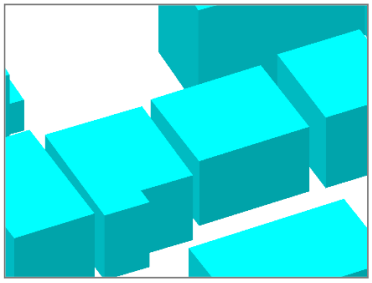
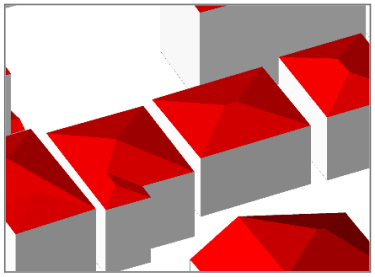
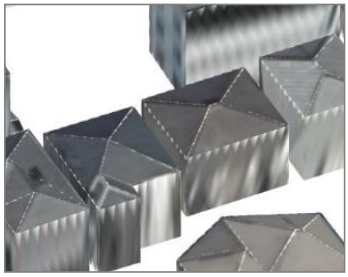
PLATEAU by MLIT

PLATEAU Technical Report

3D 都市モデル活用のための技術資料







AI 等を活用した LOD2 自動作成ツールの開発及び OSS 化技術検証レポート

series No. 61

Technical Report on Development and Open Sourcing of LOD2 Automated Generation Tool Utilizing AI and Other Technologies

目次

1. 実施概要

1.1. 本レポートの目的

国土交通省都市局では令和 2 年度から Project PLATEAU を開始し、スマートシティの社会実装をはじめとするまちづくりのデジタルトランスフォーメーションを推進するための基盤データとして、3D 都市モデルの整備・活用・オープンデータ化事業を進めている。

本レポートは、令和 4 年度に開発を行った LOD2 建築物モデル自動作成ツールの改修、新規に開発を行った LOD1-2 道路モデル自動作成ツールの開発成果に関する技術レポートである。3D 都市モデルの自動作成技術に資するツールの技術仕様や実行環境構築方法、精度検証結果等を公開することを目的としている。

1.2. 業務内容

本業務は令和 4 年度の建築物 LOD2 自動作成ツール開発成果「AI 等を活用した LOD2 自動生成ツールの開発及び OSS 化技術検証レポート」

（https://www.mlit.go.jp/plateau/file/libraries/doc/plateau_tech_doc_0056_ver01.pdf）等を踏まえ、更なる 3D 都市モデルの自動作成技術の開発を実施し、都市モデル自動作成ツールの改修・拡充を行った。

令和 5 年度の業務内容は以下のとおりである。

■ LOD2 建築物モデル自動作成ツールの改修

改修計画を立案するため、地域特性の異なる複数都市において LOD1 建築物モデル自動作成ツールを用い、公共測量成果となっている DSM 及び空中写真を入力データとした大規模なデータ生成実証を行い、精度、形状再現性、経済性を評価した上で課題を抽出した。抽出した課題を解決するための機能開発を検討し、開発計画を取りまとめた。開発成果は既存ツールと統合し、OSS として公開した。

■ LOD1-2 道路モデル自動作成ツールの開発

LOD1 及び LOD2 道路モデルの自動作成ツールを開発し、OSS として公開した。
自動作成ツールの開発内容を検討し、開発計画を取りまとめた。また、開発計画に従いシステム開発を行った後、有用性検証を行った。

1.3. 業務実施フロー

LOD2 建築物モデル自動作成ツールの改修の業務フローを図 1-1 に、LOD1-2 道路モデル自動作成ツールの業務フローを図 1-2 に示す。

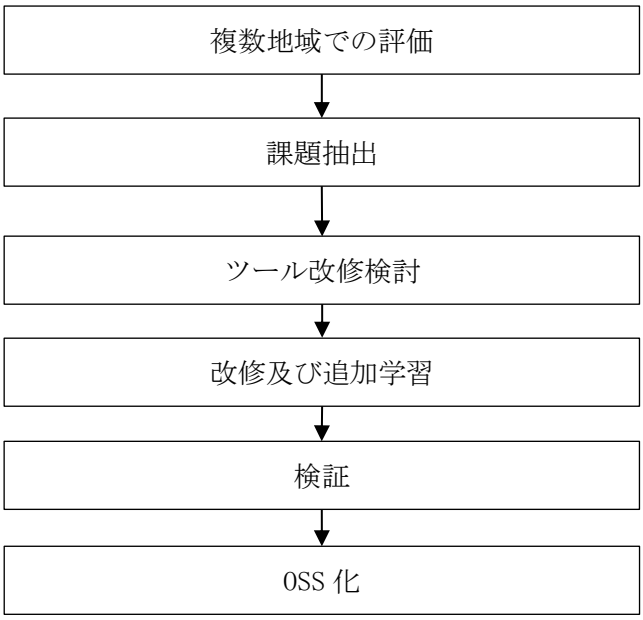


図 1-1　業務実施フロー　LOD2 建築物モデル自動作成ツール

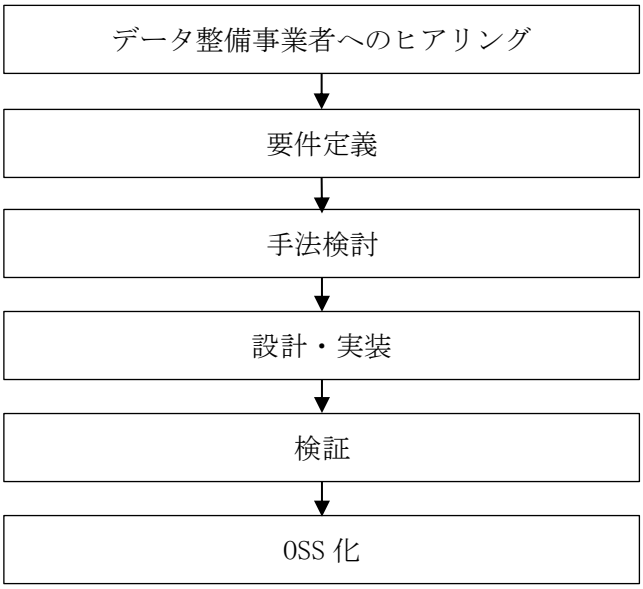


図 1-2　業務実施フロー　LOD1-2 道路モデル自動作成ツール

2. LOD2 建築物モデル自動作成ツールの改修

2.1. 改修の目的

令和 4 年度に作成した LOD2 建築物モデル自動作成ツール（以下、「ベース技術」）は 3D 都市モデルのデータ整備事業者や地方自治体に向けて、OSS として GitHub で公開している。本レポートでは、ベース技術の利用を促進するために、複数都市での検証から課題を抽出し、形状再現性及びユーザビリティの向上のための改修を行った。

本章は 2.2 にベース技術の不具合や技術的課題の調査結果、2.3 に改修内容、2.4 に改修後のツールでの LOD2 建築物モデル作成結果の検証について述べ、2.4.3 に本改修のまとめと課題を記載する。

2.2. LOD2 建築物モデル自動作成ツールの課題抽出

本節ではベース技術の技術的課題の抽出結果を記載する。2.2.1 にベース技術の概要、2.2.2 にシステム面での課題、2.2.3 ではベース技術で作成したモデルを評価して得られた課題、2.2.4 に 2.2.2 と 2.2.3 から改修すべき項目の検討結果を記載した。

ベース技術の仕様、システム開発結果、検証結果等は、2022 年度の技術検証レポート「AI 等を活用した LOD2 自動生成ツールの開発及び OSS 化技術検証レポート」

（https://www.mlit.go.jp/plateau/file/libraries/doc/plateau_tech_doc_0056_ver01.pdf）を参照。

2.2.1.ベース技術の概要

ベース技術は図 2-1 に示すシステム構成で、LOD1 建築物モデルの CityGML、航空写真、DSM 等から LOD2 建築物モデルの CityGML とテクスチャ画像、建物形状の OBJ を出力するシステムである。②モデル要素生成機能で AI を活用している。

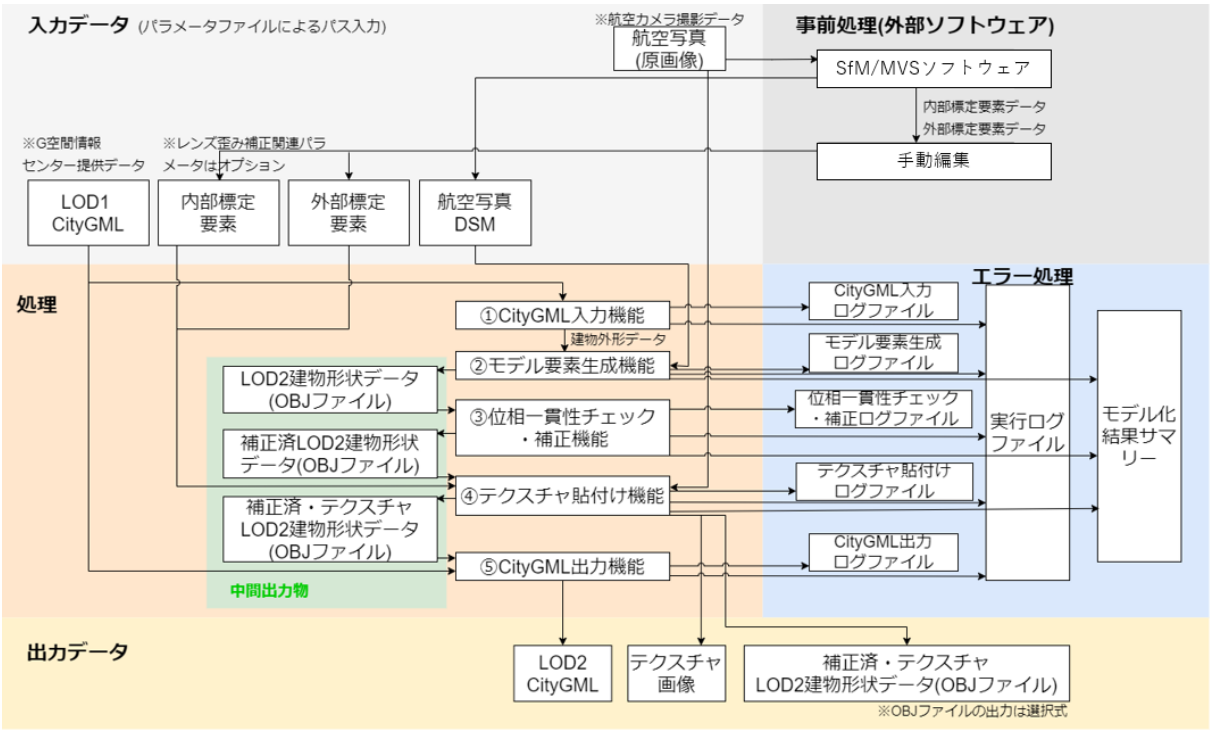


図 2-1　ベース技術のシステム構成

ベース技術の機能を表 2-1 に示す。今年度で改修した機能は No.2、No.4 である。

表 2-1　ベース技術の機能

No.	機能	機能詳細
1	CityGML 入力機能	CityGML 入力ファイルを読み込み、本システムに必要な建物属性データを取得する。
2	モデル要素生成機能	航空写真 DSM 点群と CityGML 入力モジュールが作成した建物外形データを使用して、LOD2 相当の建物 3D モデルの作成を行う。
3	位相一貫性チェック・補正機能	モデル要素生成モジュールにて出力された LOD2 建物形状データの各建物に対して、位相一貫性検査を行い、異常形状やデータの矛盾等を抽出して自動補正を行う。
4	テクスチャ貼付け機能	航空写真（原画像）から屋根、壁面のテクスチャ画像を抽出し、LOD2 建築物モデルに貼付ける処理を行う。 描画性能等を考慮したテクスチャ付与手法検討を行い、協議のうえ、必要な範囲の成果を組み込む
5	CityGML 出力機能	LOD1 CityGML データに、前述までの処理で作成した LOD2 建物形状データを追加し、LOD1 と LOD2 を含む CityGML データを作成する。

活用している技術について表 2-2 に示す。屋根形状の分類、屋根線の検出、バルコニー領域の抽出に AI 技術を用いている。

表 2-2　活用技術

No.	技術	内容
1	SfM/MVS	写真同士の対応関係を解析することで、三次元形状を復元する技術。航空写真（原画像）から 航空写真 DSM を生成するために利用。
2	ResNet34	画像分類手法。屋根形状を分類する AI 手法として利用。 Deep Residual Learning for Image Recognition https://arxiv.org/abs/1512.03385
3	HEAT	エッジ検出手法。屋根線を検出する AI 手法として利用。 HEAT: Holistic Edge Attention Transformer for Structured Reconstruction https://heat-structured-reconstruction.github.io
4	TransUNet	セマンティックセグメンテーション手法。バルコニー領域を抽出する AI 手法として利用。

2.2.2.ベース技術のシステム的課題

本節ではベース技術のシステムで発生する課題について記載する。

2.2.2.1. 建築物モデルテクスチャ縁部の不具合

R4 年度のユーザビリティ検証において、LOD2 建築物モデルのテクスチャの縁部に白いギザギザが発生していることが確認されていた（図 2-2）。

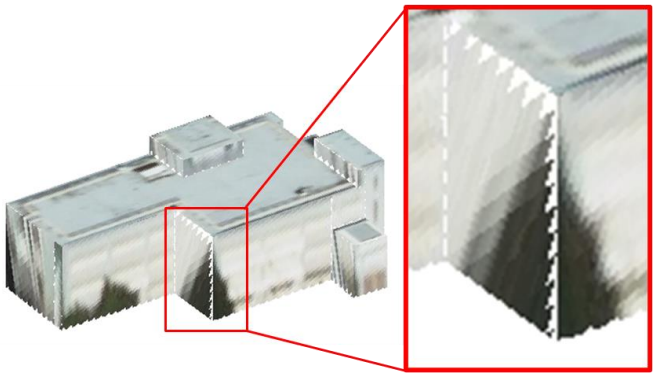


図 2-2　テクスチャ貼付けモジュールの不具合

上記の不具合が発生する原因は、テクスチャ切り出し時にマージンを設けずに建物範囲のみを切り出したことにより、テクスチャ貼付け時にテクスチャ画像の背景色が読み込まれて LOD2 建築物モデルに貼り付けられたことであった。

2.2.2.2. テクスチャ貼付けモジュールの処理時間とストレージ圧迫

ベース技術では LOD2 建築物モデルのテクスチャ貼付けを必ず行う設定となっていた。テクスチャ貼付けは建築物ごとに対応する位置の航空写真画像を取得し、建築物モデルの形状に合わせて画像を切り出したうえでテクスチャ画像保存を行うため、時間が掛かる処理であった。テクスチャが不要なユーザは不要な処理時間が掛かるうえに、テクスチャ画像ファイルが生成されるため PC ストレージを圧迫してしまうことになった。

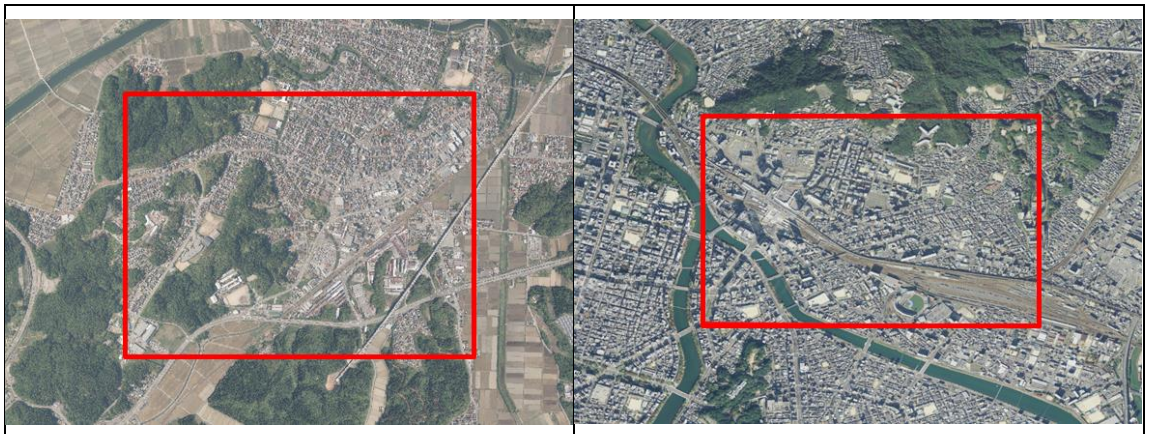
2.2.3.R4 年度検証対象外地域でのベース技術のモデル正確度検証と課題抽出

ベース技術は 2.2.1 に記載のとおり、モデル要素生成処理の複数の内部処理で AI 技術を活用して LOD2 建築物モデルを自動作成している。建物形状分類処理では画像分類 AI の ResNet34 を使用し、屋根線抽出処理では線を検出する AI の HEAT、バルコニー判定処理ではセグメンテーション AI の TransUNet を使用している。ベース技術ではこれらの 3 つの AI 技術を使用してモデルを作成するが、AI 技術において一般的に指摘される問題として汎化性能の問題がある。汎化性能の問題とは、ある地域で作成した学習データを用いて学習したモデルを他の地域に適用すると、認識精度が低下することである。ベース技術の汎化性能を確認するためには学習で用いた地域以外の地域で有効性を確認する必要性がある。

本検証ではベース技術で検証対象とした三鷹市と川崎市以外の地域で自動作成した LOD2 建築物モデルを評価し、ベース技術の課題を抽出した。

2.2.3.1. 評価対象建築物の選定

石川県加賀市（以下、「加賀市」）及び広島県広島市（以下、「広島市」）で建物サンプルをそれぞれ 609 棟、474 棟選定し、評価対象とした。各評価対象地域は図 2-3 に示す範囲とした。加賀市は大聖寺駅北側の住宅が多い地域を、広島市は広島駅周辺のビルが多い地域を選定した。



石川県加賀市	広島県広島市

図 2-3　評価対象地域

評価に用いた入力データの諸元を

表 2-3 に示す。ベース技術の入力データの仕様は地上解像度 25cm であるため、図 2-4 に示すフローでダウンサンプリングを実施した上で、ツールに入力した。

表 2-3　入力データの諸元

都市		建築物 LOD1 作成範囲	建築物 LOD2 作成範囲	原典データ情報
石川県	加賀市	146.19km^2	23.71km^2	撮影条件（OL/SL/GSD）：80%/60%/15cm
広島県	広島市	400.851km^2	3.115km^2	撮影条件（OL/SL/GSD）：80%/74%/14cm

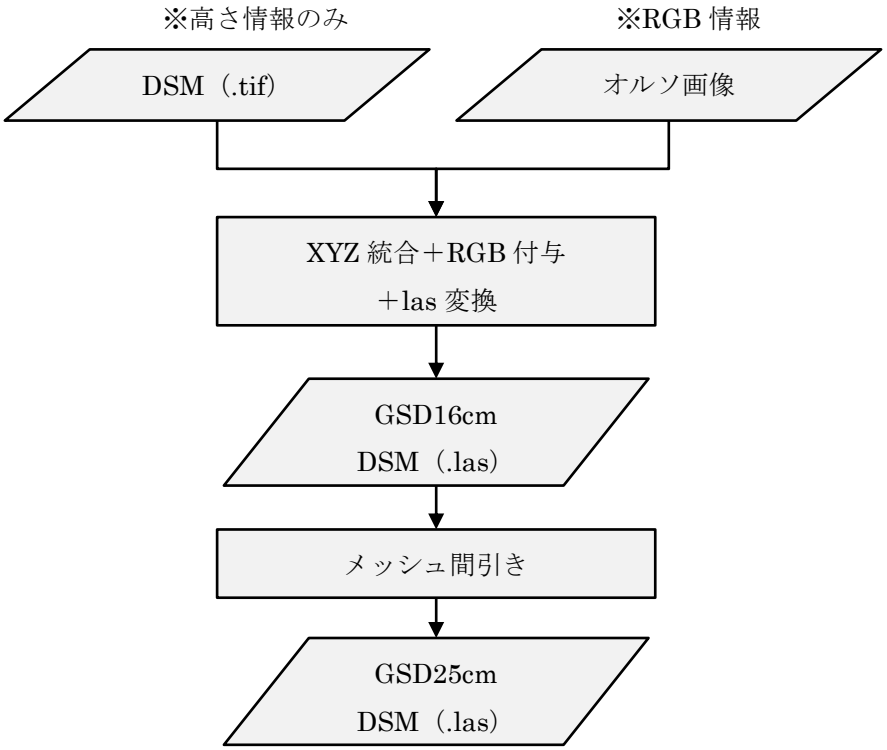


図 2-4　ダウンサンプリングのフロー

評価対象建物を図 2-5 の建築物の形状（以下、「建物形状」）ごとに分類し、各地域の特徴を把握した。各地域の建物形状ごとの棟数を表 2-4 に示す。加賀市は住宅が多い傾向があり、広島市はビルが多い傾向があるため、異なる建物形状に対するベース技術のモデル正確度を検証することが可能である。

表 2-4　評価対象地域の建物形状棟数

評価対象地域	建物形状			合計
	単純形状家屋	複雑形状家屋	ビル	
石川県加賀市	189	287	133	609
広島県広島市	142	68	264	474

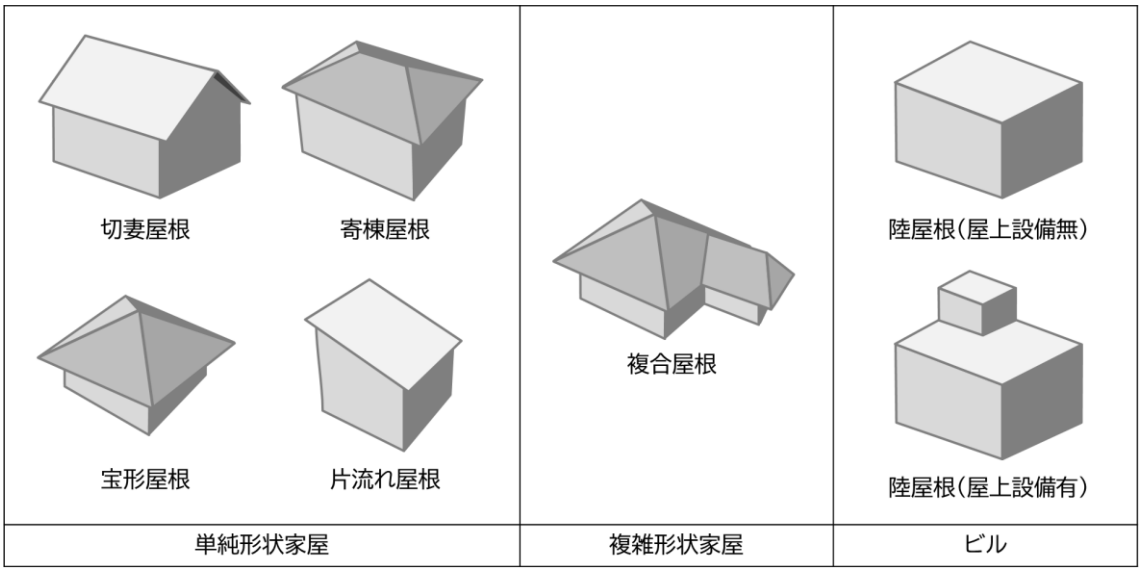


図 2-5　建物形状の分類

2.2.3.2. 評価方法

選定した建物サンプルについてベース技術を用いて LOD2 建築物モデルを生成し、表 2-5 に示した評価基準で目視による形状評価を行った。

表 2-5　評価基準

評価	評価基準
A	3D モデルが正しく、修正の必要がない
B	屋根形状は正しいが、1，2 箇所程度部分的に異なる
C	屋根形状が誤っている 又は 3 箇所以上部分的に異なる
D	元の屋根形状と大きく異なる、面が閉じていない、破綻している等

LOD2 建築物モデルの生成においては、「3D 都市モデル整備のための測量マニュアル」に定めた空中写真の撮影条件（オーバーラップ率：60%、サイトラップ率：30%、地上画素寸法：25cm）で計測したデータを用いてモデルを生成した。

2.2.3.3. 評価結果

加賀市の評価対象地域の評価結果を表 2-6 に示し、モデル生成結果例とその評価の例を図 2-7～図 2-9 に示す。

単純形状家屋と複雑形状家屋については評価 A が少なく、より低い評価の評価 B/C になるほど件数が増えたことが分かった。一方でビル形状は評価 A が最も多く、評価 B/C は割合として少ないことが分かった。評価 D については全体の 1 割未満であった。

以上から、加賀市においては単純および複雑形状家屋のモデル生成が課題であることが分かった。

表 2-6　加賀市_建物形状別の評価

建物形状	評価			
	A	B	C	D
単純形状家屋	11	77	95	6
複雑形状家屋	1	70	209	7
ビル	107	20	21	4
合計	119	167	325	17

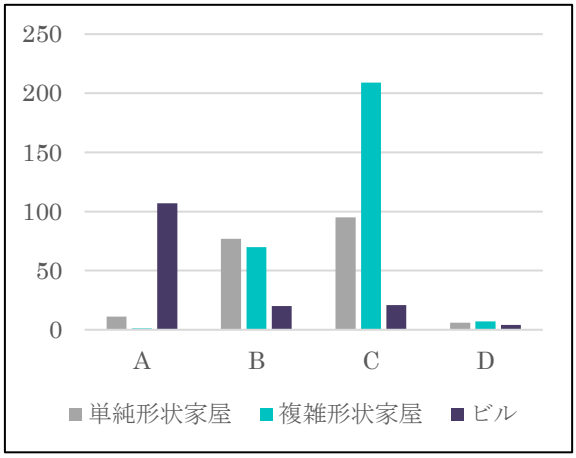
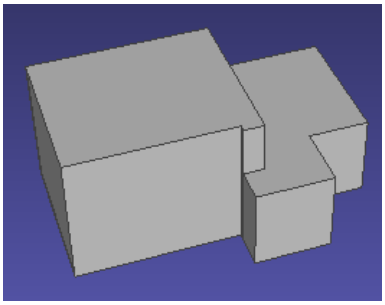


