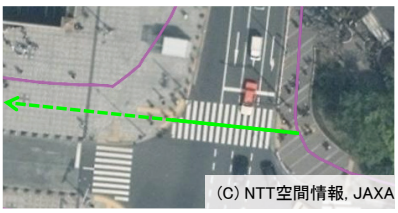

図 18 歩道中心線と横断歩道が交差しない例

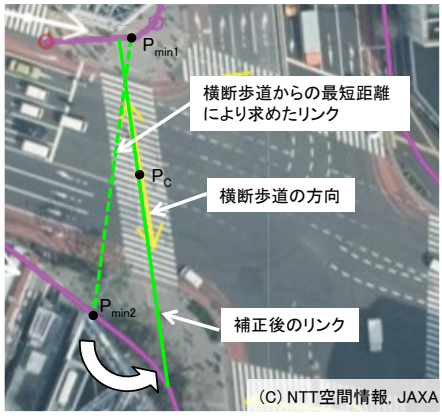

図 19 横断歩道を起点としたリンク作成の例

4. 評価

提案手法のうち，歩道中心線・幅員の抽出処理における細線化からベクター化までの工程と横断歩道の位置・向きの検出処理は Python3 系で実装し，複数の歩道ポリゴンデータや航空写真を並列処理できるようにした．また，歩道中心線・幅員の抽出処理における不要セグメント削除及び反り修正処理と，歩道中心線と横断歩道の接続処理は Javascript で実装し，ウェブブラウザ上の地図表示サービス上で処理結果をプレビューしながらのパラメータ調整や，Node.js 上での一括処理のどちらにでも対応できるようにした．

これらの実装を用いて，JR 山手線内のほぼ全域に相当する緯度 35.6197～35.738062，経度 139.701334～139.778837 の範囲における幅員情報付き歩道ネットワークデータを作成し，評価した．入力データとしては NTT 空間情報株式会社から販売されている電子地図と 10cm 解像度のオルソ航空写真を利用した．歩道中心線と幅員の抽出には，歩道ポリゴンデータをどの程度の大きさのビットマップに変換するかで処理時間が大幅に変わるが，ここでは 10cm 解像度相当（1 ピクセルが緯度 10cm に相当）で処理した．評価に使用した計算資源は CPU: Intel Core i7-7700HQ 2.8GHz，メモリ: 16GByte, OS:Windows10 の PC1 台のみである．

評価項目とそれらの結果を表 1 に示す．処理時間に関しては，すべての工程を足し合わせても，PC1 台で 1 日程度あれば JR 山手線内全域の処理が完了する．幸いにも歩道の形状を表すデータは複数のポリゴンに分かれており，航空写真も分割できるため，クラウド等の計算資源を用いて処理の並列化をさらに進めることも容易である．また，各

種精度もすべて 90%以上となっており，最終的には専門家による確認作業が必要になるものの，その作業は軽減されたものになると思われる．

表 1　評価結果一覧

評価項目	結果	備考
歩道中心線・幅員抽出の処理時間	6.5 時間	JR 山手線内には約 1 万の歩道ポリゴンデータが存在
横断歩道検出の処理時間	16 時間	
歩道中心線と横断歩道データの連結処理時間	10 分	
歩道中心線・幅員抽出の精度	97.5%	無作為に 200 箇所の歩道を抽出し，中心線が大まかに取れているかを目視で確認，幅員は航空写真上の 2 点をクリックすると，その間の幅が算出できるツールを作成して誤差 0.5m 未満を正解とした
横断歩道検出の再現率	93.7%	無作為にエリアを選び，その範囲に存在した 239 箇所の横断歩道で確認
横断歩道検出の適合率	100%	
歩道中心線と横断歩道データの連結精度	95%	リンクを作成すべき箇所を無作為に 100 箇所選び評価

5. 結論

本論文では，障がい者等を電子的にナビゲートするために不可欠な幅員情報付き歩道ネットワークデータを自動作成する方法を提案し，評価した．今後は構築した歩道ネットワークデータに，他の手法で収集された歩道の斜面や段差等の属性情報を組み込み，より有用な歩行空間データとして整備していく予定である．