図 2-6　加賀市_建物形状別の評価

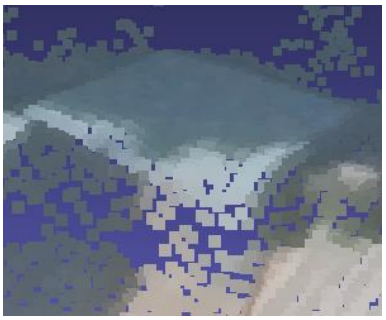
評価	A	説明	概ね一致している

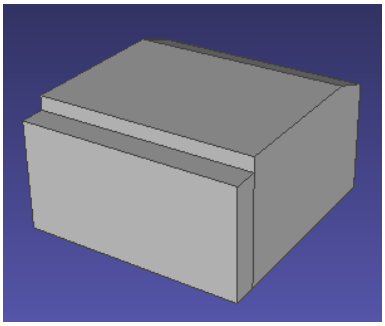


オルソ画像	DSM 点群	モデル生成結果

図 2-7　加賀市_評価結果例 A

評価	B	説明	高さ調整が必要

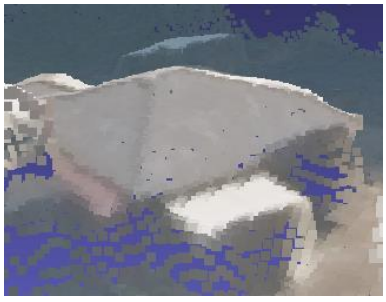


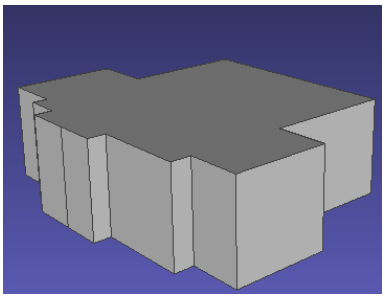


オルソ画像	DSM 点群	モデル生成結果

図 2-8　加賀市_評価結果例 B

評価	C	説明	高さ・形状不一致





オルソ画像	DSM 点群	モデル生成結果

図 2-9　加賀市_評価結果例 C

広島市の評価対象地域の評価結果を表 2-7 に示し、モデル生成結果例とその評価の例を図 2-12～図 2-14 に示す。

広島市においても加賀市同様に、単純形状家屋、複雑形状家屋については評価 A が少なく、より低い評価の評価 B/C になるほど件数が増えたことが分かった。ビル形状は評価 A と評価 C がほぼ同数である。一部のビル形状建築物で高低差のあるビル同士の間が狭く点群の精度が良くないことや、図 2-11 に示すような影の影響を受けていることが原因で評価 C が増えたと考えられる。評価 D については全体の 1 割未満であった。

以上から、広島市においても単純/複雑形状家屋のモデル生成と、建築物が密集する場合のモデル生成が課題であることが分かった。

表 2-7　広島市_建物形状別の評価

建物形状	評価			
	A	B	C	D
単純形状家屋	26	48	64	4
複雑形状家屋	3	21	44	0
ビル	126	22	113	3
合計	155	91	221	7

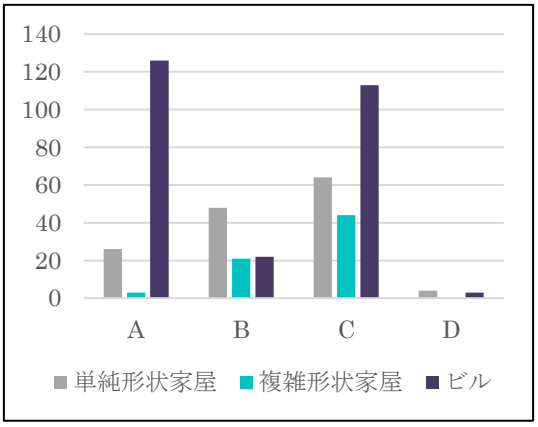


図 2-10　広島市_建物形状別の評価

図 2-11　広島市_影の影響

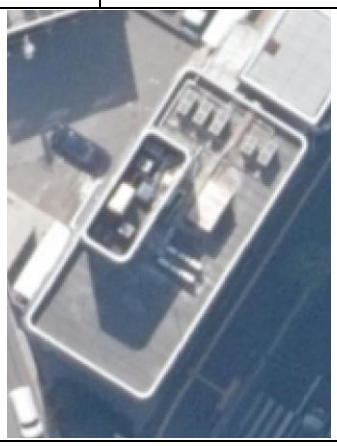
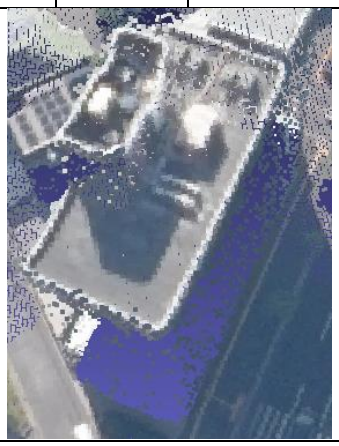
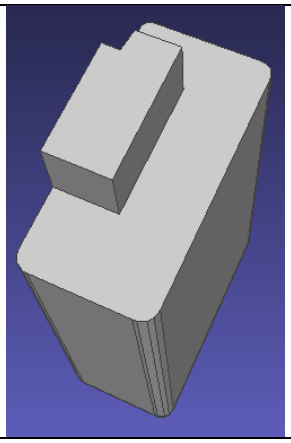
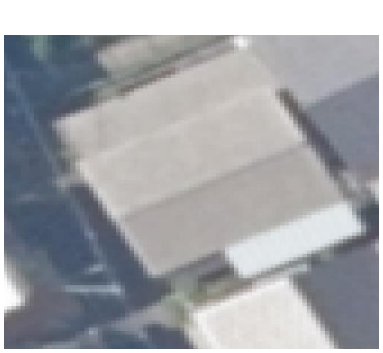
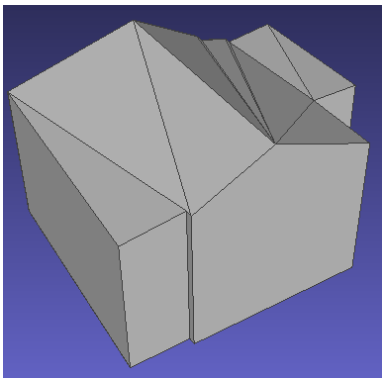
評価	A	説明	一致している
			
オルソ画像		DSM 点群	モデル生成結果

図 2-12　広島市_評価結果例 A

評価	B	説明	屋根要修正（一部）

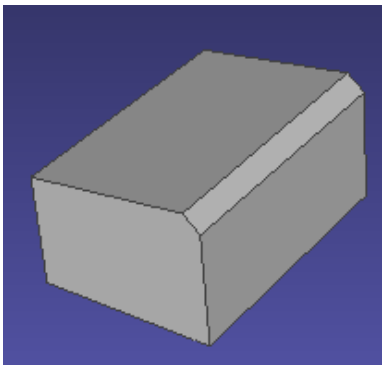




オルソ画像	DSM 点群	モデル生成結果

図 2-13　広島市_評価結果例 B

評価	C	説明	DSM 点群と不一致



オルソ画像	DSM 点群	モデル生成結果

図 2-14　広島市_評価結果例 C

2.2.4.課題抽出結果

評価結果より、ベース技術には次に示す課題があると考えられる。

・AI 適用処理の精度が低い

単純/複雑家屋形状のモデル正確性が低いことは、ベース技術の建物形状分類や屋根線抽出の AI 技術を適用した処理の精度が低いことが原因と考えられる。ベース技術では東京都三鷹市で AI モデルの学習を行った。

図 2-15 に各地域の検証に用いた範囲内の建物を示した。三鷹市と川崎市は区画が整理され、建物が密集し、建物形状が比較的シンプルである。加賀市と広島市は曲線的な形状の区画が形成され、建物の間隔に余裕があり、L 字型や階層的に複数の面を持つ複雑形状の建築物が比較的多く存在する。この建築物の形状の違いにより、データを入力したときに異なる特徴を持つ建築物の認識に AI モデルが対応できなかったと考えられる。

三鷹市

川崎市

加賀市

広島市

図 2-15 地域による建物形状の違い

また、撮影条件によって発生する影の影響（図 2-16）も考えられ、日陰で暗いことにより建築物の形状分類が正常に行われない場合や、日陰と日向の境界を屋根線として誤検出してしまうことも考えられる。

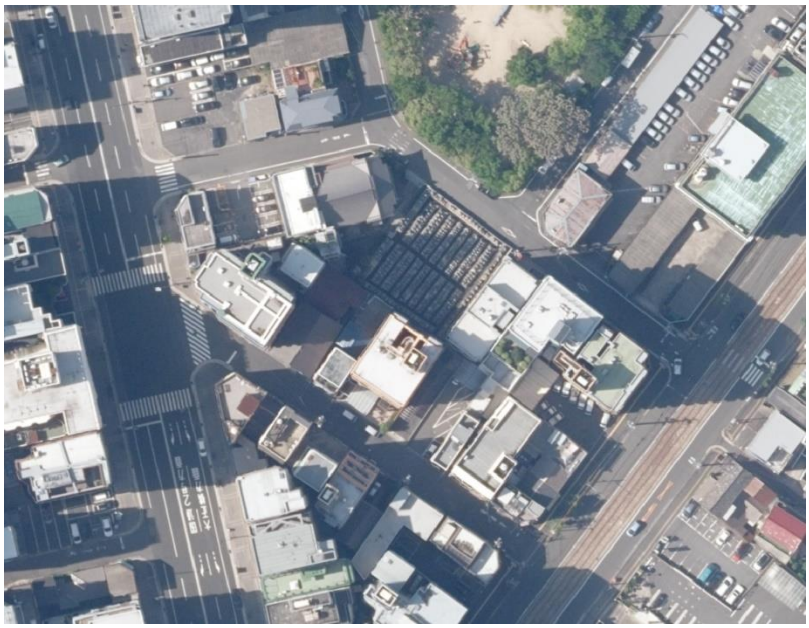


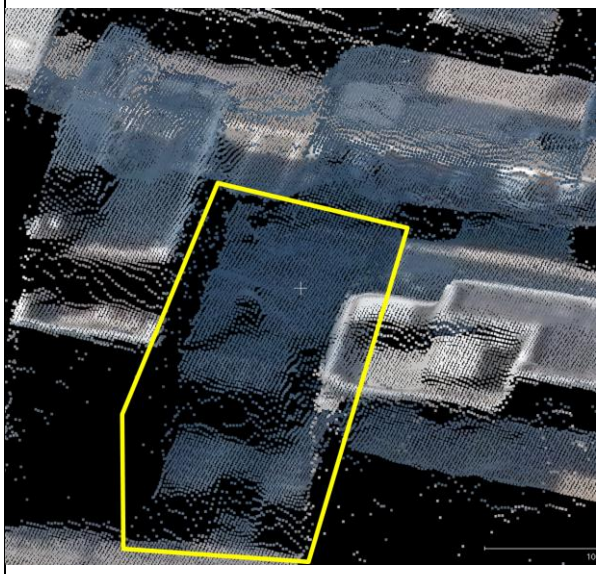
図 2-16 影が掛かる建築物（広島市）

・高低差のある建築物が隣り合う場合のモデル正確度が低い

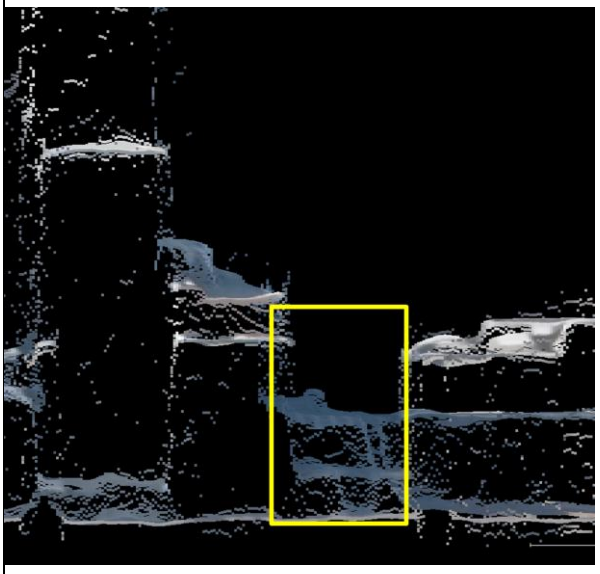
高低差のある建築物が隣り合うと低い方の建築物は DSM 点群の精度が低いことが多く、DSM 点群から正確な形状情報を取得することが難しくなる。図 2-17 の例では黄色で囲った建物が、GoogleMap の画像では屋根線が分かるのに対して、計測データのオルソ画像では影の影響で屋根線が把握しづらい。DSM 点群の断面図で確認すると屋根面の起伏が無く、正しいかわからない突起が部分的に表れていることが分かる。また、上記の課題の他に、ベース技術ではモデルの高さ付与時に建物の周辺を含む点群から屋根面の点と地面点を探索して、モデルの屋根と接地面の高さを抽出した。密集している場合には、地面の高さを抽出できないため、モデル生成が正常に行われづらい。

オルソ画像（計測データ）

オルソ画像（GoogleMap）



DSM 点群　鳥観図



DSM 点群　断面図

図 2-17 隣り合う建築物が高い時の影響

（広島市、DSM 点群 GSD14cm）

上記の課題のうち、高低差のある建築物が隣り合う場合については高解像度の点群を使うことが有効であると考えられ、また機械学習技術の進歩によってもこの課題の解決が期待できるが、本ツールの改修においては、2.2.2 に記載したシステム的課題の改修と、AI モデル自体の精度向上や他地域での汎用性向上に焦点を当てて取り組むこととした。

次章で改修内容の詳細について記載する。

2.3. LOD2 建築物モデル自動作成ツールの改修

本章では LOD2 建築物モデル自動作成ツールの改修内容を記載する。

2.3.1.テクスチャ貼付けモジュールの改修

従来のマスク画像によるテクスチャ切り出しに枠線 4pixel 分のマージンを設けて画像を切り出すことで修正を行った（図 2-18）。修正後の LOD2 建築物モデル（図 2-19）では縁部に白いギザギザが発生しないことが確認できた。

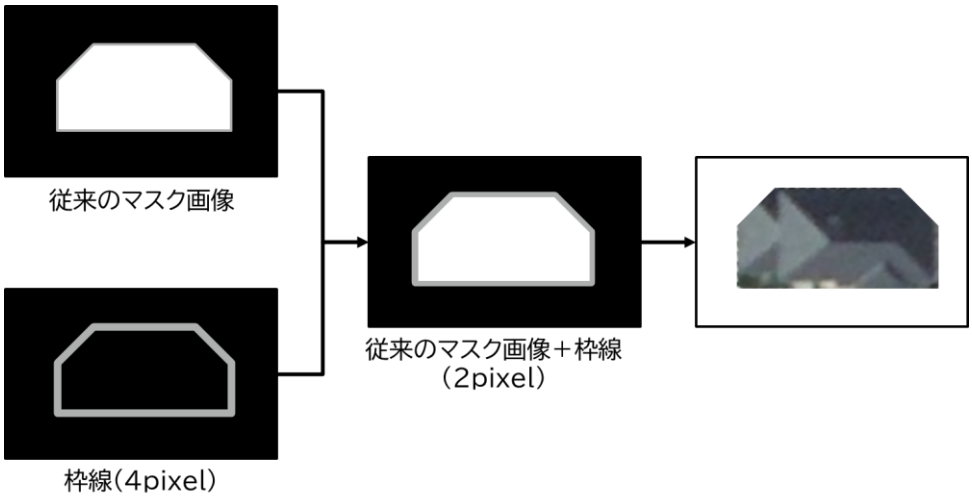


図 2-18　テクスチャ貼付けモジュールの改修

図 2-19　テクスチャ貼り付けモジュールの改修結果

2.3.2.テクスチャ貼付け ON/OFF オプションの追加

ベース技術では、LOD2 建築物モデルのテクスチャ貼付けを必ず行う設定となっていた。

テクスチャ貼付けが不要なユーザには、不必要な処理時間が掛かってしまうことや、テクスチャ画像ファイルが生成されてしまうことになっていたため、本改修においてツール実行時のパラ

メータファイルからテクスチャ貼付け機能の ON/OFF をオプションで設定可能とした。図 2-20 の赤枠内の”OutputTexture”を true で ON、false で OFF となる。

図 2-20　テクスチャ貼付け機能の ON/OFF 設定

2.3.3.AI モデルのチューニング

本章では建物自動分類処理で使用されている画像分類 AI モデルのチューニングについて記載する。チューニングでは AI モデル、転移学習、高さ情報付与の 3 段階に分けて検証を行い、建物自動分類処理に適している AI モデルを作成した。

検証結果から本ツール開発では、RGB 色情報を入力とする ResNet50 の転移学習モデルを建物形状分類 AI モデルに使用することとした。

2.3.3.1. AI モデルの検証

ベース技術では画像分類 AI モデルに ResNet34 を使用した。ResNet34 は VRAM 使用量の少なさ、分類精度、計算速度が優れているため採用した。本検討では異なる AI モデルで検証を行い、より建物自動分類処理に適している AI モデルを選定した。

検証では ResNet34、ResNet50、EfficientNet-b2、EfficientNet-b3、ViT の 5 つの AI モデルを使用した。各 AI モデルの特徴について次に説明する。

・ResNet34 および ResNet50

ResNet とは 2015 年に発表された画像分類 AI モデルである。比較的古い手法であるものの、その精度が高いことから現在でも広く利用されている。ResNet の特徴は図 2-21 の Residual Block（以下、「ResBlock」と呼ぶ。）という構造を持つことであり、図 2-22 のように ResBlock を複数層重ねること（一般的に”深くする”という。）によって画像からより詳細な特徴量を抽出することが可能である。AI モデルの検証で用いる ResNet の末尾の数字は ResBlock を何層重ねたかを示すもので、数字が大きいものの方はモデルが大きく、より深い特徴量が取得でき高い性能の画像分類が可能であると言える。ただし、層が深いほど AI モデルの計算量と消費メモリ容量が増えてしまうため、処理時間が長くなってしまう欠点がある。

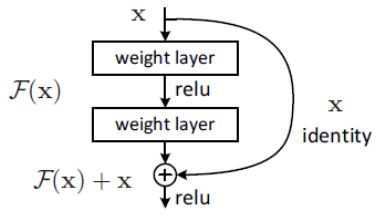


図 2-21 Residual block の構造[1]

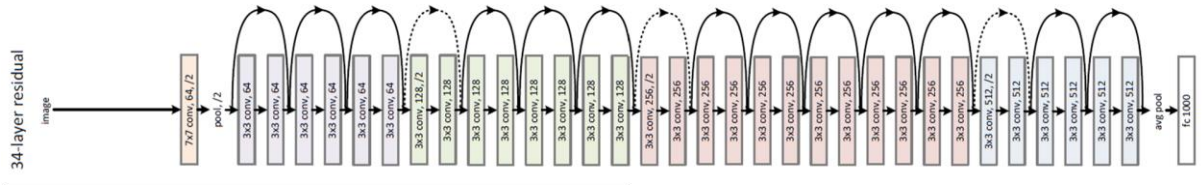


図 2-22 ResNet34 のネットワークアーキテクチャ

[1] 出典 : HE, Kaiming, et al. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. p. 770-778.

・EfficientNet-b2 および EfficientNet-b3

EfficientNet は 2019 年に発表された画像分類 AI モデルである。畳み込みニューラルネットワークの深さと広さ、入力画像の解像度のバランスを調整することにより高い精度で分類可能にした AI モデルである。EfficientNet は図 2-23 の MBConv という計算ブロックを使用しながら、効率的な特徴量抽出を行う。AI モデルの検証で用いる EfficientNet の末尾の b2, b3 は AI モデルの大きさを示すもので、数字が大きいものの方はモデルが大きい。モデルが大きいものほど性能が高い傾向があるが、計算量とメモリ消費量が増えてしまうため、処理時間が長くなってしまう欠点がある。

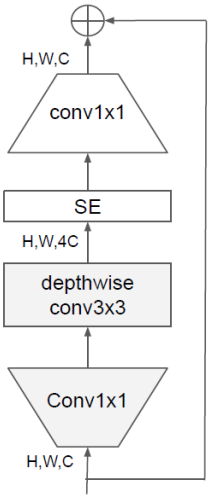


図 2-23 MBConv の構造[2]

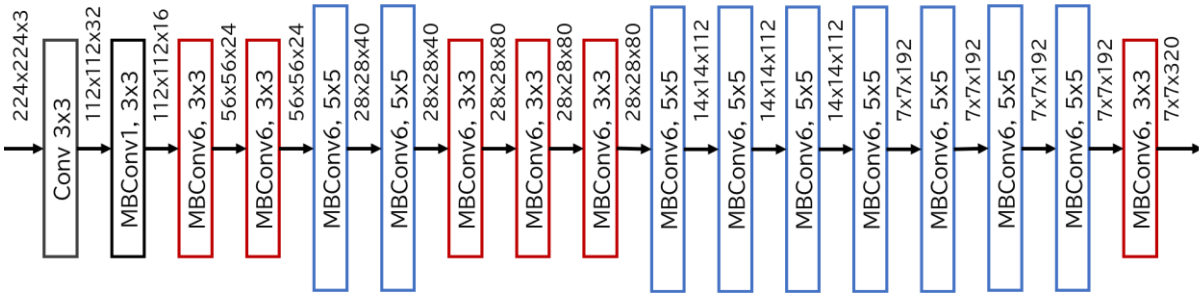


図 2-24 EfficientNet-b3 のネットワークアーキテクチャ

[2] 出典 : TAN, Mingxing, LE, Quoc. Efficientnetv2: Smaller models and faster training. In: International conference on machine learning. PMLR, 2021. p. 10096-10106.

・ViT

ViT(Vision Transformer)は 2020 年に発表された画像分類 AI モデルであり、2023 年現在、画像認識分野で最も利用される AI モデルと言える。自然言語処理分野で高い性能を達成した Transformer 構造を画像認識に応用したものであり、従来の畳み込み処理ベースの AI モデルよりも高い性能で画像分類可能とされている。図 2-25 に示すように ViT では画像を複数枚の小さなパッチに分割し、パッチ画像同士がどのような関連性を持つか計算する。関連性をもとに全てのパッチ画像で他のどのパッチ画像を注意すべきかを学習することによって、重要な部分に焦点を当てて画像を認識することが可能である。

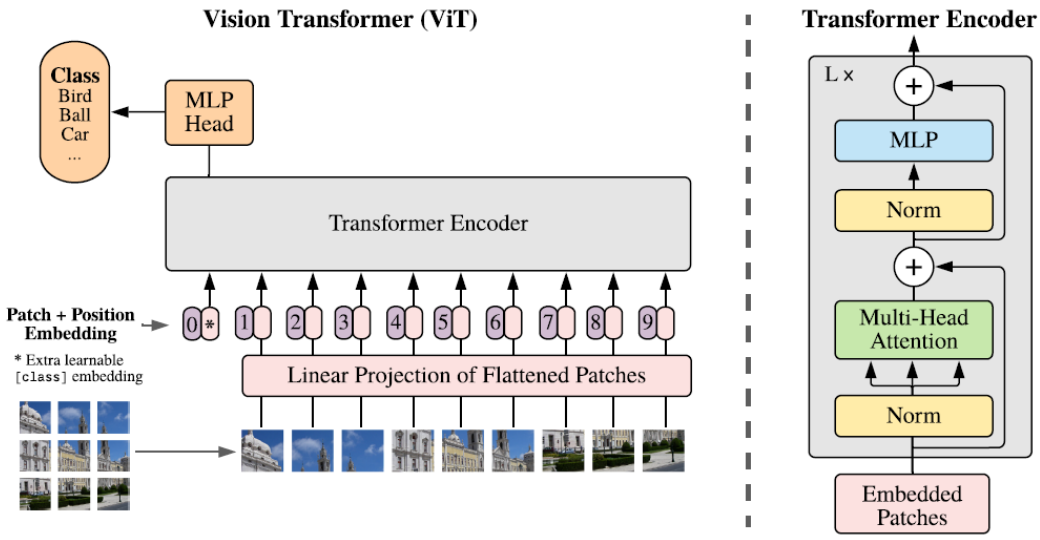


図 2-25 ViT のネットワークアーキテクチャ[3]

[3] 出典 : DOSOVITSKIY, Alexey, et al. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929, 2020.

精度の評価指標は F1、Precision Recall を使用した。各評価指標は次式である。

$$F1 = 2\frac{Precision \cdot Recall}{Precision + Recall}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

TP(TruePositive)：真が＋のデータを＋と推論
FN(FalseNegative)：真が＋のデータを-と推論
FP(FalsePositive)：真が-のデータを＋と推論
TN(TrueNegative)：真が-のデータを-と推論

Precision は適合率と呼ばれ、システムが出力した結果の正しい割合を示す指標である。Recall は再現率と呼ばれ、データ全体のうちどれだけが正しく推論できたかを示す指標である。F1 は適合率と再現率の加重平均であり、二つの指標を合わせた指標である。

検証結果を表 2-8 に示す。各評価指標についてみると、Precision では EfficientNet-b3、EfficientNet-b2、ResNet50 の順で精度が良い。Recall では ResNet50、ResNet34、EfficientNet-b3 の順で精度が良い。これらの結果から、EfficientNet は Precision が良く、ResNet は Recall が良い傾向があることが分かった。Precision と Recall の総合的な指標である F1 では ResNet50、EfficientNet-b3 の順で精度が良い。最新の AI モデルである ViT は F1 で最も精度が悪い。ViT は大規模なデータセットで十分に学習しなければ有効に働かないことが原因と考えられる。処理時間ではベース技術で使用している ResNet34 が最も早かった。以上の結果から、本検証では ResNet50 と EfficientNet-b3 が良いことが分かった。

表 2-8　AI モデルの検証結果　赤字は最も良い数値

AI モデル	精度			処理時間
	Precision[%]	Recall[%]	F1[%]	[ms/枚]
ResNet34	80.4	80.5	80.4	0.273
ResNet50	81.6	81.3	81.4	0.364
EfficientNet-b2	82.2	78.5	79.4	0.446
EfficientNet-b3	84.1	79.9	80.8	0.482
ViT	80.5	76.9	77.7	0.374

2.3.3.2. 入力：高さ情報付与の検証

2.2.3 のモデル正確度検証結果から、ベース技術は単純および複雑家屋形状のモデル正確度が低いことが分かり、家屋形状を陸屋根形状に誤って作成していることから建物形状分類処理の精度が低いことが課題と考えられる。屋根面の全てが平面で構成される陸屋根形状と、屋根面の多くが斜面で構成される家屋形状の違いは、屋根面が平面であるか、斜面であるかを高さ情報で見ればわかりやすい。

本検証では、2.3.3.1 の結果から ResNet50 の転移学習ありのモデルを用いて、画像分類 AI モデルの入力を RGB 色情報と DSM 点群の高さ情報を合わせた 4 次元データとし、RGB 色情報の 3 次元データ入力との比較を行った（図 2-26）。RGB 色情報は 0～255 の値とし、DSM の高さ情報は建物外形情報を用いて取得する建築物周辺の点群から抽出した地面点群の高さ値に対する建築物の高さを 0～1 に正規化した値とした。

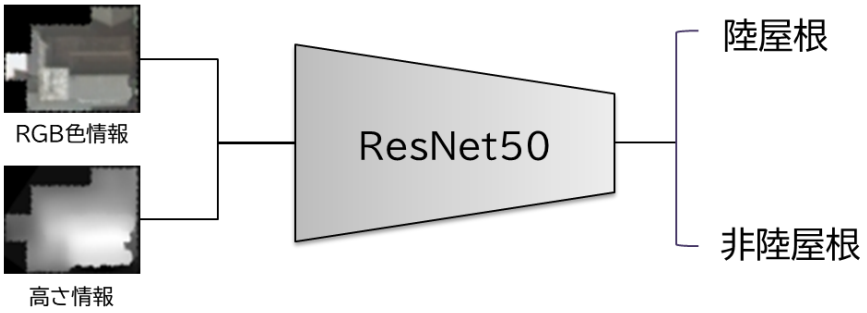


図 2-26　高さ情報を付与した建物形状分類

検証結果を表 2-9 に示す。RGB 色情報のみの 3 次元データを入力したモデルがより良いことが分かった。DSM 点群の高さ情報を付与した場合に精度が低くなるのは、地上画素寸法 25cm の DSM 点群であると建物のエッジ付近が丸みを帯びて正確な形状でないことで、入力情報として良くないことが原因であると考えられる。

表 2-9 高さ情報付与の検証結果　赤字は最も良い数値

入力データ	精度		
	Precision[%]	Recall[%]	F1[%]
R, G, B	83.6	83.1	83.3
R, G, B, 高さ	77.6	77.1	77.3

2.3.4.AI モデルの学習データ追加学習

建物形状分類 AI モデルと屋根線検出 AI モデルのそれぞれについて、ベース技術で学習した地域とは異なる地域の学習データを追加し学習を行った。2.2.4 に示した汎化性能の問題を解決するには、異なる特性を持つ学習データを増やすことが重要である。本節ではベース技術で学習した東京都三鷹市以外の地域を学習させることによる汎化性能の向上を検証した。

建物形状分類 AI モデルは、1 棟ごとに建築物が陸屋根形状であるか、非陸屋根形状（家屋形状とも表記する）であるかの判読が可能な AI モデルである。学習時には、図 2-27 のように建物外形データをもとに建物ごとに画像を切り出し、その建築物が陸屋根形状であるか、非陸屋根形状であるかをラベリングした教師データを学習する。学習した建物形状分類 AI モデルは建築物の画像を入力すると、その建築物が陸屋根形状か非陸屋根形状であるかを出力する。

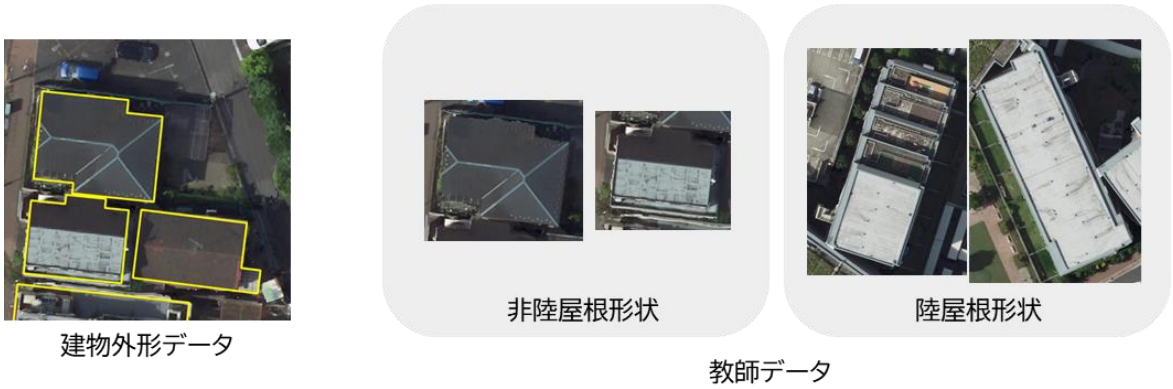


図 2-27　建物形状分類 AI モデルの学習データ例

屋根線検出 AI モデルは、家屋形状である建築物 1 棟ごとに屋根線を検出可能な AI モデルである。学習時には、図 2-28 のように家屋形状である建築物 1 棟ごとに屋根線（屋根面と屋根面の境界）を引いた教師データを学習する。学習した屋根線検出 AI モデルは建築物の画像を入力すると、その建築物の屋根線を出力する。

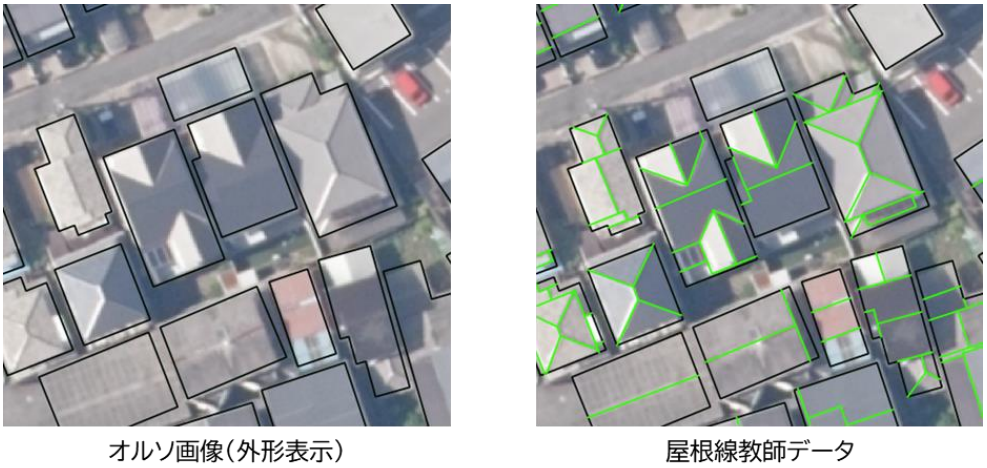


図 2-28　屋根線検出 AI モデルの学習データ例

建物形状分類 AI モデルは、2.3.3 の結果より RGB 色情報を入力とする ResNet50 の転移学習モデルを使用した。

2.3.4.1 に学習及び評価に使用した対象地域、2.3.4.2 に追加学習結果を示す。

2.3.4.1. 対象地域の選定

対象地域には、ベース技術で使用した東京都三鷹市、神奈川県川崎市の他に新たに石川県加賀市と広島県広島市を使用した。石川県加賀市は大聖寺駅北側の住宅が多い地域を、広島県広島市は広島駅周辺を選定した。対象建物の棟数を表 2-10 に示す。

表 2-10　地域別対象棟数

都市		対象棟数
東京都	三鷹市	11,014
神奈川県	川崎市	3,621
石川県	加賀市	6,356
広島県	広島市	17,890

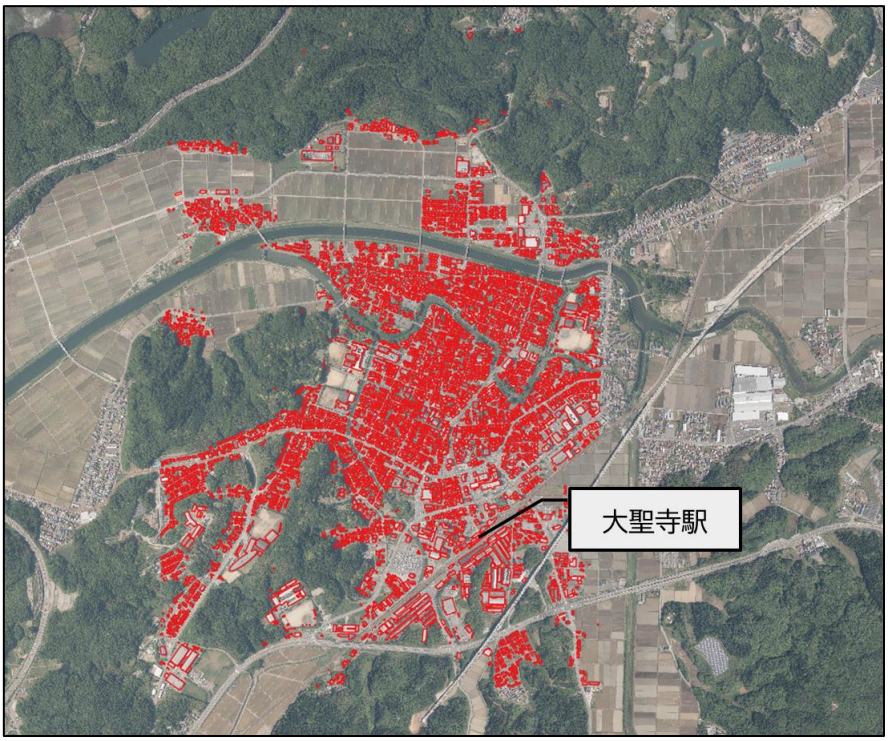


図 2-29　加賀市_検証対象地域

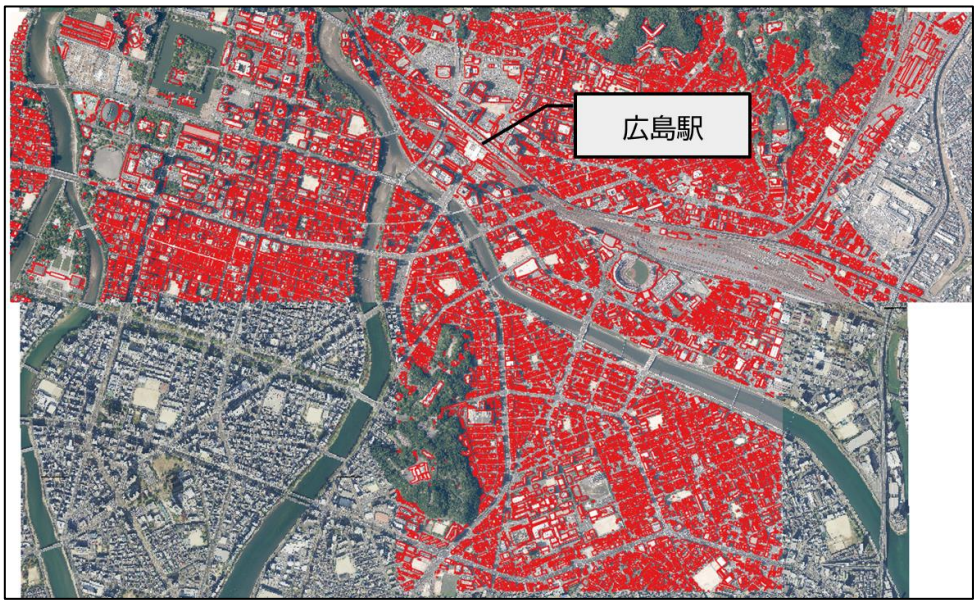


図 2-30　広島市_検証対象地域

2.3.4.2. 学習データ追加学習結果

評価対象地域において、追加学習前の AI モデルと追加学習後の AI モデルでの建物形状分類（(1)）、屋根線検出（(2)）の評価結果を比較した。AI モデルのツール搭載については(3)に記載した。

(1) 建物形状分類 AI モデル

建物形状分類 AI モデルは対象地域の建物を表 2-11 に示すように学習用と評価用で分割して学習と評価を行った。ベース技術では三鷹市を学習しており、R4 年度技術レポートでは三鷹市と川崎市を対象に性能検証を実施済みである。学習データ追加学習では、三鷹市、川崎市、加賀市、広島市を学習した。

表 2-11　建物形状分類 AI モデル_学習及び評価棟数

都市		学習棟数	評価棟数
東京都	三鷹市	10,442	572
神奈川県	川崎市	2,535	1,086
石川県	加賀市	2,302	3,525
広島県	広島市	10,545	4,833

評価結果を表 2-12 に示す。上段 2 段はベース技術で評価を行った三鷹市と川崎市を対象とした評価である。ベース技術がすべての評価指標で性能が優れていることが分かった。次に、下 2 段は新たに対象地域とした加賀市、広島市の評価である。ベース技術と比較して、学習データ追加学習モデルは 10%以上性能が向上したことが分かった。

これらの結果から、学習データ追加学習モデルはより汎用的に高い精度で建物形状分類ができたことが分かった。また、異なる特徴を持つ地域の学習データを追加することによる AI モデルの学習の必要性が分かった。

表 2-12　建物形状分類 AI モデル_評価結果（赤字が性能向上した指標）

学習地域	評価エリア	精度		
		Precision[%]	Recall[%]	F1[%]
三鷹市＋川崎市　[R4 年度]	三鷹市＋川崎市	82.2	80.7	81.4
三鷹市＋川崎市＋加賀市＋広島市	三鷹市＋川崎市	81.2	79.2	80.1
三鷹市＋川崎市　[R4 年度]	加賀市＋広島市	75.7	75.5	75.6
三鷹市＋川崎市＋加賀市＋広島市	加賀市＋広島市	86.9	86.3	86.6

(2) 屋根線検出 AI モデル

屋根線検出 AI モデルは対象地域の建物を表 2-13 に示すように学習用と評価用で分割して学習と評価を行った。ベース技術では三鷹市を学習しており、R4 年度技術レポートでは三鷹市と川崎

市を対象に性能検証を実施済みである。学習データ追加学習では、三鷹市、川崎市、加賀市、広島市を学習した。

表 2-13　屋根線検出 AI モデル_学習・評価棟数

都市		学習棟数	評価棟数
東京都	三鷹市	6,030	532
神奈川県	川崎市	1,035	140
石川県	加賀市	1,934	1,348
広島県	広島市	3,587	2,398

評価結果を表 2-14 に示す。上段 2 段はベース技術で評価を行った三鷹市と川崎市を対象とした評価である。Precision はベース技術が優れており、Recall は学習データ追加モデルが優れる。総合的な指標の F1 は学習データ追加モデルが優れている。次に、下 2 段は新たに対象地域とした加賀市、広島市の評価である。すべての指標において学習データ追加モデルが優れており、Recall では 20%以上性能向上したことが分かった。

評価結果から、学習データ追加モデルは学習データを追加して学習したことにより、大幅に汎用性が向上したことが分かった。

表 2-14　屋根線検出 AI モデル_評価結果（赤字が性能向上した指標）

学習地域	評価エリア	精度		
		Precision[%]	Recall[%]	F1[%]
三鷹市＋川崎市　[R4 年度]	三鷹市＋川崎市	60.9	54.5	57.5
三鷹市＋川崎市＋加賀市＋広島市	三鷹市＋川崎市	59.6	57.9	58.8
三鷹市＋川崎市　[R4 年度]	加賀市＋広島市	53.0	38.7	44.7
三鷹市＋川崎市＋加賀市＋広島市	加賀市＋広島市	63.1	60.3	61.7

(3) AI モデルのツール搭載

これまでに示した AI モデルのデータ追加学習結果をツールに反映するために、ツールに搭載している AI モデルの変更と学習済み AI モデルの更新を行った。この変更及び更新は AI 活用処理の入出力に変更はないため、ツールの挙動に影響を与えることは無く、対象とする AI 活用処理の性能を向上させることが可能である。

2.4. LOD2 建築物モデル自動作成ツールの検証

2.4.1.検証概要

改修を行った LOD2 建築物モデル自動作成ツール（以下、「本ツール」）の有効性を検証するため、モデル正確度検証を行った。

2.4.2.モデル正確度検証

本ツールで自動作成される LOD2 建築物モデルの正確度を評価検証した。2.4.2.1 に評価対象地域の選定、2.4.2.2 に評価基準、2.4.2.3 に評価基準による評価結果、2.4.2.4 にモデル生成例を記載する。

2.4.2.1. 対象地域の選定

検証対象地域は 2.3.4 の学習データ追加学習の効果を確認するために、ベース技術で学習した地域と異なる石川県加賀市及び広島県広島市を選定した。評価対象は図 2-31、図 2-32 に赤く示した建築物である。AI モデルの学習に用いた地域は評価対象外としている。評価対象建築物の建物形状別棟数は表 2-15 のとおりである。

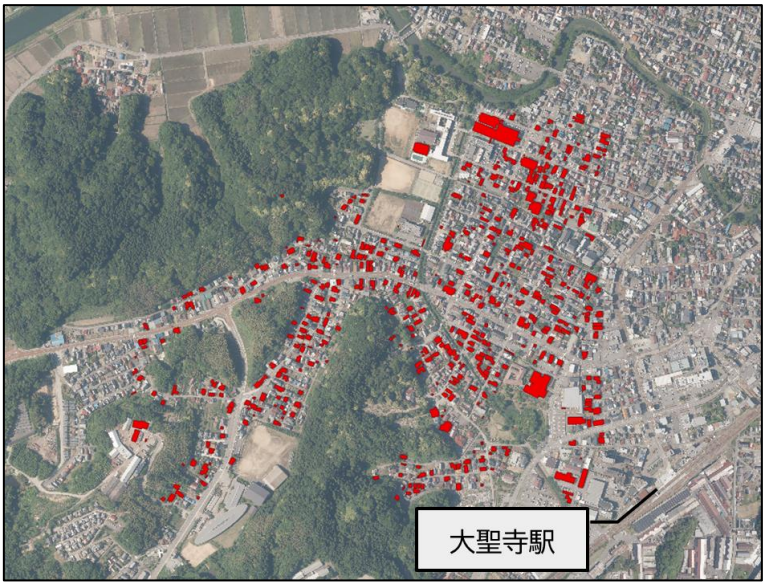


図 2-31　加賀市_評価対象建築物

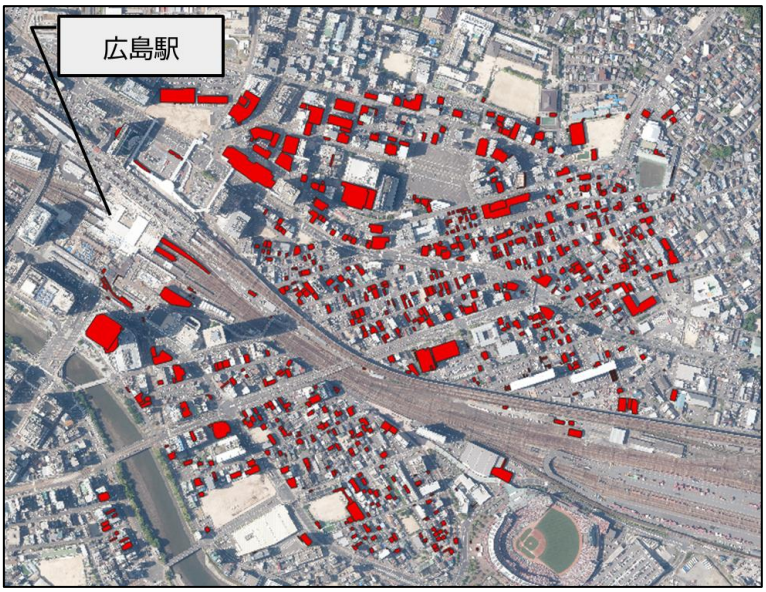


図 2-32　広島市_評価対象建築物

表 2-15　評価対象建築物の建物形状別棟数

評価対象地域	建物形状			合計
	単純形状家屋	複雑形状家屋	ビル	
石川県加賀市	103	308	129	540
広島県広島市	150	231	300	681

2.4.2.2. 評価基準

表 2-16 の評価基準で目視による定性的な正確度評価を行った。経年変化により建築物が存在しない場合や、建物外形と DSM 点群の位置が合わないことによりモデルの生成ができない建築物は対象外とした。

表 2-16　正確度評価基準

モデルの評価ランク	定義
A	概ね正確で修正不要
B	一部不自然な形状があり要修正(少)
C	要修正

2.4.2.3. 評価結果

モデル正確度の評価結果を表 2-17 に示す。表中の形状内割合は、建物形状ごとで A/B/C 評価の割合を算出したものである。

加賀市については、単純形状家屋、複雑形状家屋で評価 C が多い結果であった。複雑形状家屋では評価 C が 81%と非常に多い。加賀市の建物の多くが階層からなる複雑形状であることが原因と考えている。一方で、ビルについては評価 A が多いことが分かった。

広島市については、単純形状家屋では評価 A が最も多く 47%である。複雑形状家屋では、評価 C が最も割合が大きいが、評価 A と評価 B を合わせると評価 C よりも多いため概ね正しくモデルが生成できていることが分かる。ビルについては評価 A が最も多く正しくモデルが生成できたことが分かった。

表 2-17　モデル正確度評価結果

建物形状	評価ランク	石川県加賀市		広島県広島市	
		棟数	形状内割合	棟数	形状内割合
単純形状家屋	A	16	15.8%	70	47.0%
	B	34	33.7%	53	35.6%
	C	51	50.5%	26	17.4%
複雑形状家屋	A	10	3.6%	71	32.0%
	B	43	15.4%	63	28.4%
	C	226	81.0%	88	39.6%
ビル	A	81	64.8%	145	50.9%
	B	26	20.8%	45	15.8%
	C	18	14.4%	95	33.3%

2.4.2.4. モデル生成例

加賀市と広島市の広域のモデル作成結果を図 2-33、図 2-34 に示す。

図 2-33　加賀市モデル作成結果（テクスチャなし）
上段：オルソ画像（黄色は画角）、下段：自動作成モデル

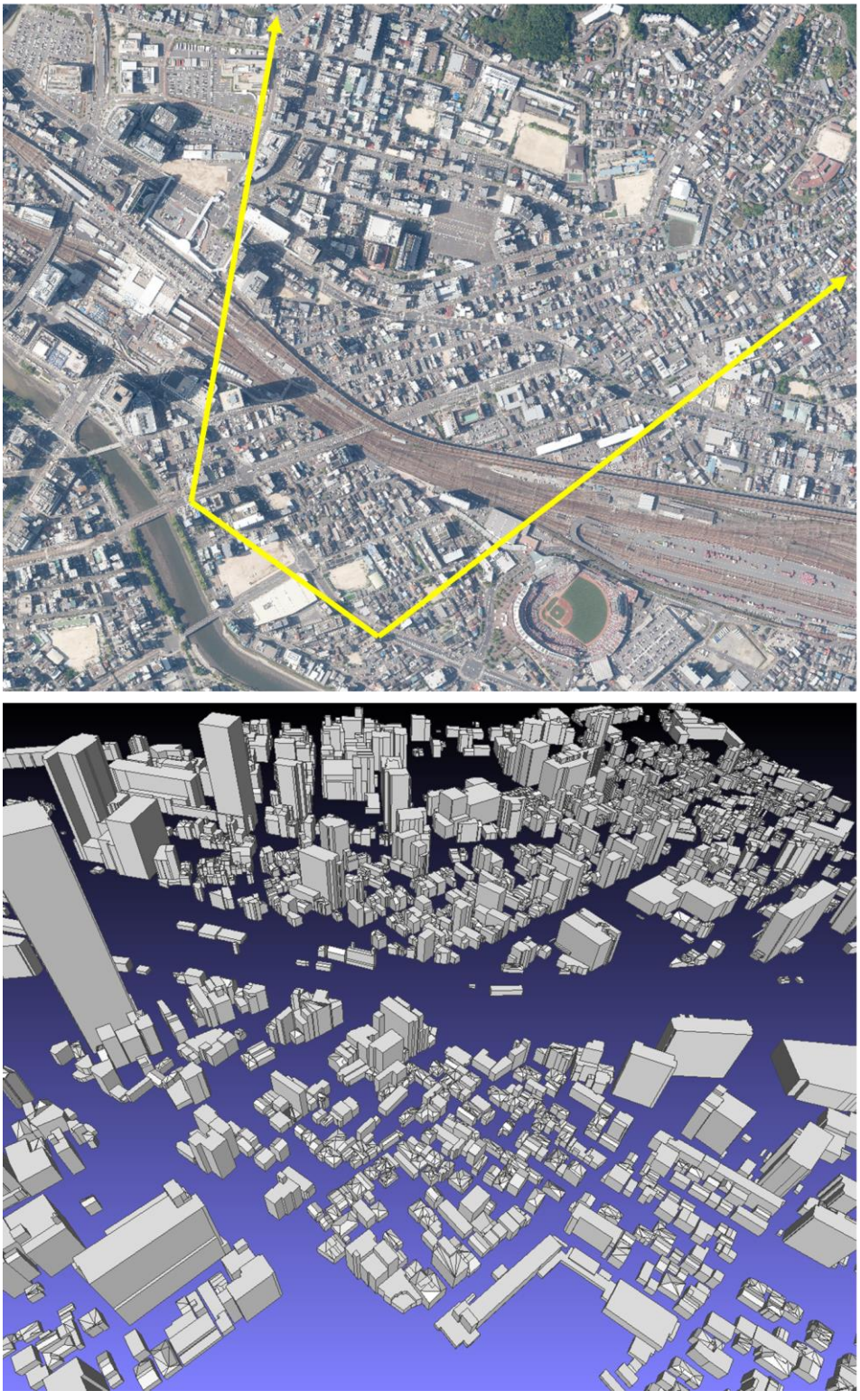
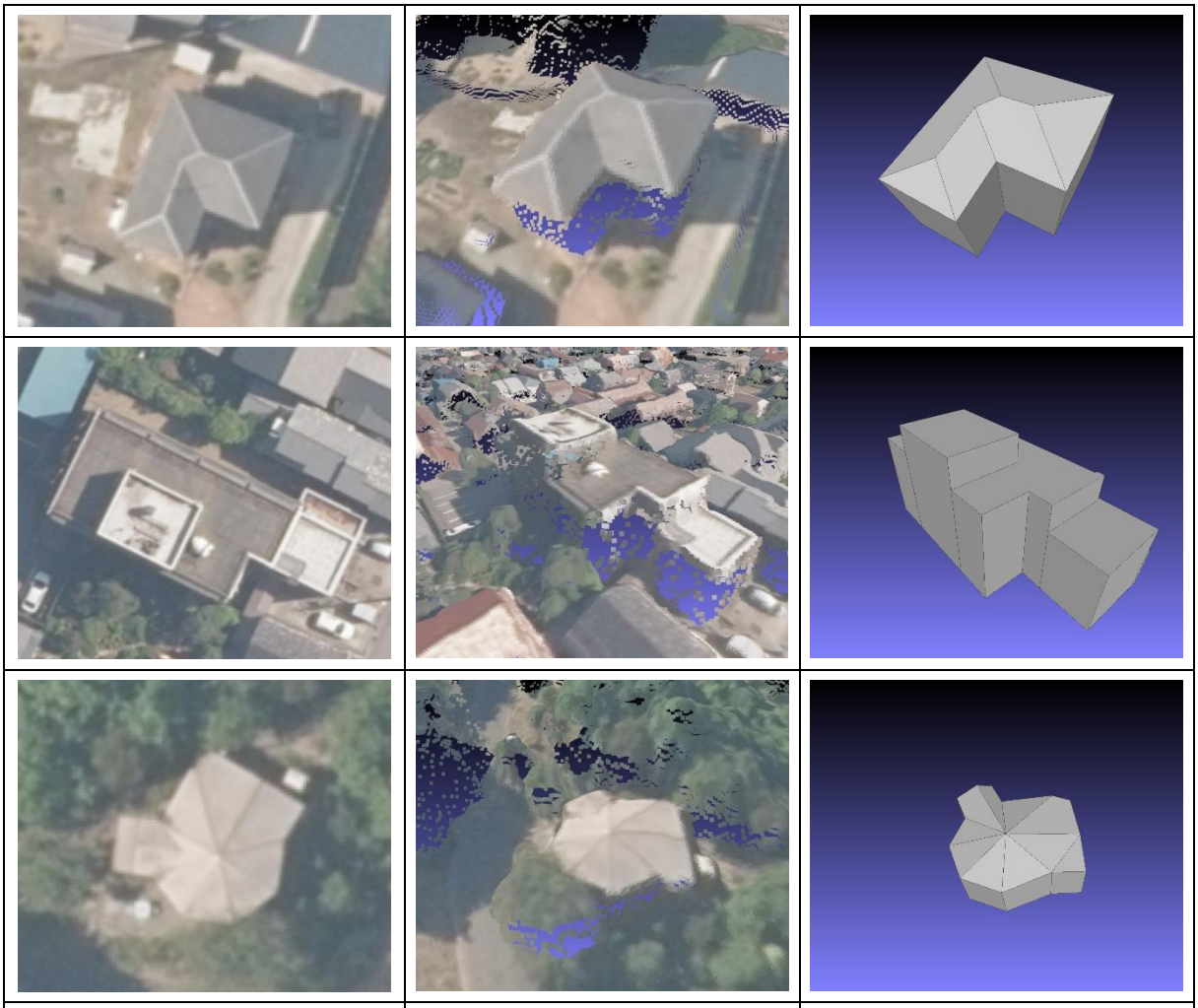