参考文献

[1] 国土交通省, 歩行空間ネットワークデータ等整備仕様, 2018, http://www.mlit.go.jp/common/001244374.pdf
[2] Sozialhelden, E.V.: Wheelmap, http://wheelmanp.org
[3] Miura, T., Yabu, K., Ikematsu, S., Kano, A., Ueda, M., Suzuki, J.Sakajiri, M. and Ifukube, T.: Barrier-free Walk: A Social Sharing Platform of Barrier-free Information for Sensory/Physicaly-impared and Aged People, Proc SMC ’12, pp.2927—2932 (2012).
[4] 山本千尋, 船越要, 小長井俊介, 小西宏志, 川野辺彰久: 歩行者移動支援のためのバリアフリー関連情報収集手法の提案, 信会技報, Vol.116, No.23, LOIS2016-8, pp.39—44 (2016).
[5] Harak, K., Le, V., and Froehlich, J.E.: Combining Crowdsourcing and Google Street View to Indentify Street Level Accessbility Problems, Proc. CHI '13, pp.631--640 (2013).
[6] Rundle, A., Bader, M., Richards, C., Neckerman, K.and Teitler, J.: Using Google Street View to Aduit Neighborhood Environments, American Journal of Preventive Medicine, Vol.40, No.1, pp.94--100 (2011).
[7] Badland, H., Opit, S., Witten, K., Kearns, R. and Mavoa, S: Can Virtual Streetscape Audits Reliably Replace Physical Streetscape Audits?, Journal of Urban Health, Vol.87, No.6, pp.1007--1016 (2010).
[8] Eriksson, J., Giriod, L., Hul, B., Newton, R., Madden, S. and Balakrishnan, H.: The Pothole Patrol: Using a Mobile Sensor Network for Road Surface Monitoring, Proc. MobiSys ’08, pp.29—39 (2008).
[9] Mohan, P., Padmanabhan, V.N. and Ramjee, R.: Nericell: Rich Monitoring of Road and Traffic Conditions using Mobile Smartphones, Proc. SenSys ’08, pp.323—336 (2008).
[10] 牧恒雄, 竹内康, 松田誠: 歩道の凹凸評価方法に関する研究, 第 1 回舗装工学講演会論文集, pp.151—158 (1996).
[11] 石田眞二, 亀山修一, 岳本秀人, 姫野賢治, 鹿島茂: 車椅子の走行負荷に基づいた歩道の路面凹凸評価方法, 土木学会論文集 E, Vol.62, No.2, pp.295—305 (2006).
[12] 岡田美好: 車いすの乗り心地に着目した歩行者系舗装の性能指標に関する一考察, 土木学会舗装工学論文集, Vol.14, pp.189—194 (2009).
[13] 岩澤有祐, 矢入郁子: 多次元時系列データ解析によるアクセシビリティ可視化システムの開発, JSAI ’14 (2014).
[14] 隅田康明, 松永勝也, 合志和晃, 志堂寺和則: 車いす使用者向け経路探索のための路面の傾斜及び段差測定システムの開発, 信学技報, Vol.114, No.357, WIT2014-64, pp.63—68 (2014).
[15] Kuwahara, N., Nishimura, M., Shiomi, Y., Morimoto, K., Iwawaki, Y. and Nishida, N.: A Study on a Ubiquitous System for Colloecting Barrier-free Information of Evaluation Centers for Wheelchair Users, Proc. CASEMANS ’10, pp.36—39 (2010).
[16] 佐藤匠, 廣森聡仁, 山口弘純, 東野輝夫: スマートフォンと靴センサを活用した災害時通行路の状況推定, DICOMO’14, pp.258—265 (2014).
[17] 藤井海斗, 羽田野真由美, 西田京介, 戸田浩之, 澤田弘, 鹿島久嗣: 歩行者クラウドセンシングによる路面状態の推定, DEIM ’16 (2016).
[18] 宮田章祐, 荒木伊織, 王治順, 鈴木天詩, 健常歩行者センサデータを用いたバリア検出の基礎検討, 情報処理学会論文誌, Vol.59, No.1, pp.22—32 (2018).
[19] Sghaier, M.O. and Lepage, R.: Road Extraction From Very High Resolution Remote Sensing Optical Images Based on Texture Analysis and Beamlet Transform. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 2016, 9, 1946–1958.
[20] Yuan, J. and Cheriyadat, A.M.: Image driven GPS trace analysis for road map inference. In Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Orland, FL, USA, 5—8, Nov. 2013;pp.480—483.
[21] Li, H., Kulik, L., Ramamohanarao, K.: Automatic Generation and Validation of Road Maps from GPS Trajectory Data Sets. In Proceedings of the 25th ACM International Conference on Information and Knowledge Management, Indianapolis, IN, USA, 24—28 October 2016; pp. 1523—1532.
[22] Zhang, Y., Liu, J., Qian, X., Qiu, A. and Zhang, F.; An Automatic Road Network Construction Method Using Massive GPS Trajectory Data, ISPRS Int. J. Geo-Inf. 2017, 6,400.
[23] Zhang, T. Y. and Suen, Ching Y.: A Fast Parallel Algorithms For Thinning Digital Patterns, Communication of the ACM, Vol.27, No.3, pp.236-239 (1984).
[24] 画像情報教育振興協会, ディジタル画像処理[改訂新版], 2015.