図 2-34　広島市モデル作成結果（テクスチャなし）
上段：オルソ画像（黄色は画角）、下段：自動作成モデル

加賀市と広島市の良好なモデル生成結果の例を図 2-35、図 2-36 に示す。

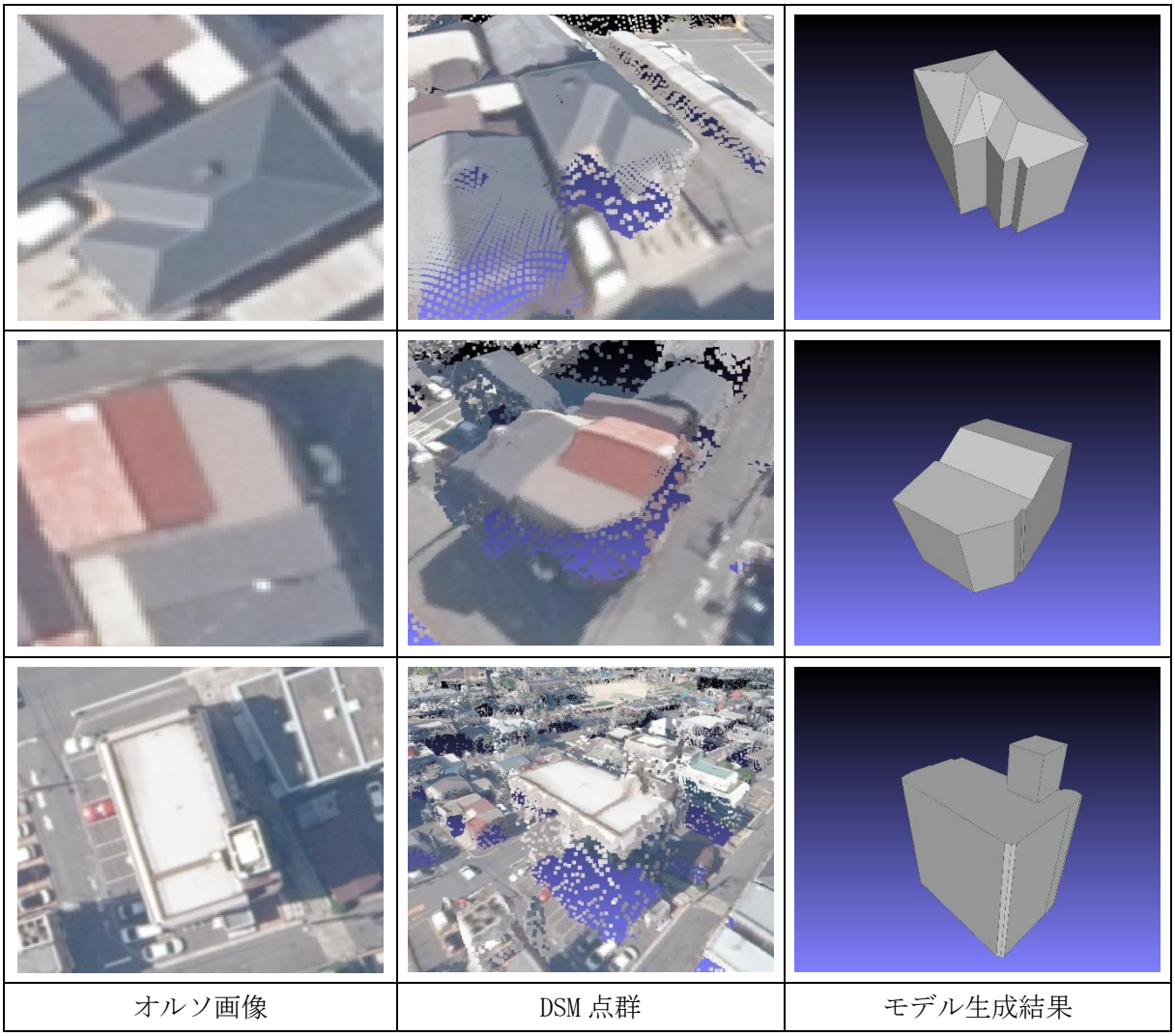


オルソ画像	DSM 点群	モデル生成結果

図 2-35　加賀市_LOD2 建築物モデル自動作成結果例

上段：寄棟屋根の複合形状、中段：ビル、下段：特殊形状建築物

※DSM 点群画像は視認性のために地上画素寸法 16cm を使用



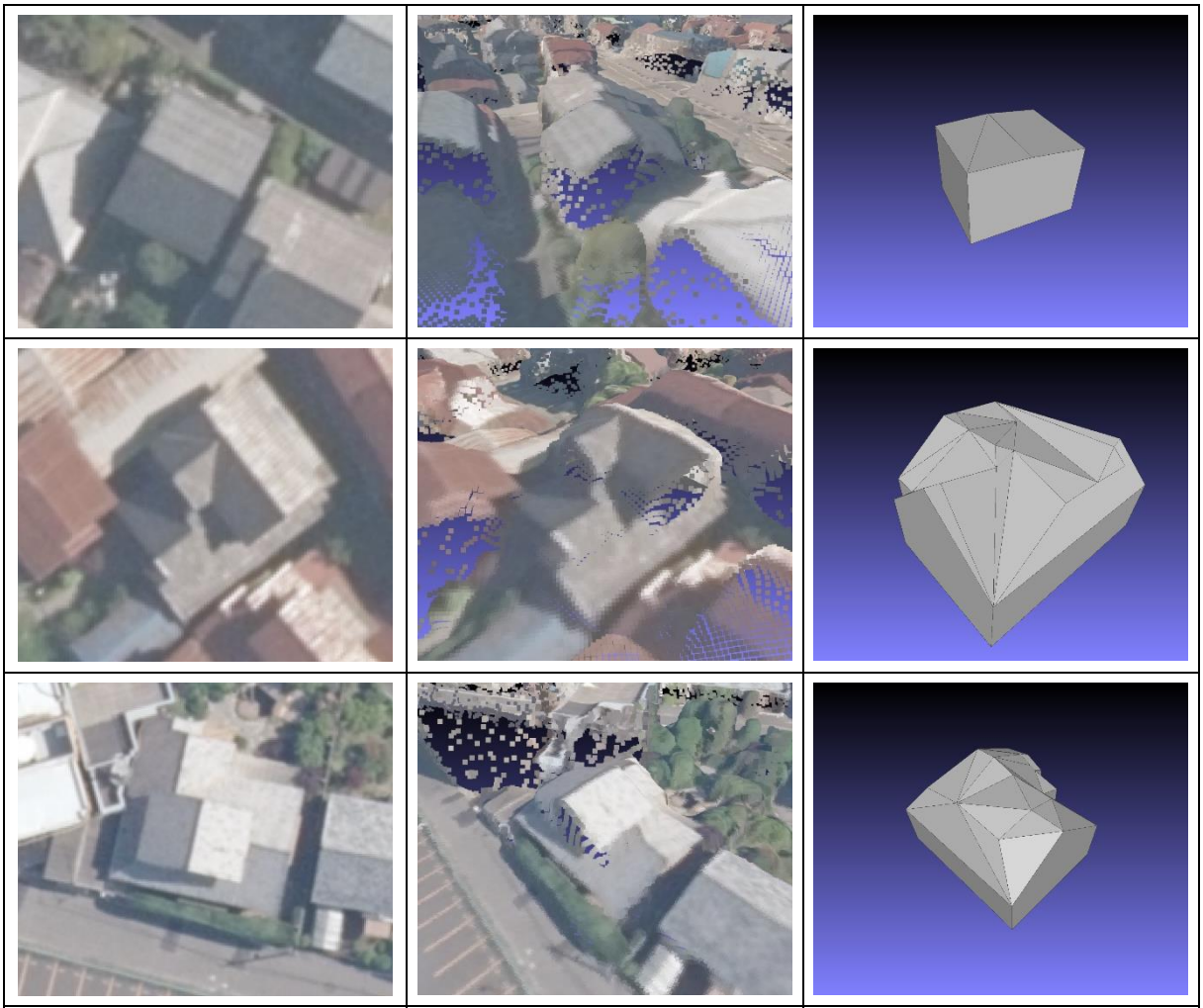
オルソ画像	DSM 点群	モデル生成結果

図 2-36　広島市_LOD2 建築物モデル自動作成結果例

上段：寄棟屋根の複合形状、中段：切妻屋根と陸屋根の複合形状、下段：ビル

※DSM 点群画像は視認性のために地上画素寸法 16cm を使用

加賀市と広島市の課題が残るモデル生成結果の例を図 2-37 に示す。



オルソ画像	DSM 点群	モデル生成結果

図 2-37　LOD2 建築物モデル自動作成の失敗例

上段：切妻屋根建築物（加賀市）、中段：階層のある建築物（加賀市）

下段：階層のある建築物（広島市）

※DSM 点群画像は視認性のために地上画素寸法 16cm を使用

単純形状家屋は概ね良好なモデル生成結果を得られていた。モデル生成が失敗した中には陸屋根形状の建築物に間違えた例や、屋根線は正しく検出できているものの屋根の高さ付与が十分にできておらず平面的な形状になって生成が失敗した例（図 2-37 上段）などが確認できた。前者は建物形状分類処理時に AI モデルによる分類が誤ったことが原因である。後者は高さ付与時に屋根の高さ（Z 座標）最低地点と最高地点の取得が上手く働かなかったことや、DSM 点群の精度やノイズが原因と考えられる。

複雑形状家屋では図 2-35 上段や図 2-36 上中段のように基本的な屋根形状（図 2-5 左）が 2 つ程度で組み合わされている建物形状では良好なモデル生成結果が得られた。また、図 2-35 下段のような特殊な建物形状であってもモデルが綺麗に生成されることが分かった。一方で、モデル生成が失敗したものの多くは、図 2-37 中下段のような階層構造を持つ形状であることが分かった。本ツールは階層構造の LOD2 建築物モデルの自動作成には対応していないためモデルの生成が失敗してしまう。加賀市では特に階層構造を持つ建築物が多く、この点から加賀市における複雑形状家屋の評価が低くなったことが分かった。

ビル形状では良好な結果が得られた。

2.4.3.まとめと課題

モデルの正確度検証結果から、ビルと単純形状の家屋では概ね良好なモデル生成結果が得られた。一方で、複雑な形状の家屋では作成が失敗した例が多く見られた。複雑な形状の家屋の微細な屋根変化を捉えるためには、屋根の高さに関する高精度の情報が必要である。現状の手法では、高さ情報は DSM 点群から取得している。したがって、高解像度の DSM 点群の使用や航空レーザ点群との融合による精度向上が考えられる。また、最新の機械学習モデルを適用することで、DSM 点群の高解像度化や高さを推定することも今後の取り組みである。

3. LOD1-2 道路モデル自動作成ツールの開発

3.1. 開発の目的

3D 都市モデルの道路モデルの作成では手作業によるジオメトリ作成が必要な部分があり、コストや期間の低減化が課題である。LOD1 道路モデル作成では、DM データの道路縁から道路面を作成し、車道交差部や道路構造変化点で分離する作業に人手による部分が多くある。LOD2 道路モデル作成では、車道・歩道等を人が判読し、分離する必要がある。このような道路モデル作成の工程を幾何処理又は AI による判読を利用したツールを開発することで効率化することが本開発の目的である。

3.2. 事前調査

要件定義に当たり、データ整備事業者へ、利用データと現行作業についてヒアリングを実施した。現行の工程で手作業が多い作業を中心に自動化することにより、作業の効率化が期待されるため、データ整備事業者等に対して、ヒアリング調査を実施した。

3.2.1.ヒアリングの実施

3.2.1.1. ヒアリング内容

ヒアリングは以下の内容について調査用紙に記載する形式で実施した。

1. 作成実績
2. 利用データ
3. 利用ソフトウェア、手作業の有無
4. 手作業が多い作業
5. 現状の課題・問題点
6. 自由意見

次ページに調査用紙を添付する。

道路モデル作成に関する調査

2023 年 5 月　アジア航測株式会社

◆道路モデル作成に関する調査　ご協力のお願い◆

2023年度のProject PLATEAUの仕様拡張・LOD2自動生成ツールの業務において、アジア航測では道路LOD1-2の自動生成ツールの開発に取り組んでいます。

そこで道路モデル作成の事業者の皆様にヒアリングを行い、自動生成ツールの要件を抽出しています。実際の作業に即した有用性の高いツールとするために、皆様へ本調査へのご協力をお願い申し上げます。

1. 道路 LOD1、LOD2、LOD3 の作成実績について選択ください。(複数選択可)
   - ☐ LOD1 作成実績あり
   - ☐ LOD2 作成実績あり
   - ☐ LOD3 作成実績あり

2. 道路モデル LOD1 作成にあたり、形状(道路縁)の元データとして使用しているデータを選択してください。(複数選択可)
   - ☐ DM データ
   - ☐ 基盤地図情報
   - ☐ その他

3. 【LOD2作成実績ありと回答いただいた方のみ】道路モデル LOD2 作成にあたり、元データとして使用しているデータを選択してください。(複数選択可)
   - ☐ 道路 LOD1 データ(使用している場合は以下より形式を選択してください)
     - ☐ CityGML
     - ☐ シェープファイル
     - ☐ その他の形式
   - ☐ 航空写真
   - ☐ 点群データ
     - ☐ MMS
     - ☐ 航空レーザー
     - ☐ その他点群データ
   - ☐ その他

4. 道路モデル作成にあたり、作成した道路面の形状の確認に用いるなど、補助的に使用しているデータがあればご記入ください。(例:航空写真、点群データ、地理院地図、道路台帳平面図)。

1

図 3-1　調査用紙　1 ページ目

5. 道路モデルの作成作業に利用しているソフトウェアを教えてください。差し支えなければ市販ソフト名・OSS ソフト名（例：ArcGIS, QGIS・・・）もご記入ください。

① 道路のラインデータから道路面データを作成する作業

☐ 市販・OSS ソフト
☐ 自社製ソフト
☐ 作業実施なし

② 道路面の交差部などを分割する作業

☐ 市販・OSS ソフト
☐ 自社製ソフト
☐ 作業実施なし

③ 属性を付与する作業

☐ 市販・OSS ソフト
☐ 自社製ソフト
☐ 作業実施なし

④ CityGML 形式へ変換する作業

☐ 市販・OSS ソフト
☐ 自社製ソフト
☐ 作業実施なし

6. 手作業が多い作業があれば、教えてください。

7. 現状の作業での課題・問題点があれば、教えてください。

8. 道路モデル作成について、自由な意見をご記入ください。

質問は以上です。ご協力ありがとうございました。

図 3-2　調査用紙　2 ページ目

3.2.1.2. ヒアリング結果

(1) 作成実績

ヒアリングを行った 3 社（国際航業株式会社、朝日航洋株式会社、株式会社パスコ）とも LOD1 道路、LOD2 道路の作成実績があった。

(2) 利用データ

利用データについて、ヒアリング結果を以下に示す。

① 入力データ

- LOD1 は基盤地図情報、DM データを使っている。
- LOD2 は DM データ、LOD1 道路のシェープファイルを使っている。
- 自治体ごとにどのデータを使うかが異なる。道路のベクトルデータとして複数の種類を組み合わせることはない。
- 公共測量成果を用いる。
- 建物と道路で共通のデータを使用している。
- シェープファイルを入力としている。（DM データはまずシェープファイルに変換する）
- DM データから形状のみを使っている。
- 単線（徒歩道）は対象外とする。

② 補助的に道路モデル作成時に使っているデータとその利用目的

- 航空写真：目視で参考情報として利用
- 地理院タイル：目視で参考情報として利用
- 道路中心線：交差部の抽出に利用
- 建物データ：建物がある場所は道路ではないという判断に利用
- 道路台帳：車道部と歩道部の分割の参考

(3) 利用ソフトウェア、手作業の有無

表 3-1 に工程ごとの利用ソフトウェア、手作業の有無を示す。

表 3-1　利用ソフトウェア、手作業の有無

工程	ソフトウェア	手作業の有無
道路のラインデータから道路面データを作成する作業	ArcGIS	有
道路面の交差部などを分割する作業	ArcGIS、QGIS、AutoCAD	有（手作業が多い）
属性を付与する作業	ArcGIS	有
CityGML 形式へ変換する作業	FME	無

(4) 手作業が多い作業

- 道路の区切り
- 道路面の作成
- 属性付与
- 階層ごとのデータ作成

(5) 現状の課題・問題点

- 令和 2 年度のデータは立体交差部が階層となっていなかったり、属性が付与されていなかったりする場合があり、一部の更新を行った際に自治体内で仕様が一律とならないことが生じる。
- 付与する属性が多いため、属性の内容の検討が必要である。
- 交差部等の分割を行う際、市販ソフトウェアでは仕様に沿った分割とならないため、手作業が必要である。

3.3. LOD1 道路モデル自動作成ツールの開発

LOD1 道路モデル自動作成ツールを本章では本システムという。

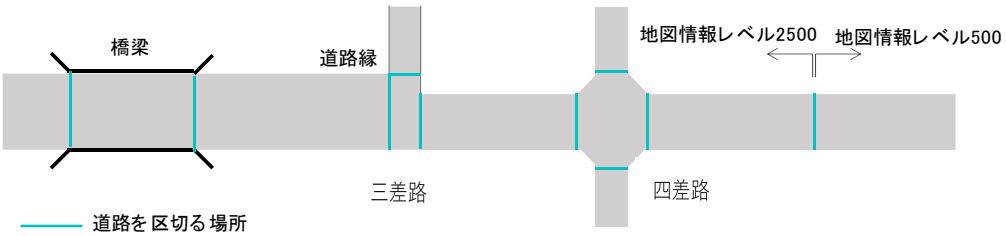
3.3.1.LOD1 道路モデルの仕様

本システムでは「3D 都市モデル標準製品仕様書」「3D 都市モデル標準作業手順書」に従い、DM データをインプットとして LOD1 道路面モデルを作成する。

LOD1 道路面モデルは、道路縁により囲まれた範囲の面を作成し、その面を仕様（表 3-2）に定める以下の場所で区切る必要がある。

- 車道交差部(十字路、丁字路、その他 2 つ以上の道路が交わる部分)
- 道路構造(トンネル、橋梁等)の変化点
- 位置正確度や取得方法が変わる場所

表 3-2　LOD1 道路仕様[4]

	LOD1
取得例	
説明	道路縁により囲まれた範囲を面として取得し、以下の場所で区切る。 4　交差部（四差路、多差路及び三差路） 5　道路構造（トンネル、橋梁）が変化する場所 6　位置正確度や取得方法が変わる場所 高さは 0 とする。

3.3.1.　本システムの開発目的

本システムは道路モデルの仕様及び現行作業調査に基づき幾何処理をツール化し、手作業によるコストや期間が課題となっている LOD1 道路モデル作成を効率化することを目的とする。

3.3.2.現行作業調査

現行作業の詳細な把握のため、その一例として LOD1 道路モデル作成の作業フローを確認した。LOD1 道路モデルは DM データの道路縁を入力として作成するが、DM データから GIS ソフトウェア等で編集可能なシェープファイルに変換し、そのシェープファイルのジオメトリや属性を編集し、作成されている。DM データは道路縁をラインデータで持っているため、シェープファイルに変換

[4] 国土交通省都市局 Project PLATEAU,
3D 都市モデル標準製品仕様書 ver.3.5（2024.3）p.159. 表 4-17 引用

した後に道路面のポリゴンを生成し、そのポリゴンを仕様に定める位置で区切り、道路面を作成する処理が必要である。シェープファイルでの編集が完了した後に CityGML への変換が行われている。

LOD1 道路モデルの CityGML を作成する手順は次の①～⑤となる。（図 3-3）

① 道路縁のラインデータがループで囲われている箇所は街区として範囲全体のポリゴンから抜くことで道路部分のポリゴン（道路面データ）を抽出する。
② ①のデータをもとに交差部を区切る。その際に航空写真なども用いて、交差部を判断する。
③ 作成したシェープファイルを目視チェックする。その際に航空写真なども用いて確認する。
④ ③までの処理を経て、道路形状の処理済みのシェープファイルに属性付与する。
⑤ FME で CityGML に変換する。

道路 LOD１の作成は ArcGIS 等の GIS ソフトで①～④を自動・手動を併用して実施している。作成だけでなく、チェックにも多くの時間を要している。また⑤の CityGML 変換は FME を用いている。

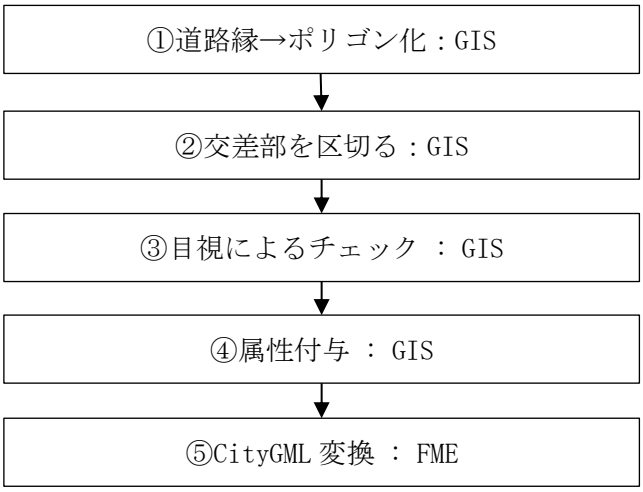


図 3-3　LOD1 道路モデル作成の作業フロー

図 3-4 に道路縁から道路面データを抽出する処理の流れを示す。

図 3-4　道路面データの抽出

3.3.3.要件の抽出

事前調査から抽出されたシステムの要件は表 3-3 のとおりである。

表 3-3　要件一覧

項目	要件
入力データ	DM データ又はシェープファイル
出力データ	シェープファイル、CityGML ファイル
機能	道路縁から道路面を生成する
	車道交差部を抽出し、道路面を分離する
	立体交差部を抽出する
	立体交差部、構造変化点（橋梁、トンネル、アンダーパス）で道路面を分離する
	シェープファイルから CityGML に変換する
	道路面が正常に作成されなかった箇所を抽出する

3.3.4.前提、制約条件

道路形状によっては自動化が困難なケースや、属性付与は人により判断が必要なケースがあるため、道路モデルの CityGML まで一連で行うツールよりも、中間で形状の修正や属性付与作業を行うことを想定したツールが有用である。

属性付与については、自治体により付与する属性や DM データの持つ属性が異なること、空間結合では解決できない属性もあることから、自動で CityGML に属性を付与するのは困難である。そこで、本システムで形状作成したシェープファイルに対して GIS ソフトウェアで属性を付与し、属性を付与後のシェープファイルの符号化処理を本システムで行うことを想定する。

形状処理については、以下の図 3-5～図 3-7 に示すように、自動処理が困難と見られる道路形状があるため、手動による編集が発生する可能性がある。編集作業は GIS ソフトウェアで行うことから、本自動化ツールではシェープファイルを出力ファイルとした。

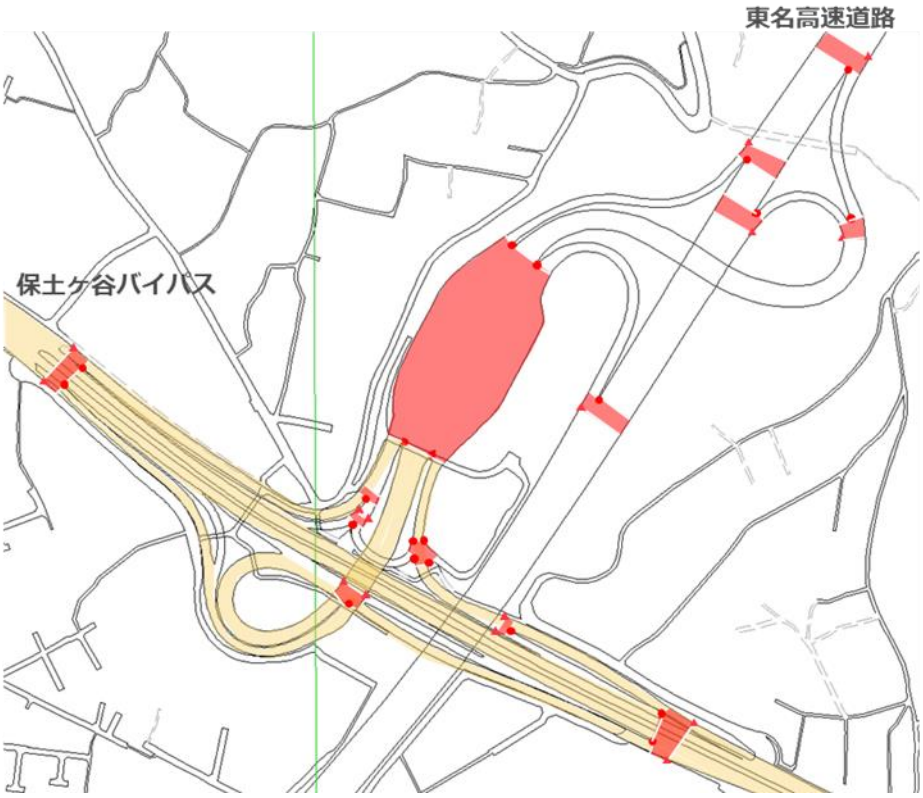


図 3-5　ジャンクションの交差部抽出例

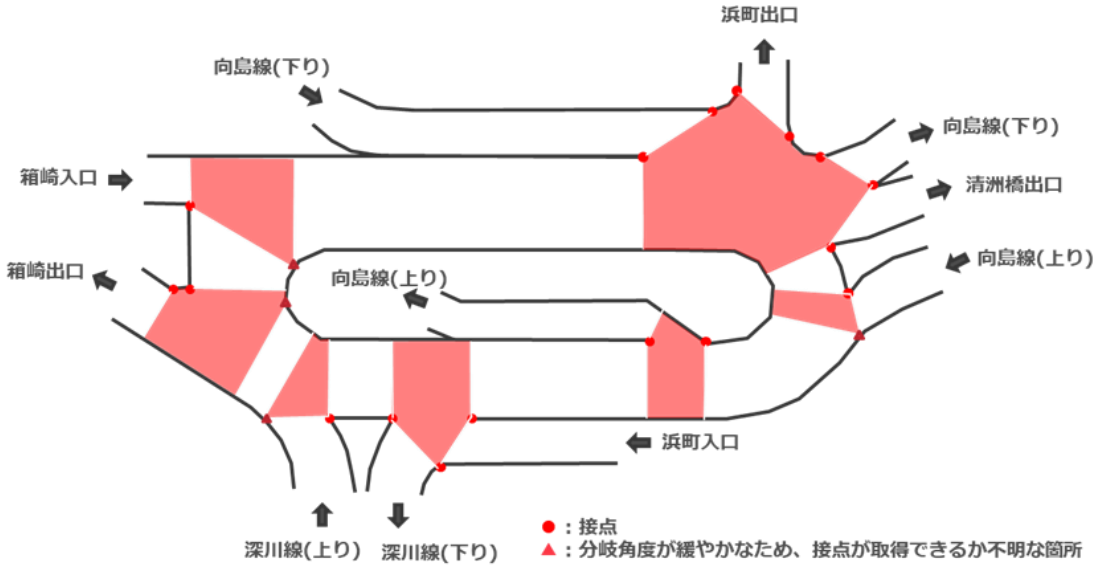


図 3-6　ロータリーの交差部抽出例

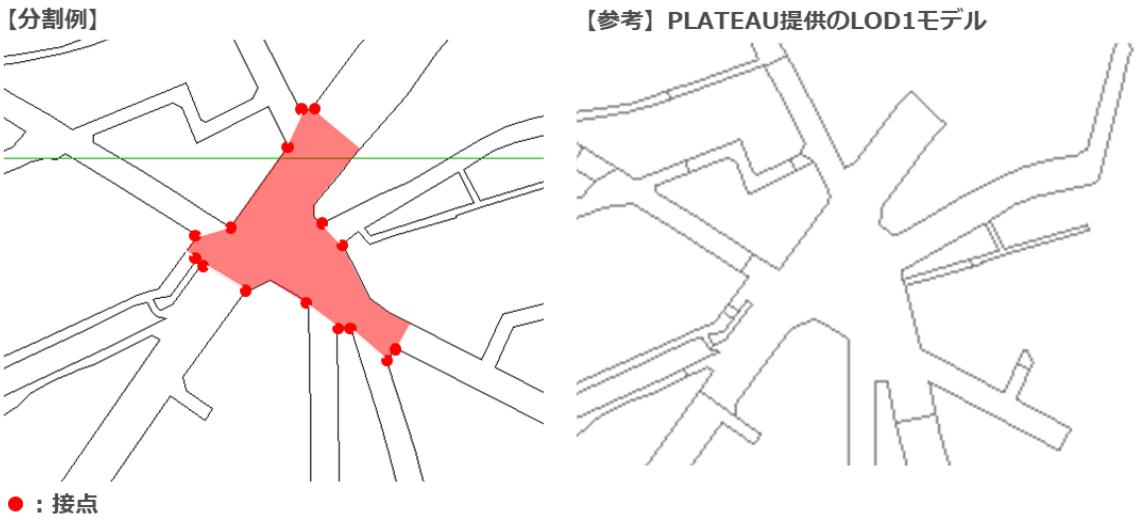


図 3-7　複叉路の交差部抽出例

3.3.5.本システムを適用したフロー

本システムを使用した場合の業務フローを図 3-8 に示す。

本システムでは、道路縁のシェープファイルから LOD1 道路モデルのシェープファイルを作成する部分を「LOD1 道路面生成ツール」として、LOD1 道路モデルのシェープファイルを CityGML ファイルに変換する部分を「LOD1 道路モデル CityGML 変換ツール」として自動化する。LOD1 道路面生成ツールでは、複雑な形状の道路など自動で道路面を作成できない場合があるため、自動作成した LOD1 道路モデルのシェープファイルに対して手動修正を行う必要がある。また、属性付与に関しては、自治体による属性の違いやユーザによる判断を介する必要がある属性があるため、LOD1 道路モデルのシェープファイルに対する手動での属性付与を想定したフローとする。設定する属性は自治体ごとに異なるため、ユーザが設定した属性名と CityGML の属性名の対応情報ファイルを用意した上で、LOD1 道路モデル CityGML 変換ツールを用いて LOD1 道路モデルのシェープファイルを CityGML ファイルに変換する。

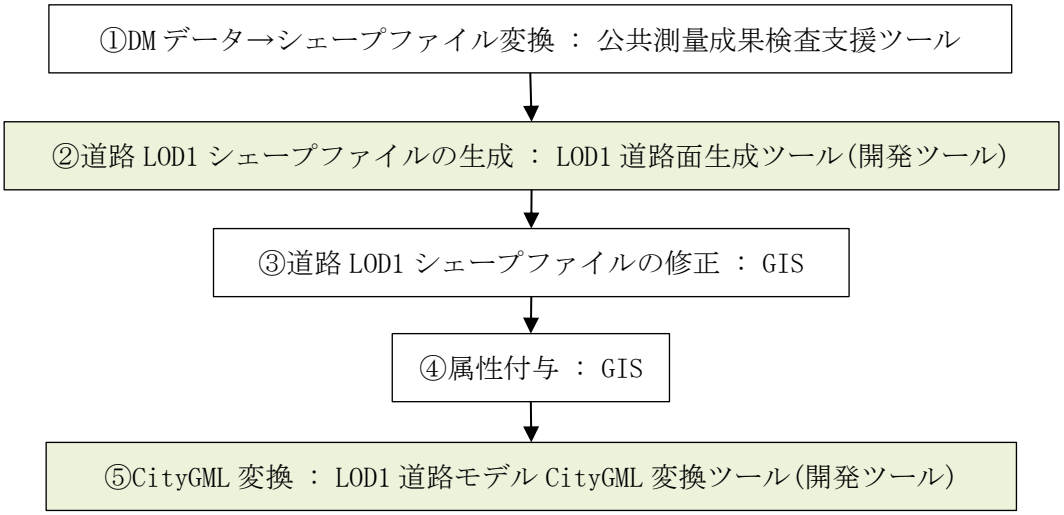


図 3-8　本システム導入後の業務フロー

3.3.6.システム構成

本システムのシステム構成図を図 3-9 に示す。本システムは、LOD1 道路面生成ツールと LOD1 道路モデル CityGML 変換ツールで構成し、道路縁から LOD1 の道路 CityGML データを作成する。

LOD1 道路面生成ツールは、道路縁から LOD1 道路面モデルを作成し、シェープファイルに出力する機能を有する。また、作成した道路面モデルのエラー確認を行い、道路面モデルに不備がある場合は、不備が発生している道路面モデルの位置情報を出力する。

LOD1 道路モデル CityGML 変換ツールは、シェープファイル形式の LOD1 道路面モデルデータを CityGML ファイルに変換する機能を有する。なお、CityGML の道路面情報に付与する属性情報は、シェープファイルに記載されている属性情報を使用する。CityGML の属性名とシェープファイルの属性名の対応付けに関しては、別途入力するマップファイルに対応付け情報が記載されているものとする。

本システムは Visual Studio C++を用いて開発した。開発環境、利用ライブラリについては 3.3.9 に記載する。

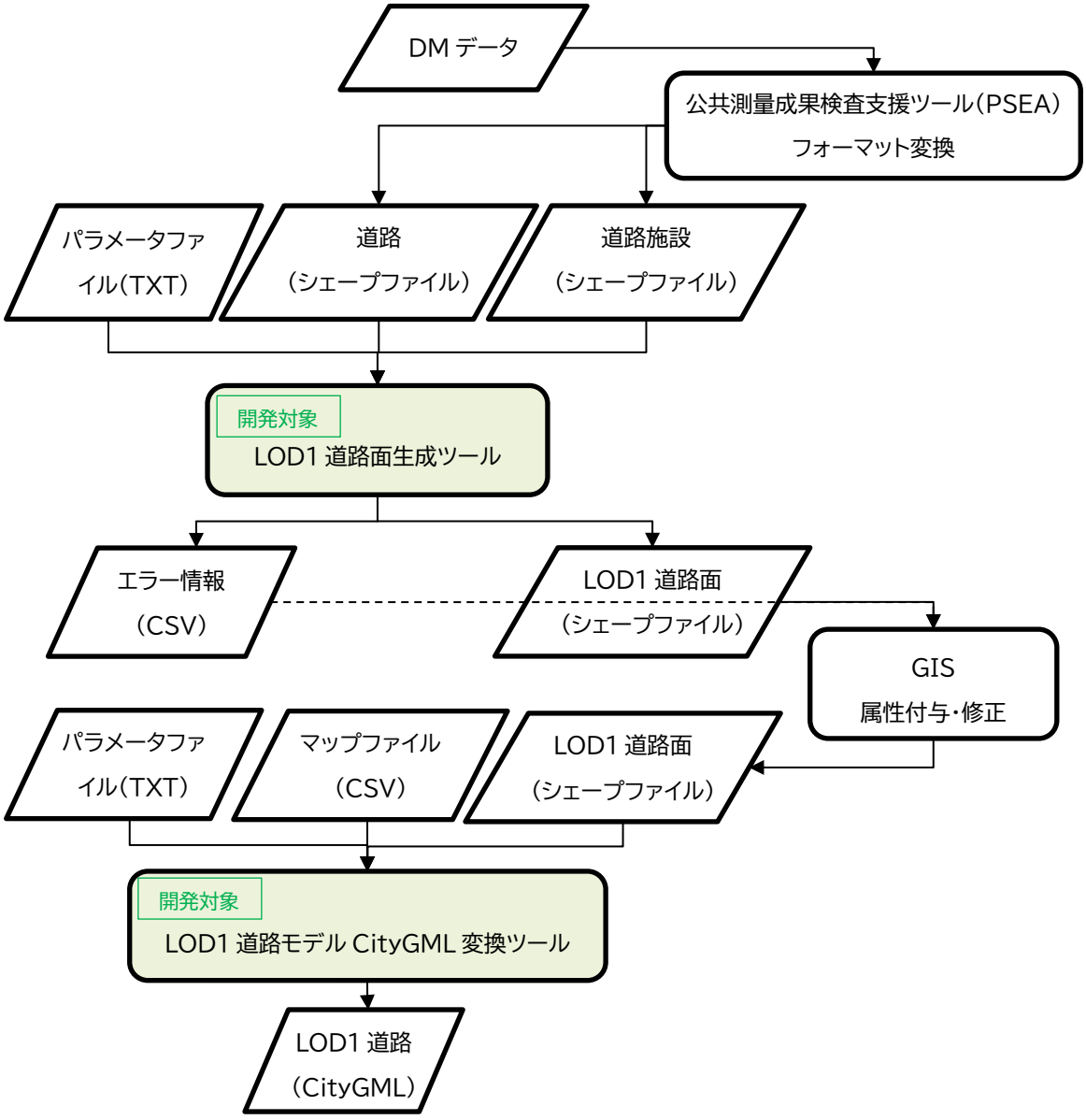


図 3-9　システム構成図

3.3.7.システムインタフェース

3.3.7.1. 外部インタフェース

- **公共測量成果検査支援ツール(PSEA)**

  本システムに入力する前に、DM データをシェープファイルに変換する。

  公共測量成果検査支援ツール ｜ 国土地理院（gsi.go.jp）

  https://psgsv2.gsi.go.jp/koukyou/public/sien/pindex.html#psea03

3.3.7.2. 内部インタフェース

内部処理でのファイルのやり取りはシェープファイルとする。

3.3.7.3. ユーザインタフェース

本システムの実行は、コマンド プロンプトを使用して下記のコマンドで行うものとする。

[LOD1 道路面生成ツール]

> AutoCreateLod1Road.exe create_param.txt

AutoCreateLod1Road.exe ： LOD1 道路面生成ツールの実行ファイル

create_param.txt ： 設定パラメータファイル

[LOD1 道路モデル CityGML 変換ツール]

> ConvertShapeToCityGML.exe conv_param.txt

ConvertShapeToCityGML.exe ： CityGML 変換ツールの実行ファイル

conv_param.txt ： 設定パラメータファイル

3.3.8.動作環境

本システムの推奨環境を表 3-4、必要環境を表 3-5 に示す。

表 3-4　推奨環境

OS	Windows 10 /11
CPU	Intel® Core™ i7 以上
Memory	32GB 以上
HDD 空き容量	5GB 以上
ネットワーク	不要

表 3-5　必要環境

OS	Windows 10 /11
CPU	Intel® Core™ i7 以上
Memory	16GB 以上
HDD 空き容量	1GB 以上　※データ容量による
ネットワーク	不要

3.3.9.開発環境

3.3.9.1. 開発環境

本システムの開発環境を表 3-6 に示す。

表 3-6　開発環境

OS	Windows 10 Pro
開発環境	Visual Studio 2019
言語	C++
CPU	Intel® Core™ i7
Memory	16GB

3.3.9.2. 利用ライブラリ

利用したライブラリを表 3-7 に示す。

表 3-7　利用ライブラリ

ライブラリ	用途
Boost C++ Libraries	形状処理
shapelib	シェープファイルの読み書き
concaveman-cpp	凹包作成

3.3.10.LOD1 道路面生成ツール

3.3.10.1.機能一覧

LOD1 道路面生成ツールの機能を表 3-8 に示す。各処理の詳細は 3.3.10.4 機能詳細に記載する。

表 3-8　機能一覧

No.	機能
1	パラメータファイルを読み込む。 パラメータファイルを用いて入出力データのパス、平面直角座標系の系番号を指定する。
2	入力データ(シェープファイル)を読み込む。 DM データの道路と道路施設を予めシェープファイルに変換して入力データとする。 道路のシェープファイルからは、道路縁の形状情報と形状処理に利用する属性を読み込む。道路施設のシェープファイルからは、交差部の参考情報として利用する道路施設情報(道路橋、トンネル)を読み込む。
3	立体交差部を分離する。 入力データは 2 次元平面上の線データで高さ情報がないため、実際には立体交差してい

	る道路が DM データ上では平面交差している。道路縁を接続する際に誤った接続の原因となるため、階層ごとに分離する。
4	道路縁を接続し、ループ化する。 道路縁を接続して道路縁のループを作成する。道路縁のポリラインは 1 街区分の道路縁が 1 レコードに保存されていない場合もある。1 街区分のポリゴンを作成するために連続する道路縁を結合する。
5	道路縁から道路面を生成する。 4 で結合した道路縁を利用して、道路面を作成する。
6	道路中心線を生成する。 道路中心線の交点を利用することで道路の交差地点を抽出することが可能であるが、DM データには道路中心線が存在しないため、道路中心線を作成する。
7	交差点位置を抽出する。 交差点位置として 6 で作成した道路中心線が交差する位置、接する位置を取得する。
8	車道交差部に該当する道路面を分割する。 交差点位置を用いて車道交差部の領域を形成する分割線を作成し、車道交差部に該当する道路面を分割する。
9	立体交差部、構造変化点(橋梁、トンネル、アンダーパス)で道路面を分割する。 車道交差部以外の道路面に対して、DM データの道路施設から取得した高架橋やトンネルの情報を用いて分割する。
10	道路面が正常に作成されなかった箇所を抽出する。 道路面が正常に作成できない場合にそのエラーのタイプと座標をファイルに出力する。
11	道路面をシェープファイルに出力する。 道路面をシングルポリゴンとしてシェープファイルで出力する。道路の形状属性(トンネル、高架橋、アンダーパス、交差部)を付与する。

LOD1 道路面生成ツールが作成する LOD1 道路面モデルには、CityGML の uro:RoadStructureAttribute の uro:sectionType 属性に該当する道路形状の属性(トンネル、高架橋、アンダーパス、交差部)を付与する。

処理性能の問題上、入力シェープファイルの全範囲の道路縁から一括で LOD1 道路モデル面を生成することが困難な場合は、道路縁を一定の範囲で分割してから道路面モデルを作成する。なお、道路縁を分割して道路面モデルを作成する場合は、モデル作成対象範囲と 8 近傍の範囲を使用することで、作成する道路モデルがモデル作成対象範囲の境界で不正に分割されることを防ぐ。

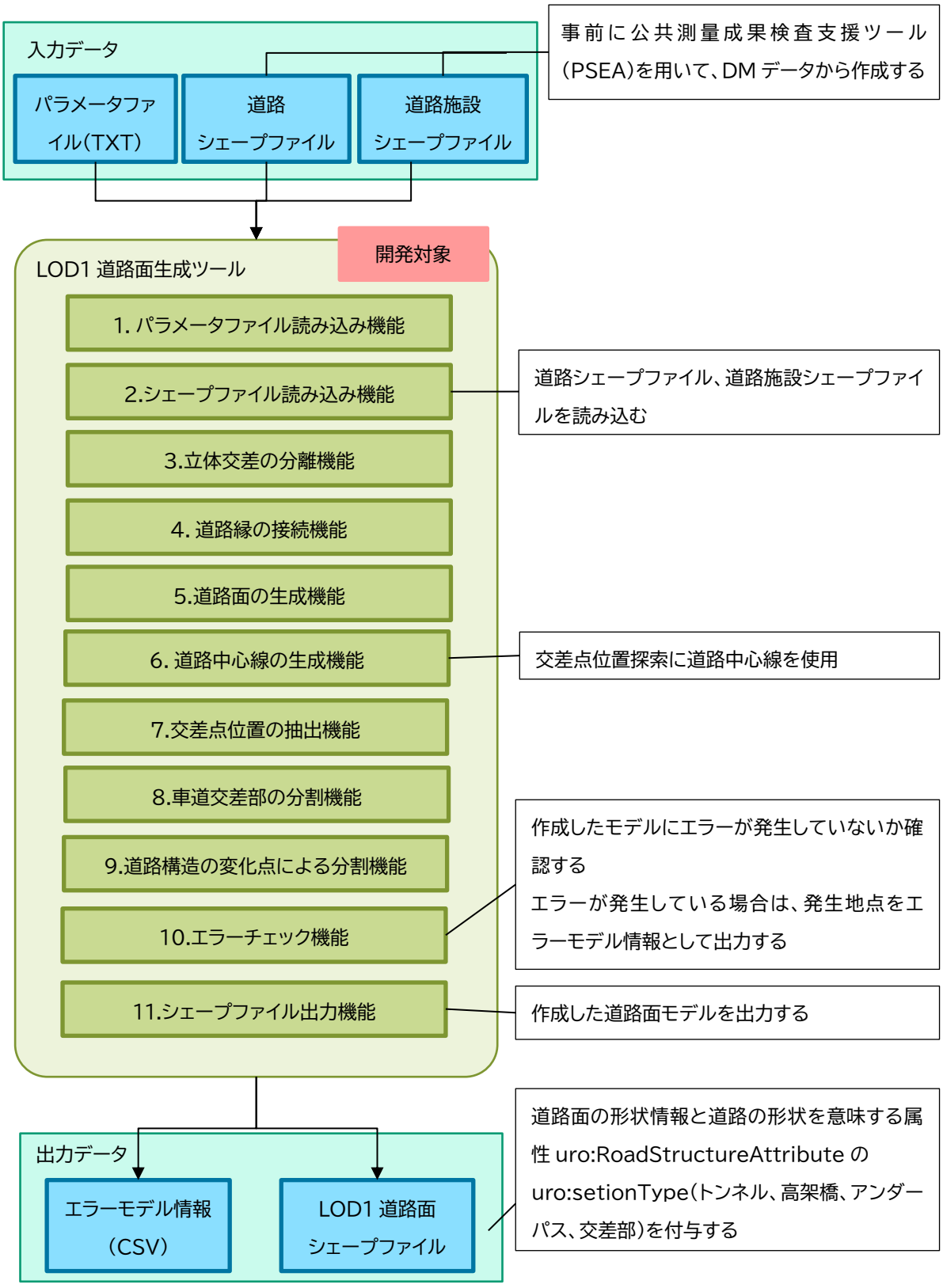


図 3-10　LOD1 道路面生成ツールの機能概要

3.3.10.2.入力ファイル

入力ファイルを表 3-9 に示す。

表 3-9　入力ファイル

項目	形式	内容
道路	シェープファイル (*.shp、*.shx、 *.dbf、*.prj)	公共測量成果検査支援ツール(PSEA)を用いて道路の DM データ(地図情報レベル 2500)をシェープファイルに変換したもの。 座標系は、平面直角座標系とする。
道路施設	シェープファイル (*.shp、*.shx、 *.dbf、*.prj)	公共測量成果検査支援ツール(PSEA)を用いて道路施設の DM データ(地図情報レベル 2500)をシェープファイルに変換したもの。 座標系は、平面直角座標系とする。
パラメータファイル	TXT	本ツールのパラメータを指定するファイル。

(1) 道路シェープファイル

道路シェープファイルは、公共測量成果検査支援ツール(PSEA)を用いて道路の DM データ(地図情報レベル 2500)をシェープファイル(*.shp、*.shx、*.dbf、*.prj)に変換したものとする。道路の DM データには、下表に示す道路種別のデータが存在するが、本ツールでは道路縁(街区線)データ(コード:2101)を使用して道路モデルを作成する。

図 3-11 に公共測量成果検査支援ツールで道路の DM データを表示した例を示す。

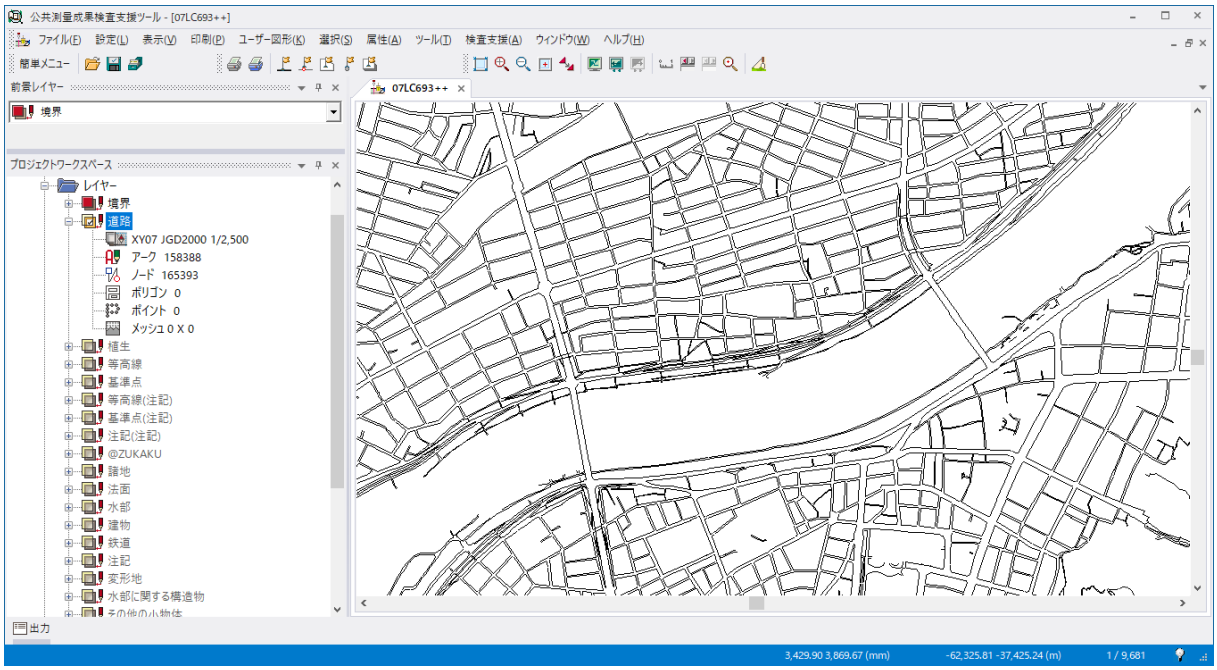


図 3-11　道路の DM データ

表 3-10 に DM データの道路種別を示す。これらの属性情報を本システムでは利用する。

表 3-10　DM データの道路種別

No.	名称	データ項目コード
1	未分類	2100
2	道路縁(街区線)	2101
3	軽車道	2102
4	徒歩道	2103
5	庭園路等	2106
6	トンネル内の道路	2107
7	建設中の道路	2109

(2)　道路施設シェープファイル

道路施設シェープファイルは、公共測量成果検査支援ツール(PSEA)を用いて道路施設の DM データ(地図情報レベル 2500)をシェープファイル(*. shp、*. shx、*. dbf、*. prj)に変換したものとする。道路施設の DM データには、下表に示す道路施設種別のデータが存在するが、本ツールでは道路橋(高架)データ(コード:2203)と、道路のトンネルデータ(コード:2219)を使用する。道路施設のシェープファイルは、点、線、面データごとにファイルが分割されているものとする。

図 3-12 に公共測量成果検査支援ツールで道路の DM データを表示した例を示す。

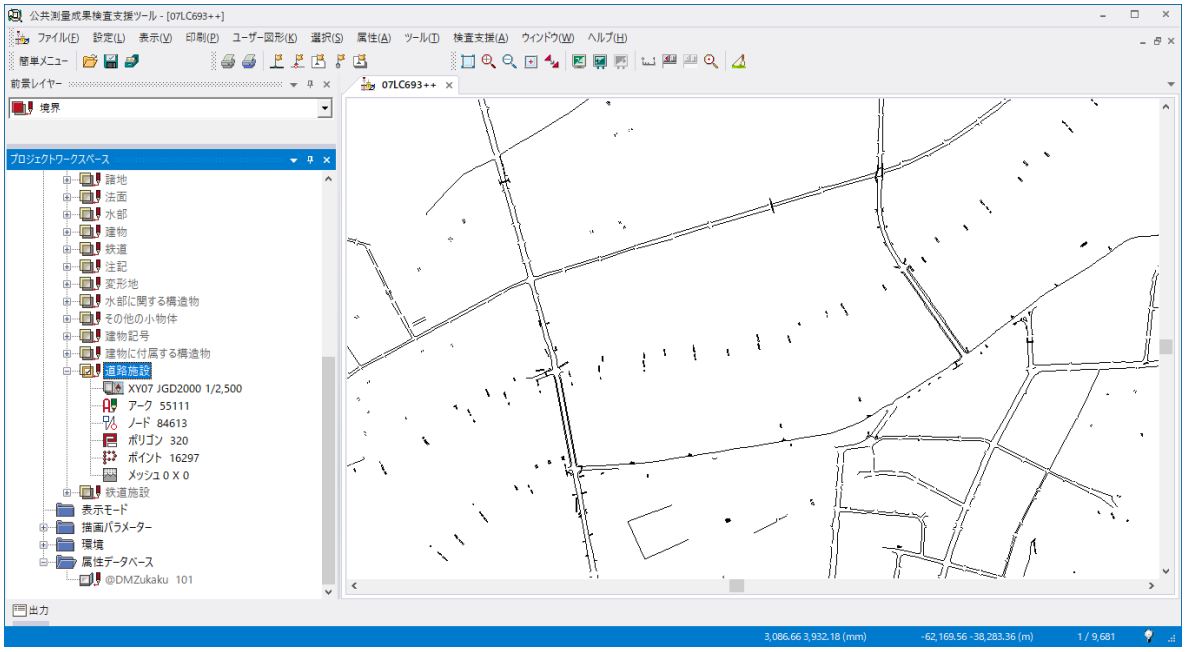


図 3-12　道路施設の DM データ

表 3-11 に DM データの道路種別を示す。これらの属性情報をツールでは利用する。

表 3-11　DM データの道路施設種別

No.	名称	データ項目コード
1	未分類	2200
2	道路橋(高架部)	2203
3	木橋	2204
4	徒橋	2205
5	桟道橋	2206
6	横断歩道橋	2211
7	地下横断歩道	2212
8	歩道	2213
9	石段	2214
10	地下街・地下鉄等出入口	2215
11	道路のトンネル	2219
12	バス停	2221
13	安全地帯	2222
14	分離帯	2226
15	駒止	2227
16	道路の雪覆い等	2228
17	側溝 U 字溝無蓋	2231
18	側溝 U 字溝有蓋	2232
19	側溝 L 字溝	2233
20	側溝地下部	2234
21	雨水桝	2235
22	並木桝	2236
23	並木	2238
24	植樹	2239
25	道路情報板	2241
26	道路標識 案内	2242
27	道路標識 警戒	2243
28	道路標識 規制	2244
29	信号灯	2246
30	信号灯 専用ポールのないもの	2247
31	交通量観測所	2251
32	スノーポール	2252
33	カーブミラー	2253
34	距離標(km)	2255

No.	名称	データ項目コード
35	距離標(m)	2256
36	電話ボックス	2261
37	郵便ポスト	2262
38	火災報知器	2263

(3) パラメータファイル

パラメータファイルの内容について表 3-12 に示す。なお、ファイルフォーマットはテキストファイルとし、文字コードは Shift_JIS とする。

表 3-12　入力パラメーター一覧

設定項目	内容
RoadSHPPath	道路シェープファイル(*.shp)パス。 記載されたシェープファイルパスと同階層に、同名の.shp、.shx、.dbf、.prj の拡張子のファイルが存在するものとする。
RoadFacilitiesPointSHPPath	道路施設(点データ)シェープファイルパス。 記載されたシェープファイルパスと同階層に、同名の.shp、.shx、.dbf、.prj の拡張子のファイルが存在するものとする。
RoadFacilitiesLineSHPPath	道路施設(線データ)シェープファイルパス。 記載されたシェープファイルパスと同階層に、同名の.shp、.shx、.dbf、.prj の拡張子のファイルが存在するものとする。
RoadFacilitiesPolygonSHPPath	道路施設(面データ)シェープファイルパス。 記載されたシェープファイルパスと同階層に、同名の.shp、.shx、.dbf、.prj の拡張子のファイルが存在するものとする。
JPZone	シェープファイルで使用している平面直角座標系の系番号。 (入力範囲:1 - 19)
OutputFolderPath	出力フォルダパス。
DMCode	道路/道路施設シェープファイルにおける DM コードの属性名。 シェープファイルから DM コードを取得する際の属性名として使用する。
GeometryType	道路/道路施設シェープファイルにおける図形区分の属性名。 シェープファイルから図形区分を取得する際の属性名として使用する。

MinArea	エラーチェックで面積が小さいポリゴンが存在しないか確認するためのしきい値として使用する。単位は m^2 。
MaxDistance	エラーチェックで車道交差部ポリゴンの中心と交差点間の距離が離れていないか確認するためのしきい値として使用する。単位は m。
inputRoadBeforeDivisionShpFilePath	指定のシェープファイルのエラーチェックをする際に使用する。交差点分割を行う前のポリゴンデータ。デバッグ用。
inputDividedShpFilePath	指定のシェープファイルのエラーチェックをする際に使用する。交差点分割を行った後のポリゴンデータ。デバッグ用。
inputIntersectionShpFilePath	指定のシェープファイルのエラーチェックをする際に使用する。交差点情報を持ったポイントデータ。デバッグ用。

［パラメータファイル サンプル］

```
[Setting]
RoadSHPPath= C:/work/data/DM_SHP/21_道路_L.shp
RoadFacilitiesPointSHPPath = C:/work/data/DM_SHP/22_道路施設_点.shp
RoadFacilitiesLineSHPPath = C:/work/data/DM_SHP/22_道路施設_線.shp
RoadFacilitiesPolygonSHPPath = C:/work/data/DM_SHP/22_道路施設_面.shp
JPZone=6
OutputFolderPath=C:/work/output

[DM_Attribute]
DMCode = 分類コーA
GeometryType = 図形区分

[ErrCheck]
MinArea = 10.0
MaxDistance = 10.0
inputRoadBeforeDivisionShpFilePath = C: /work/data/DM_SHP /BeforePolygonDivision.shp
inputDividedShpFilePath = C: /work/data/DM_SHP /AfterPolygonDivision.shp
inputIntersectionShpFilePath = C/work/data/DM_SHP /IntersectionPointList.shp
```

3.3.10.3.出力ファイル

出力ファイルを表 3-13 に示す。

表 3-13　出力ファイル

項目	形式	内容
LOD1 道路面	シェープファイル	LOD1 道路面情報を記載したシェープファイル。CityGML の道路形状を意味する属性 uro:RoadStructureAttribute の uro:sectionType(トンネル、高架橋、アンダーパス、交差部)に該当する属性情報を有する。
エラーモデル情報	CSV	エラーが発生しているモデルの位置情報とエラー内容を有する。

(1) シェープファイル

LOD1 道路面のシェープファイルは、パラメータファイルの OutputFolderPath で指定されたフォルダに出力する。シェープファイルには、LOD1 道路面の形状情報と属性名 sectType に CityGML の uro:RoadStructureAttribute の uro:sectionType に該当する属性情報を記載する。

本システムでは、高架橋は sectType = 2、車道交差部は sectType = 4、トンネルは sectType = 6 とする。(RoadStructureAttribute_sectionType.xml の全容は以下のとおりである。)

[RoadStructureAttribute_sectionType.xml サンプル]

```
<?xml version="1.0" encoding="UTF-8"?>
<gml:Dictionary                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:gml="http://www.opengis.net/gml"
xsi:schemaLocation="http://www.opengis.net/gml
http://schemas.opengis.net/gml/3.1.1/profiles/SimpleDictionary/1.0.0/gmlSimpleDicti
onaryProfile.xsd" gml:id="cl_2dff6a6b-1c22-4760-a33b-3a08cd371f85">
<gml:name>RoadStructureAttribute_sectionType</gml:name>
<gml:dictionaryEntry>
  <gml:Definition gml:id="id1">
   <gml:description>土工区間</gml:description>
   <gml:name>1</gml:name>
  </gml:Definition>
 </gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:Definition gml:id="id2">
   <gml:description>高架橋</gml:description>
```

```
  </gml:Definition>
 </gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:Definition gml:id="id3">
   <gml:description>橋梁</gml:description>
   <gml:name>3</gml:name>
  </gml:Definition>
 </gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:Definition gml:id="id4">
   <gml:description>交差部</gml:description>
 <gml:name>4</gml:name>
  </gml:Definition>
 </gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:Definition gml:id="id5">
   <gml:description>アンダーパス</gml:description>
   <gml:name>5</gml:name>
  </gml:Definition>
 </gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:Definition gml:id="id6">
   <gml:description>トンネル</gml:description>
   <gml:name>6</gml:name>
  </gml:Definition>
 </gml:dictionaryEntry>
</gml:Dictionary>
```

(2) エラーモデル情報

エラーモデル情報は、パラメータファイルの OutputFolderPath で指定されたフォルダに ErrorInfo.csv として出力する。エラーモデル情報のファイルフォーマットは CSV ファイルとし、文字コードは UTF-8 とする。エラーモデル情報の内容を表 3-14 に示す。

表 3-14　エラー情報

項目	内容
ERROR	発生しているエラー内容
X	エラーが発生しているモデルの X 座標(座標系は平面直角座標系)
Y	エラーが発生しているモデルの Y 座標(座標系は平面直角座標系)

[ErrorInfo.csv サンプル]

```
ERROR,X,Y
TOPOLOGICAL_ERR, -35305.38, -64644.52
```

エラー内容の表記を表 3-15 に示す。

表 3-15　エラー内容の表記

項目	表記	内容
1	MISSING_MODEL_ERR	モデル面の欠落
2	TOPOLOGICAL_INVAILD_ERR	トポロジー不正（不正ポリゴン）
	TOPOLOGICAL_SHORTAGE_POINT_ERR	トポロジー不正（頂点不足）
	TOPOLOGICAL_DUPLICATION_POINT_ERR	トポロジー不正（頂点重複）
3	ANGLE_ERR	モデル面の不正内角
4	INTERSECTION_MISMATCH_ERR	車道交差部と交差点の不一致 （１つのポリゴンに交差点が含まれない）
	INTERSECTION_SAME_POINT_ERR	車道交差部と交差点の不一致 （1 つのポリゴンに重畳した複数の交差点）
	INTERSECTION_DIFFERENT_POINT_ERR	車道交差部と交差点の不一致 （1 つのポリゴンに別の複数の交差点）
5	EXCESS_ERR	道路ポリゴンのはみ出し
6	SUPERIMPOSE_ERR	車道交差部の重畳
	WITHIN_ERR	車道交差部の包含
7	ROAD_DIVISION_ERR	車道交差部の道路分割線数の不正
8	MINUSCULE_POLYGON_ERR	極小ポリゴン
9	INTERSECTION_DISTANCE_ERR	車道交差部ポリゴン中心と交差点間距離

3.3.10.4.機能詳細

各機能について記載する。

(1) パラメータファイル読込機能

ツールの入力データパス、出力フォルダパス及びシェープファイルの平面直角座標系の系番号などのパラメータを記載したパラメータファイルを読み込む。

(2) シェープファイル読込機能

シェープファイルを読み込み、道路縁と道路施設のデータを取得する。道路のシェープファイルからは、道路縁の形状情報と形状処理に利用する属性を読み込む。道路施設のシェープファイルからは、交差部の参考情報として利用する道路施設情報(道路橋、トンネル)を読み込む。

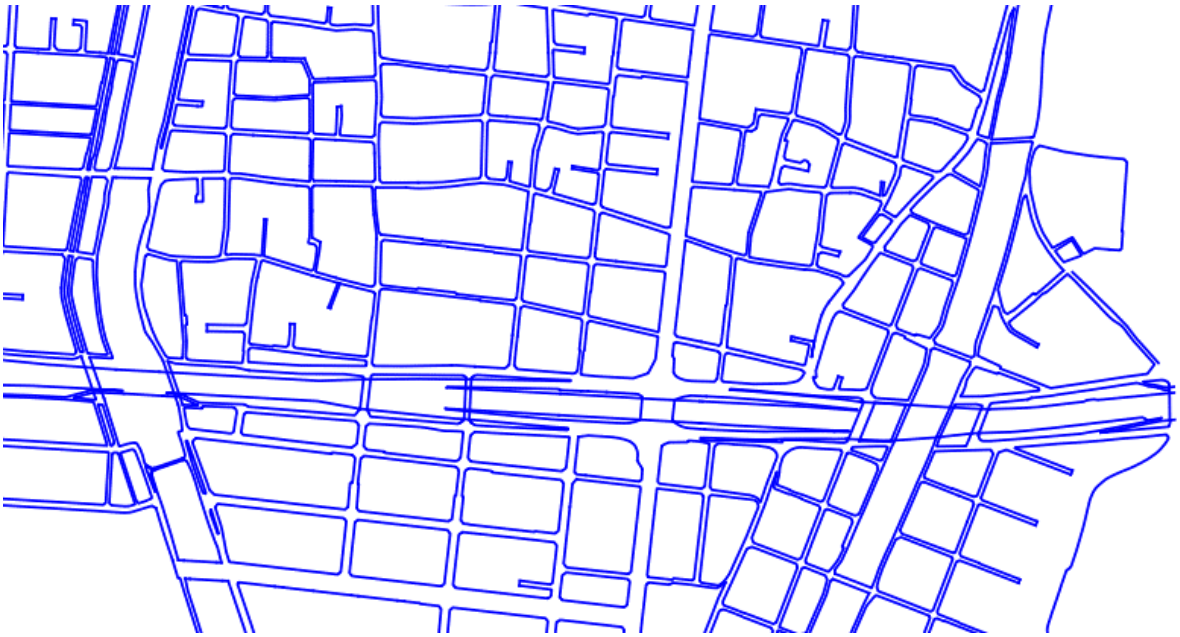


図 3-13　道路縁の形状情報

DM データの道路情報は、「表　3-10　DM データの道路種別」のとおり 7 種類に分類される。3D 都市モデル標準製品仕様では、道路モデルとは別に徒歩道モデルを定義しており、この徒歩道は「公共測量標準図式における徒歩道及び庭園路(ただし、庭園路のうち、自動車ターミナル内の道路は、広場として取得するため、徒歩道には含まない。)」を指す。そのため、DM データの道路モデルの原典データからは除外する。

DM データの道路情報において、単線として表現されている道路は道路の幅員情報が存在しないと道路面を生成することができないため、本システムではモデル化対象外とする。

図 3-14　道路縁の単線

(3)　立体交差の分離機能

立体交差が発生している道路縁を階層ごとに分離する。

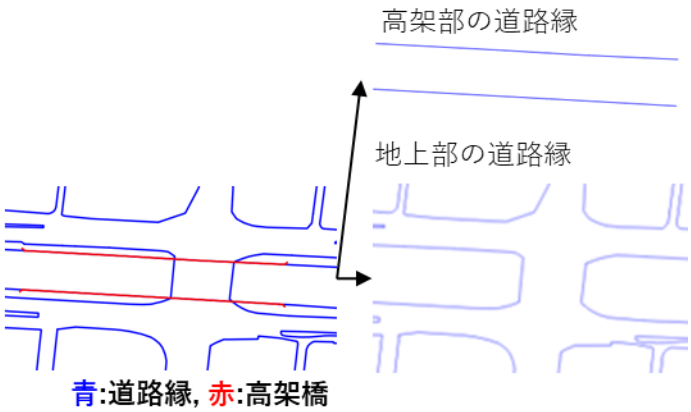


図 3-15　立体交差の分離

入力データは、2 次元平面上の線データで高さ情報がない。そのため、実際には立体交差している道路が DM データ上では平面交差する。DM データでは、図 3-16 の赤枠 1、2 のように上下道の道路縁の交差地点に、上下道それぞれの道路縁の頂点が存在する場合がある。このような場合、街区ポリゴンの作成時に上下道で重畳している頂点を誤って結合してしまい不正な街区ポリゴンを作成する可能性がある。交差地点において、実際の街区にはない経路に進む可能性があるため、正しい検出例を図 3-17、誤検出例を図 3-18 に示す。

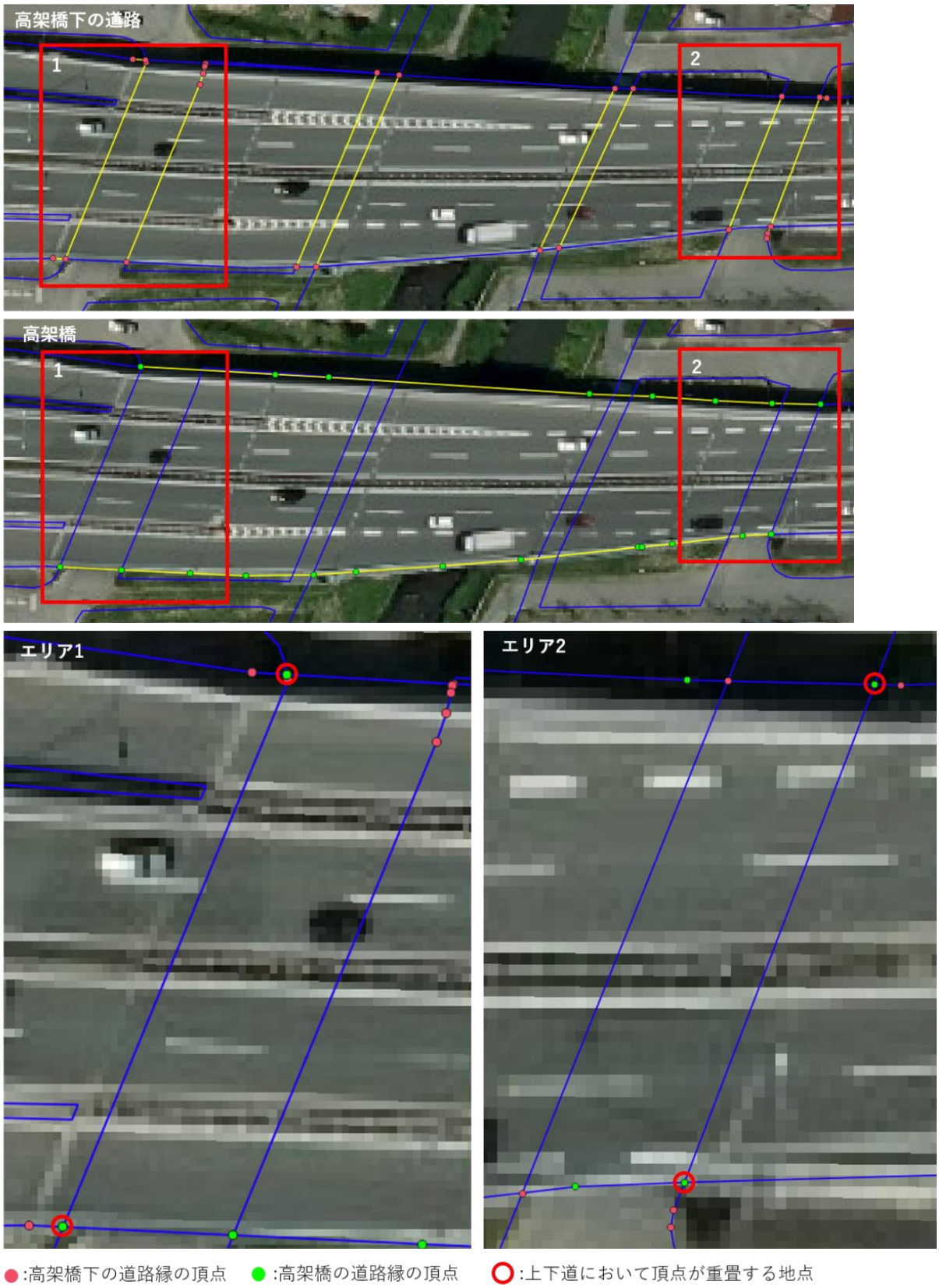

高架橋下の道路
1
2
高架橋
1
2
エリア1
エリア2
:高架橋下の道路縁の頂点
:高架橋の道路縁の頂点
:上下道において頂点が重畳する地点

図 3-16　立体交差

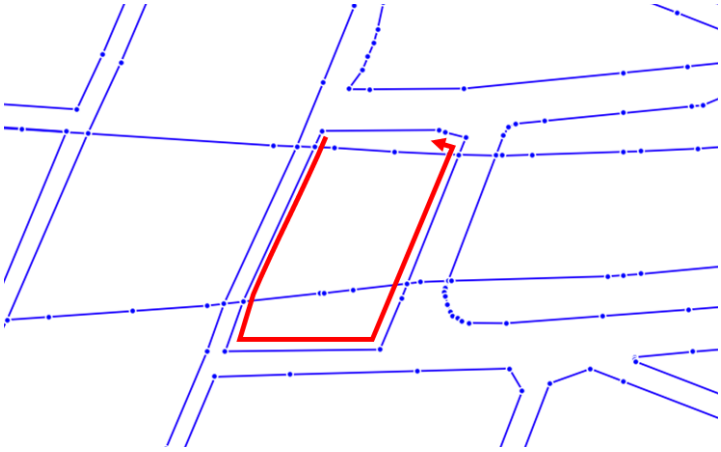


図 3-17　正しい経路検出例

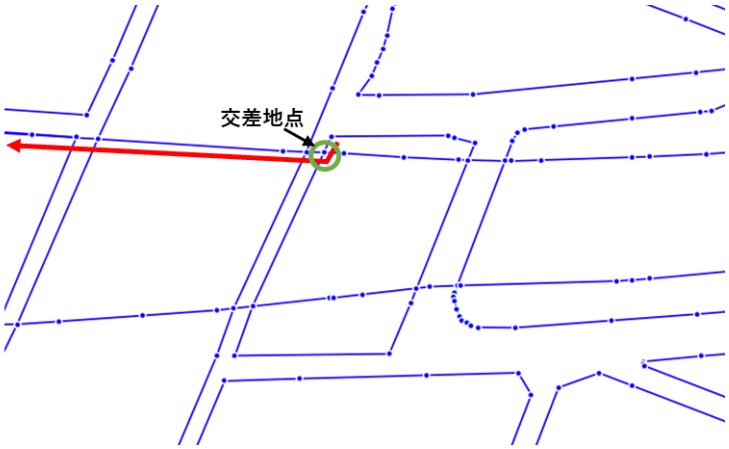


図 3-18　経路の誤検出例

立体交差の分離は、DM データの道路施設情報である道路橋(高架部)と道路のトンネル情報を使用して行う。

道路橋(高架部)は、図 3-19 の赤線で示す道路橋部分の線情報を保持しており、道路橋(高架部)と交差する道路縁が存在すれば立体交差が発生していると考えることが可能である。

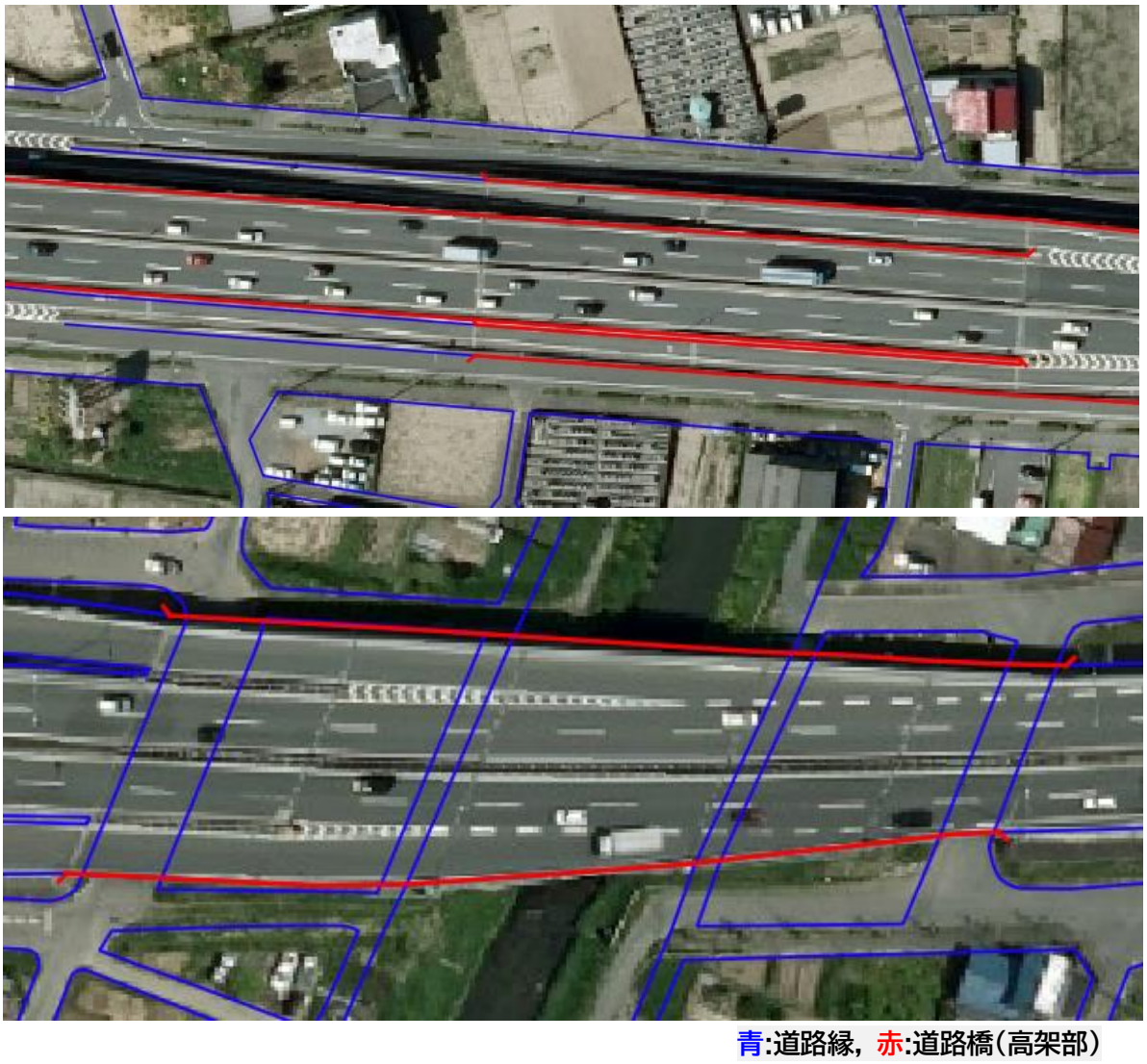


図 3-19　道路橋(高架部)

道路のトンネルは、図 3-20 に示すとおり、点、線、面形状の 3 種類のデータがある。3 種類のデータは共に、トンネルの坑口の地点を示す。

道路橋(高架部)同様、線、面形状の道路のトンネルデータと交差する道路縁が存在すれば立体交差が発生していると考えることが可能である。

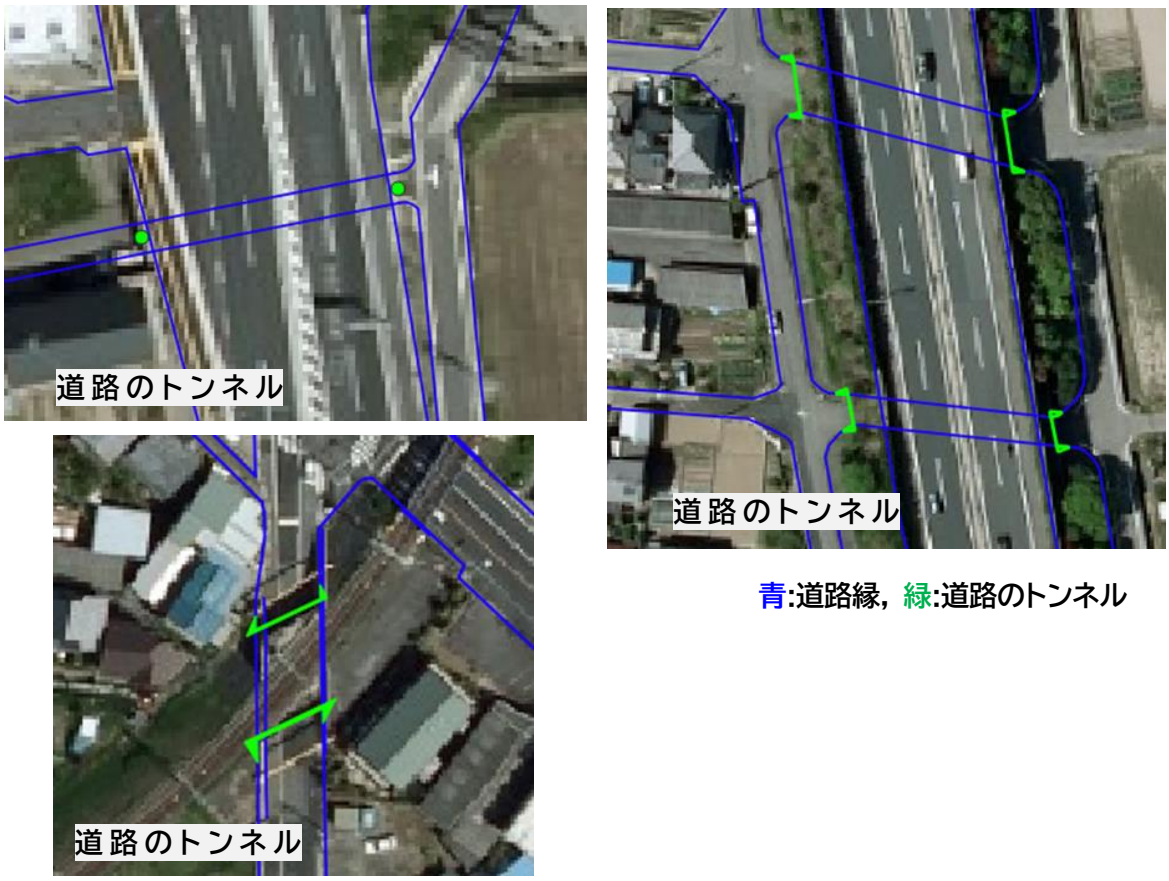


図 3-20　道路のトンネル

点形状の道路のトンネルデータに関しては、図 3-21 に示すように坑口に相当する補助線(道路のトンネルの点を通り、トンネルに該当する道路に直行する線)を作成することで、道路のトンネル(線)と同様に補助線と交差する道路縁の存在確認を行うことで立体交差の有無を判断する。

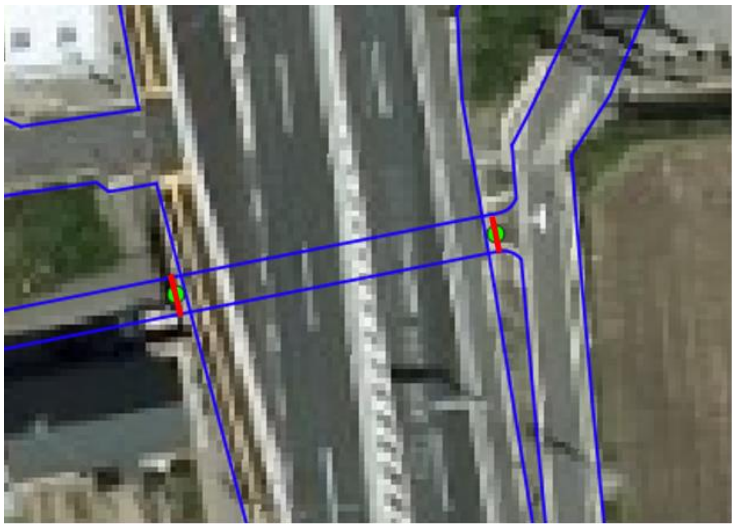
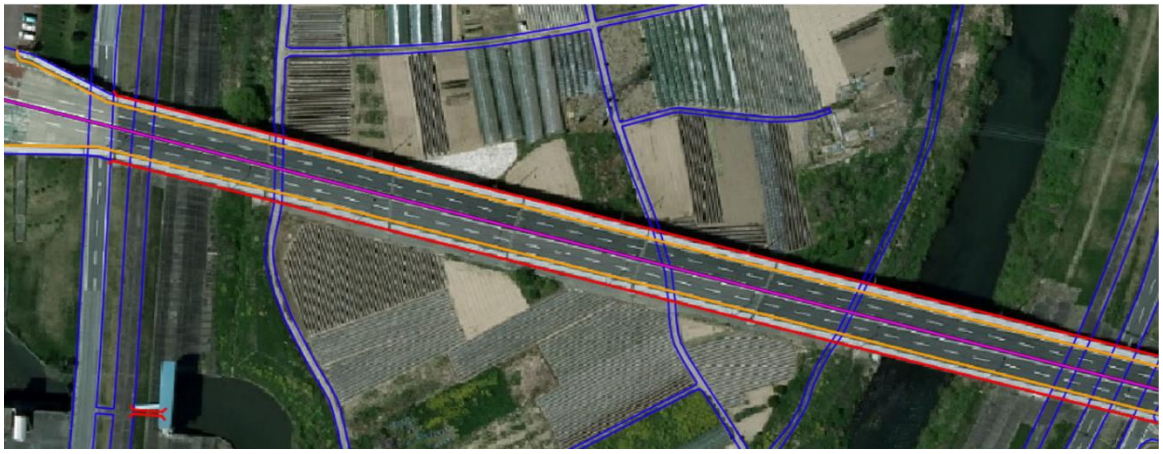


図 3-21　道路のトンネル(点)データの加工

1 つの橋を表現するためには最低 2 本の道路橋(高架部)のデータが必要である。道路のトンネルも同様に、1 つのトンネルを表現するには坑口データが 2 つ必要になる。DM データの構造確認で使用した岐阜市の DM データでは、道路橋(高架部)は道路縁ごとに、道路のトンネルは坑口ごとにデータが独立しており、道路橋やトンネルごとにデータがまとめられていない。道路橋の形状やトンネルの範囲を把握するために、道路橋やトンネルごとにデータのグルーピングを行う。

道路橋のグルーピングに関しては、道路の中心を挟んで道路橋の線データが存在しているため、注目道路橋の線分に垂直かつ道路の中心方向に探索を行うことでグルーピング対象となる道路縁を発見する。なお、道路の中心方向については、道路縁よりも内側に存在する道路施設(歩道、分離帯)の情報を利用して判断する。



青:道路縁，赤:道路橋(高架部)，橙:歩道，紫:分離帯

図 3-22　歩道・分離帯

(4)　道路縁の接続機能

道路縁を接続して道路縁のループを作成する。道路縁は不規則なポリラインで構成されているため、1 街区分の道路縁が 1 レコードに保存されていない場合もある。1 街区分のポリゴンを作成するために連続する道路縁を結合する。

図 3-23　1 レコード分の道路縁

道路縁をグラフ(ノードとエッジの集合)とみなして、深さ優先探索を行うと閉路を抽出可能である。また、データ境界などにより街区が閉路になっていない場合でも、深さ優先探索の開始点を次数が 1(連結しているエッジが 1 本)のノードにすると、街区線に相当する経路を取得可能である。グラフについて図 3-24、街区ポリゴンの抽出について図 3-25 に示す。

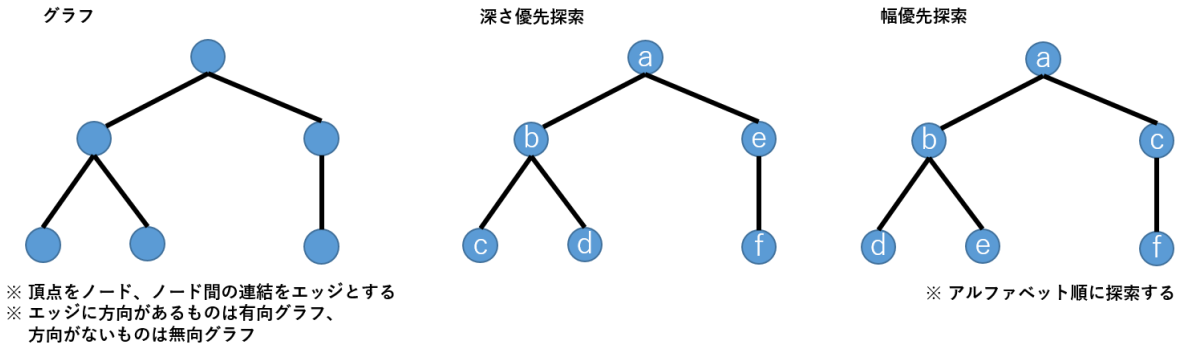


図 3-24　グラフ

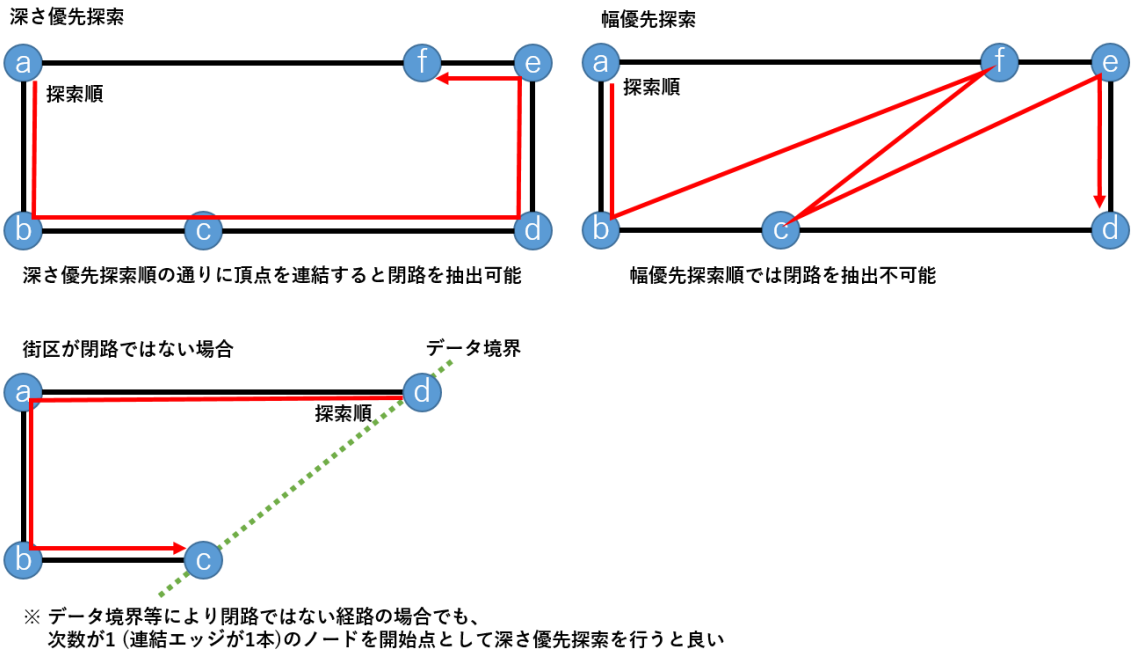


図 3-25　街区ポリゴンの抽出

(5) 道路面の生成機能

結合した道路縁を利用して、道路面を作成する（図 3-26）。道路面は、入力データ範囲（入力道路縁の外接矩形範囲）ポリゴンから、街区ポリゴンを引いた差分領域に相当する。

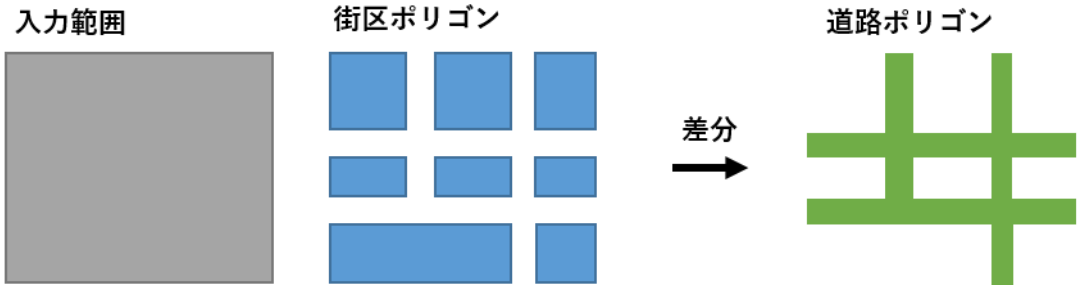


図 3-26　道路ポリゴンの作成

上記の処理によって作成された道路面の例を図 3-27 に示す。

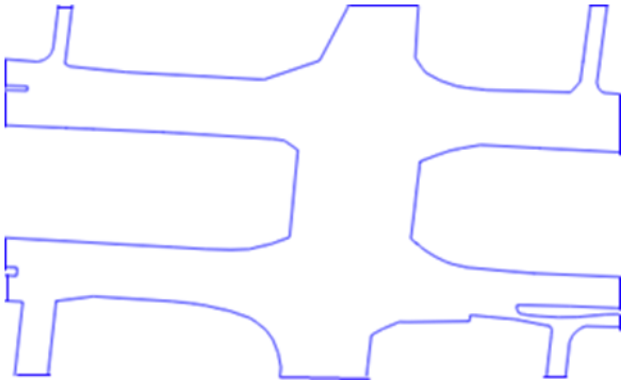


図 3-27　道路面の作成例

(6) 道路中心線の生成機能

道路中心線の交点を利用することで図 3-28 に示すように道路の交差地点を発見することが可能であるが、DM データには道路中心線が存在しないため、道路中心線を作成する。

道路縁から道路中心線を取得する方法として、国土地理院公開のベクトルタイル(https://github.com/gsi-cyberjapan/vector-tile-experiment)も検討したが、網羅性、データ時期の違いによる差異などを考慮し、道路縁と同一データから生成することとした。

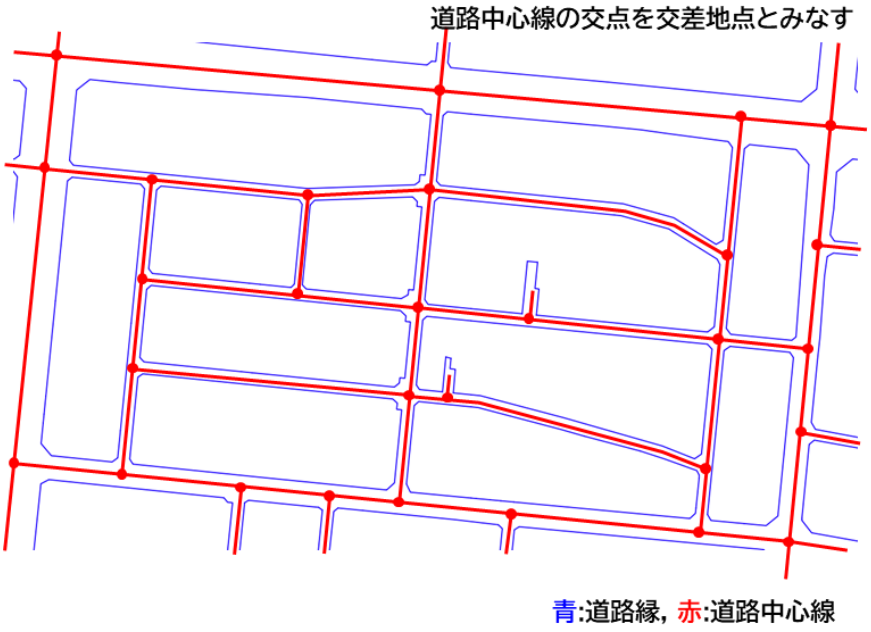
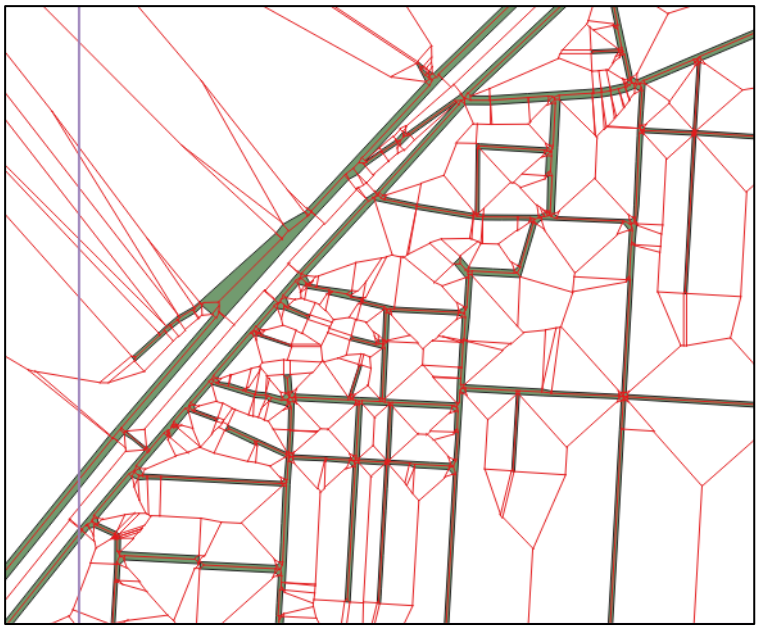


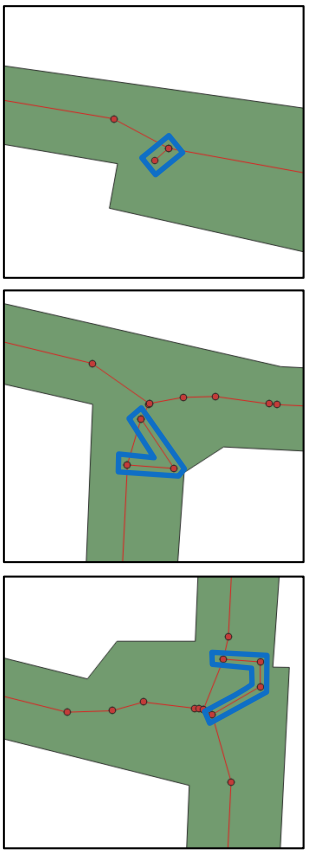
図 3-28　道路中心線の利用イメージ

道路縁を基にボロノイ分割を実施し、道路ポリゴン内に存在するボロノイ分割線（図 3-29-a）から道路ポリゴン内に存在する分割線を道路中心線として抽出する（図 3-29-b）。抽出した道路中心線には不要なエッジ（図 3-29-c）が存在するため、除去処理を行う。また、十字路などの交差点において、図 3-29 の d のように 1 つの車道交差部に道路中心線の交点が複数発生する。そのため、近傍に存在する道路中心線の交点をマージする処理を行う。

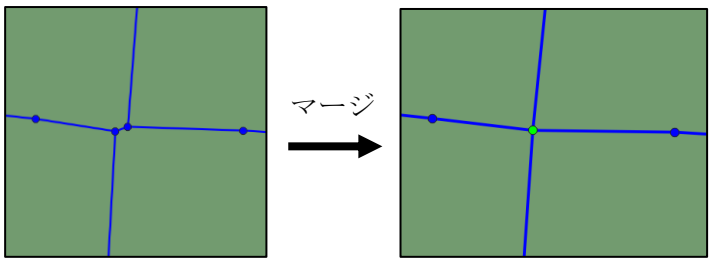


a. ボロノイ分割線

b. 道路ポリゴン内に存在する分割線のみを抽出



c. 不要なエッジ例



d. 交差点のマージ

図 3-29　道路中心線作成処理

(7) 交差点位置の抽出機能

道路中心線から交差点位置を推定する(図 3-30)。

道路中心線が交差する位置、接する位置を取得する。作成した道路中心線は端部が分岐する場合がある。この分岐を交差と判断しないようにするための処理も実装した。

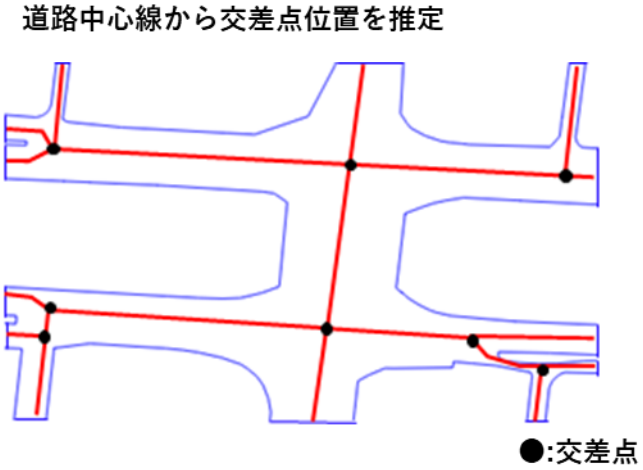


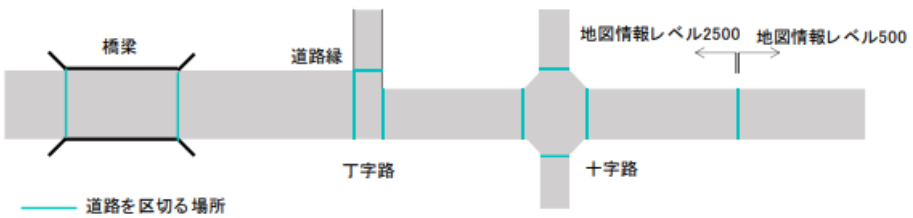
図 3-30　交差位置を推定

(8) 車道交差部の分割機能

LOD1 道路面モデルは「3D 都市モデル標準製品仕様書」に従い、以下の場所で区切る必要がある。

- 車道交差部(十字路、丁字路、その他 2 つ以上の道路が交わる部分)
- 道路構造(トンネル、橋梁等)の変化点
- 位置正確度や取得方法が変わる場所

表 3-16　LOD1 道路仕様[5]

	LOD1
取得例	
説明	道路縁により囲まれた範囲を面として取得し、以下の場所で区切る。 ● 車道交差部（十字路、丁字路、その他二つ以上の道路が交わる部分）で区切る。 ● 道路構造（トンネル、橋梁）が変化する場所 ● 位置正確度や取得方法が変わる場所 高さは 0 とする。

[5] 国土交通省都市局 Project PLATEAU,
3D 都市モデル標準製品仕様書 ver.3.5 (2024.3) p.159. 表 4-17 引用

● 交差部（四差路、多差路及び三差路）での道路の区切り方

交差部での道路の区切り方は、以下に定義する優先順位で区切る。ただし、区切った交差部同士が重なる場合は、一つの交差部とする。なお、本手順書では道路が交差、分岐又は合流する場合において、ある道路の道路縁と他の道路縁とが接する点（道路角）を「接点」と呼ぶ。

① 隅切りのない四差路及び多差路

一つの道路において、道路縁の両側に接点が存在する場合は、接点を結んで道路を区切る。

② 隅切りのある四差路及び多差路

一つの道路において、道路縁の両側に隅切りが存在する場合は、隅切りの頂点を接点とし、接点を結んだ範囲を交差部とする。また、どちらか一方に隅切りが存在しない場合は、隅切りの頂点及び接点を結んだ範囲を交差部とする。

③ 三差路

道路縁の片側にしか接点が存在しない道路が一つでもある場合（三差路）は、隅切りがある場合は隅切りの頂点を接点とし、全ての接点から垂線を引き区切る。隅切りがある場合や区切りが複数ある場合は、より外側の区切りを採用する。

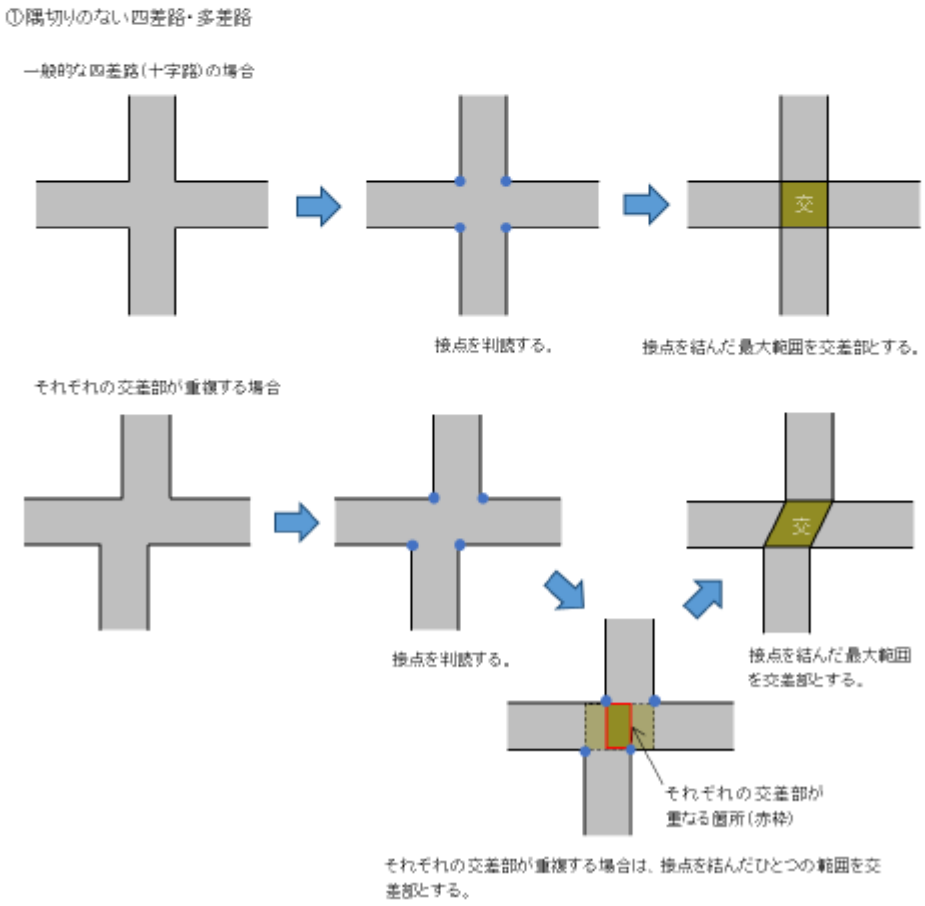


図 3-31　車道交差部での道路の区切り方（隅切りのない四差路）[6]

[6] 国土交通省都市局 Project PLATEAU,
3D 都市モデル標準作業手順書 ver.3.5（2024.3）pp.D-7. 引用

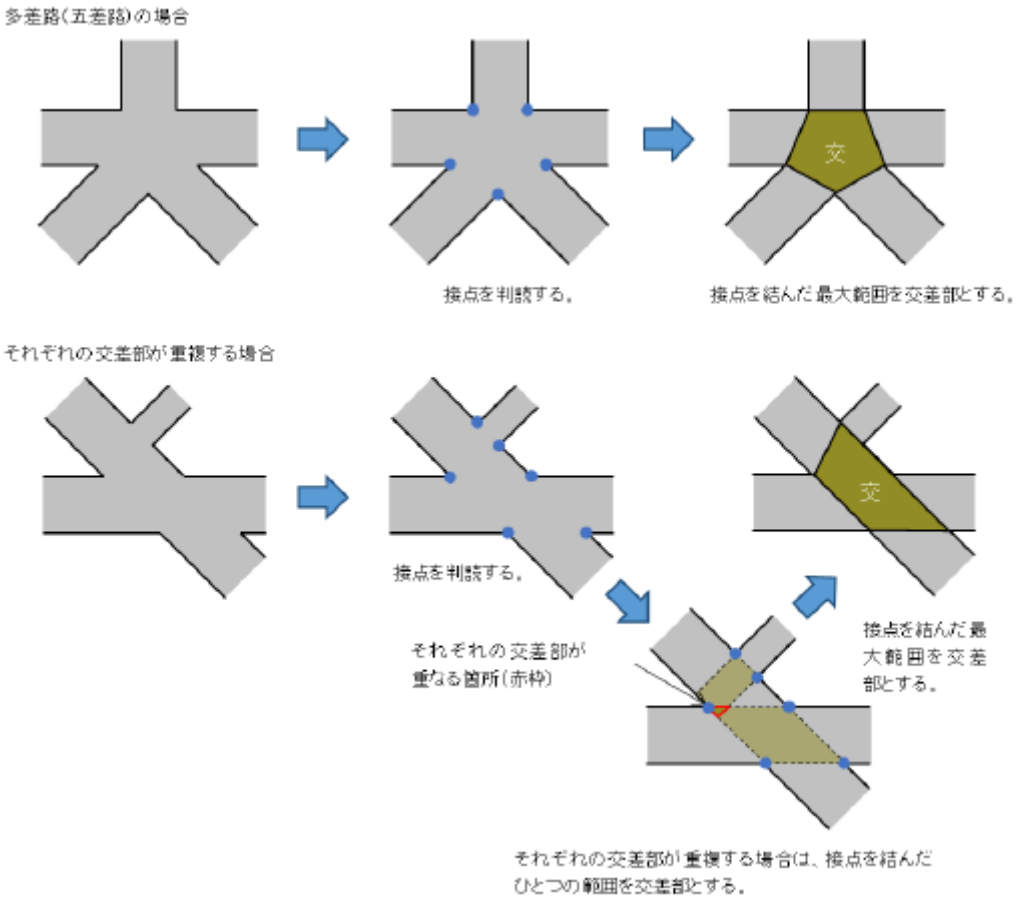


図 3-32　車道交差部での道路の区切り方（多差路）[7]

[7] 国土交通省都市局 Project PLATEAU,
3D 都市モデル標準作業手順書 ver.3.5 (2024.3) pp.D-8. 引用

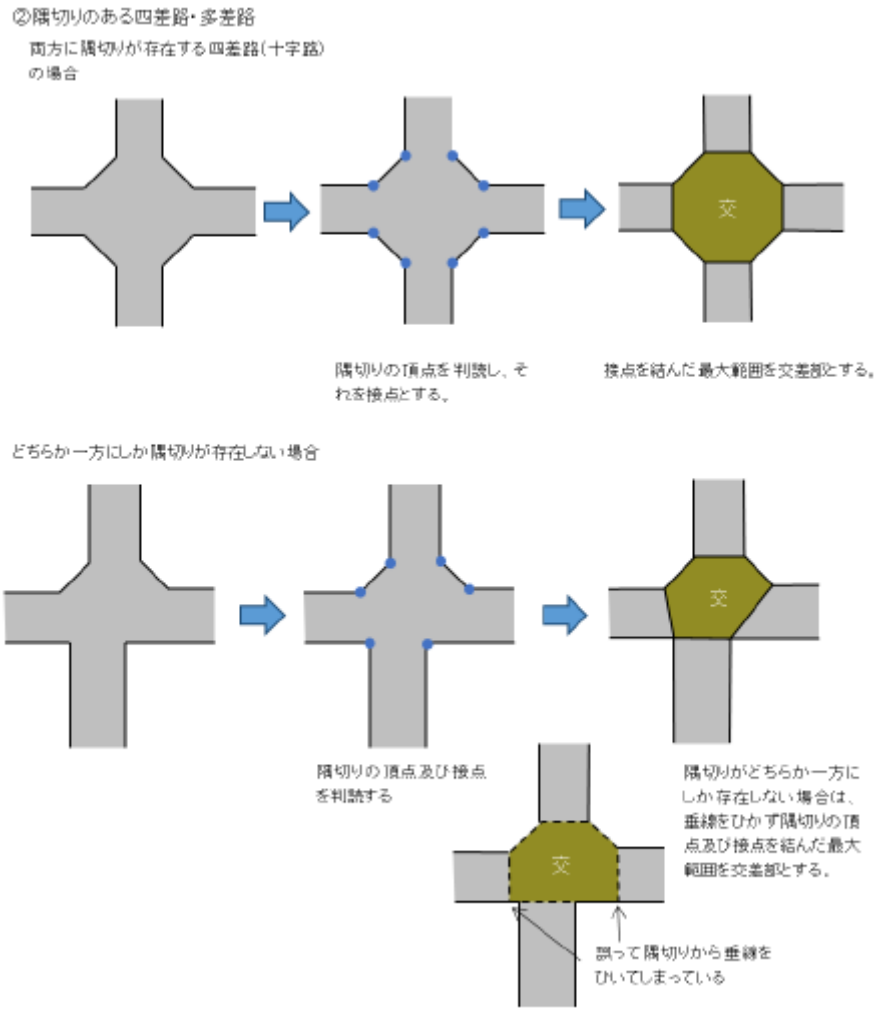


図 3-33　車道交差部での道路の区切り方（隅切りのある四差路）[8]

最初に道路ポリゴンを車道交差部のポリゴンとその他のポリゴンに分割する。分割例を図 3-34 に示す。

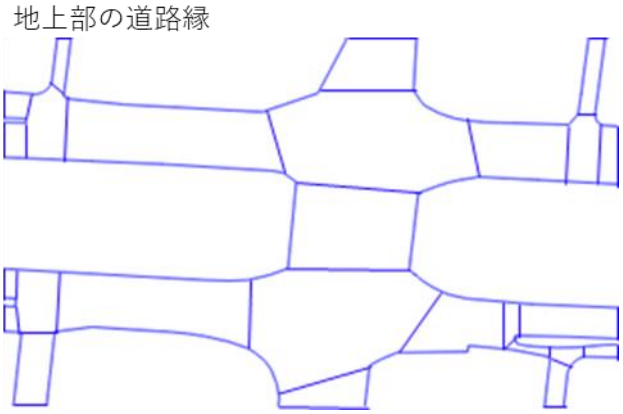


図 3-34　車道交差部の分割

[8] 国土交通省都市局 Project PLATEAU,
3D 都市モデル標準作業手順書 ver.3.5（2024.3）pp.D-9. 引用

① 交差点領域の作成（図 3-35）

1. 交差点座標を入力してボロノイ領域を作成する
2. 交差点から最近傍道路縁までの距離(概算道路幅の半分)を算出し、半径が概算道路幅の 2 倍の円形領域を作成する
3. ボロノイ領域と円形領域の重畳領域を取得する

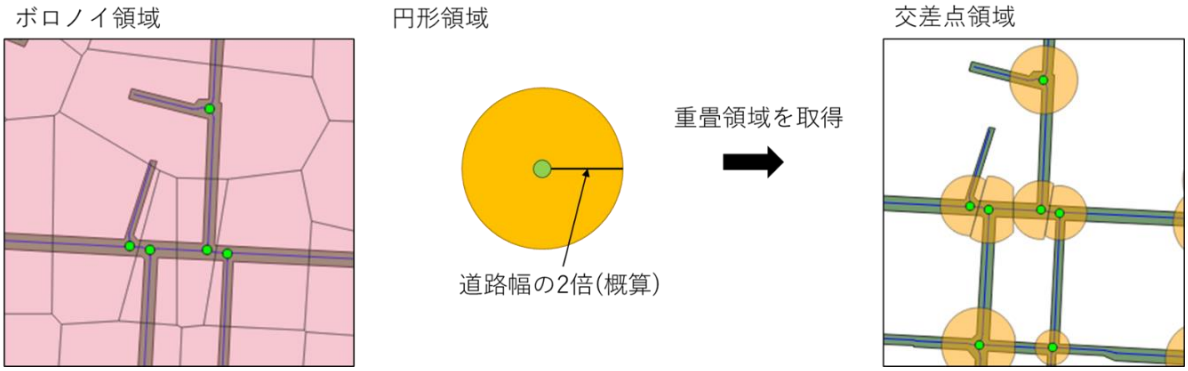


図 3-35　ボロノイ領域と円形領域、重畳領域（交差点領域）

② 車道交差部ポリゴンの作成（図 3-36）

交差点ごとに以下の処理を実施する。

1. 交差点近傍の道路縁を取得する
2. 道路縁の短縮と分割線の作成
3. 短縮後の道路縁と分割線を結合し、車道交差部ポリゴンを作成する

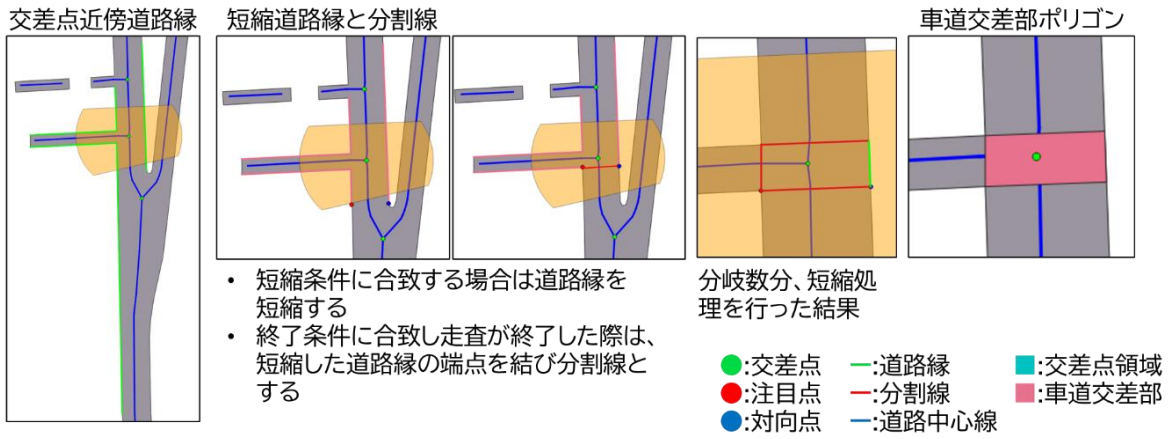


図 3-36　車道交差部の抽出

4. ポリゴン分割（図 3-37）を実施する

boost::geometry::difference()を使用して、道路ポリゴンから車道交差部や橋梁ポリゴンなどをそのまま減算するとスパイク形状のノイズ線が発生する。

道路ポリゴンの輪郭線において減算対象ポリゴンの頂点が重畳しない場合にはノイズ線が発生する傾向があるため、道路ポリゴンの輪郭線に減算対象ポリゴンの頂点を追加した。

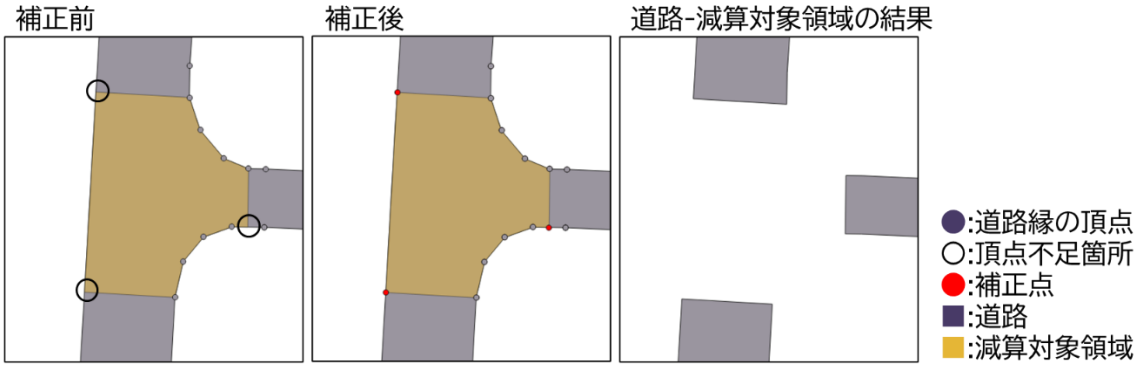


図 3-37　ポリゴン分割

(9)　道路構造の変化による分割機能

仕様に定められた道路構造の変化点による分割を行う。道路構造の変化点での区切り方を図 3-38 に示す。

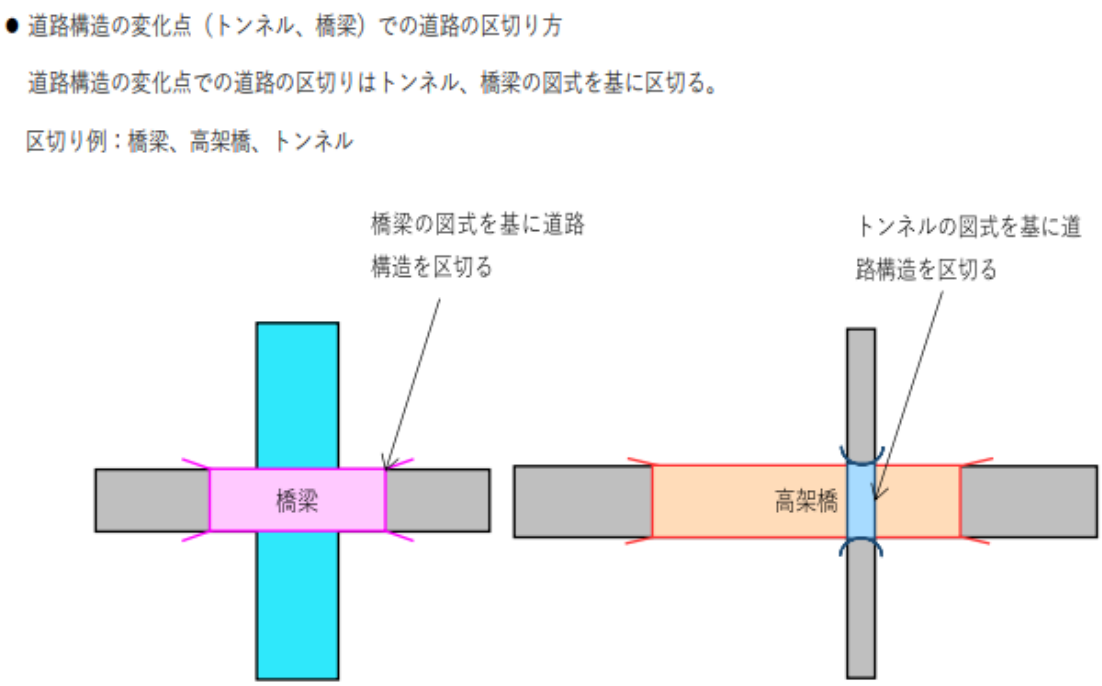


図 3-38　道路構造の変化点での道路の区切り方[9]

この処理は車道交差部のポリゴンの分割後に、高架橋やトンネルの情報を用いて車道交差部以外の道路ポリゴンに対して行う(図 3-39)。

[9] 国土交通省都市局 Project PLATEAU,
3D 都市モデル標準作業手順書 ver.3.5 (2024.3) p.D-11. 引用

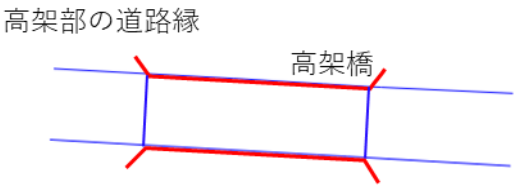


図 3-39　道路構造変化点による分離

図 3-40　道路橋、道路のトンネル例

車道交差部の分割が終了した状態では、車道交差部以外の道路ポリゴンはおおよそ長方形となるポリゴンとみなせる。（道路がカーブしている場合もあるため、必ず長方形となるわけではない。）分割後の道路ポリゴンを図 3-41 に示す。

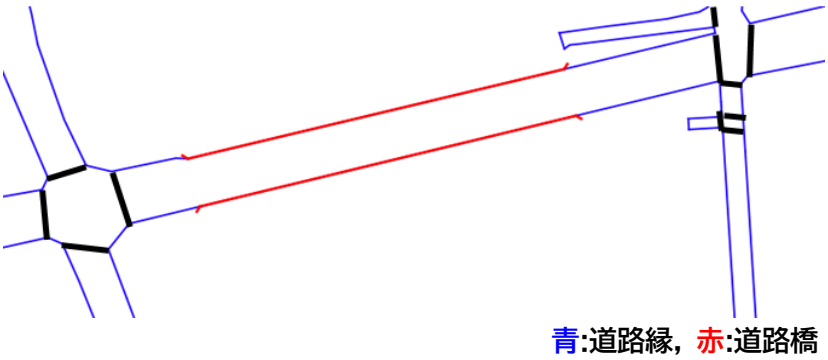


図 3-41　車道交差部分割後の道路ポリゴン

そのため、道路の延長方向に向かって道路の端から順に、トンネルや道路橋の端と交差する地点において道路ポリゴンを区切ることで、道路構造の変化によるポリゴンの分割を行う。分割対象の道路ポリゴンと道路橋が重畳しているか、道路のトンネルが交差しているかを確認して処理を行う。図 3-42 に区切り方を示す。

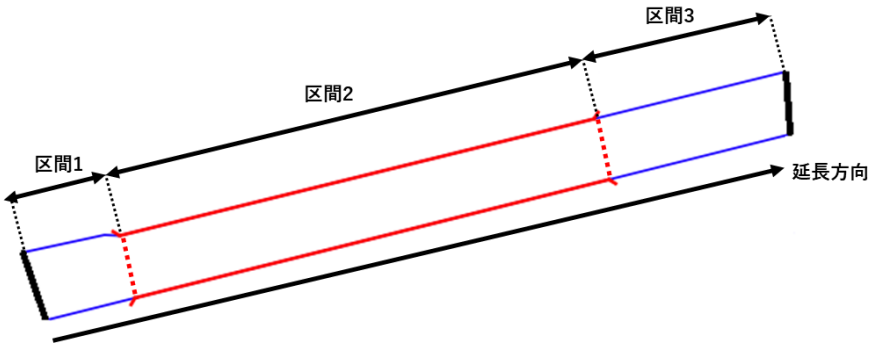


図 3-42　構造変化による道路の区切り方

①　処理方針

橋梁/トンネルの範囲に相当する道路縁から、橋梁/トンネル領域のポリゴンを作成する。道路ポリゴンから橋梁/トンネル領域のポリゴンの差分を取得することで、ポリゴンの分割を行う。図 3-43 に示す。

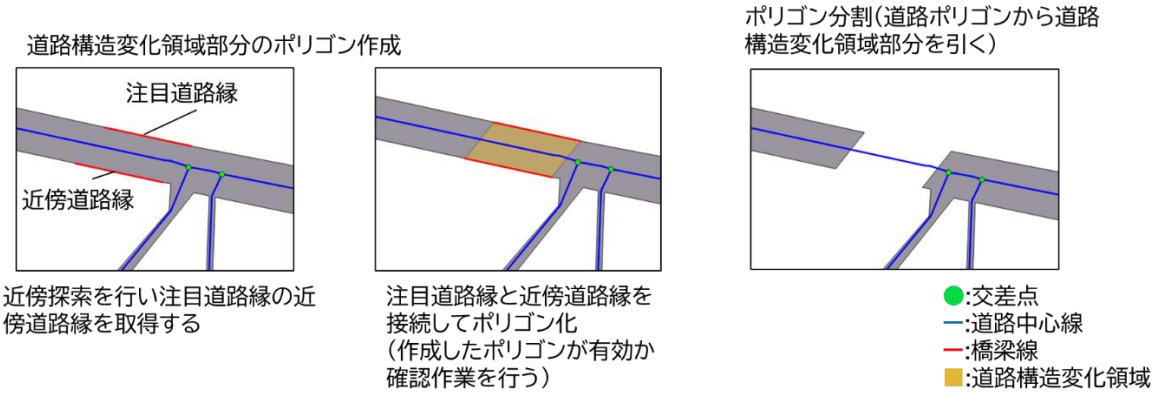


図 3-43　ポリゴン分割

②　処理手法

1. 橋梁/トンネルの範囲に相当する道路縁のポリラインの始点と終点データを使用して近傍探索用の R-Tree データを作成
2. 道路縁のポリラインごとに 3-8 の処理を繰り返す（図 3-44）
3. 注目ポリラインがポリゴン未作成の場合は、注目ポリラインの始点を基に近傍点探索を行う（近傍点探索には、K-Nearest Neighbor(K=4)を使用）
4. 5-8 の処理を近傍点ごとに繰り返す
5. 近傍点を含むポリラインを取得する
6. 注目ポリラインと近傍ポリラインを接続したポリゴンを 2 つ作成する
7. 作成したポリゴンが自己交差していないか、街区部分にはみ出ていないか確認する
8. 有効なポリゴンの場合は道路構造変化領域として採用する
   有効なポリゴンが複数存在する場合は、注目ポリラインの始点と近傍点が最短となるポリゴンを採用する

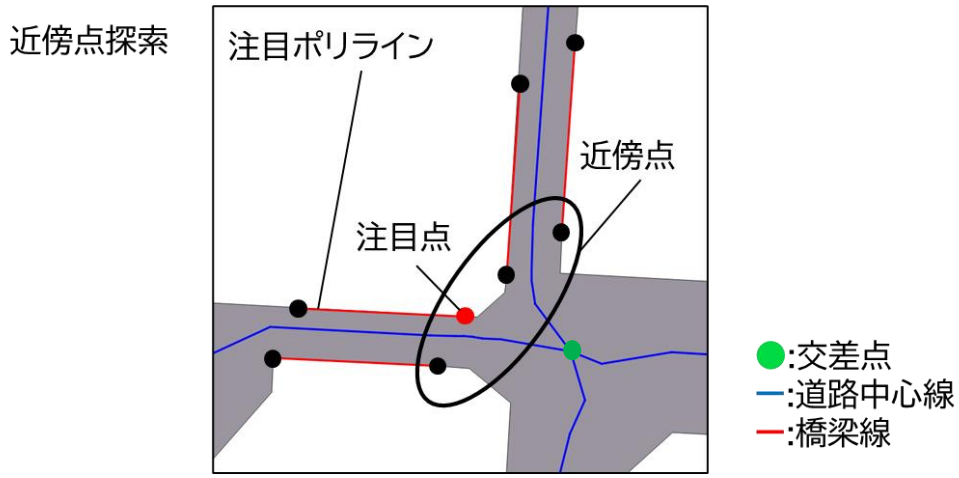


図 3-44　近傍点探索（2 の処理）

③　トンネルの処理

立体交差の分離と道路構造変化領域の分割では、トンネル部分に該当する道路縁が必要となるため、トンネルの坑口データを基にトンネル部の道路縁データを作成する（図 3-45）。まず前提として DM データの道路のトンネルは坑口位置を示すのみで、坑口ごとに独立したデータとなっている(トンネルの入り口と出口でデータがペアリングしていない状態)。そのため、坑口から道路縁に垂線を下して、道路縁との交点を取得し、その交点と他方の交点をペアにする処理を行う（図 3-46）。

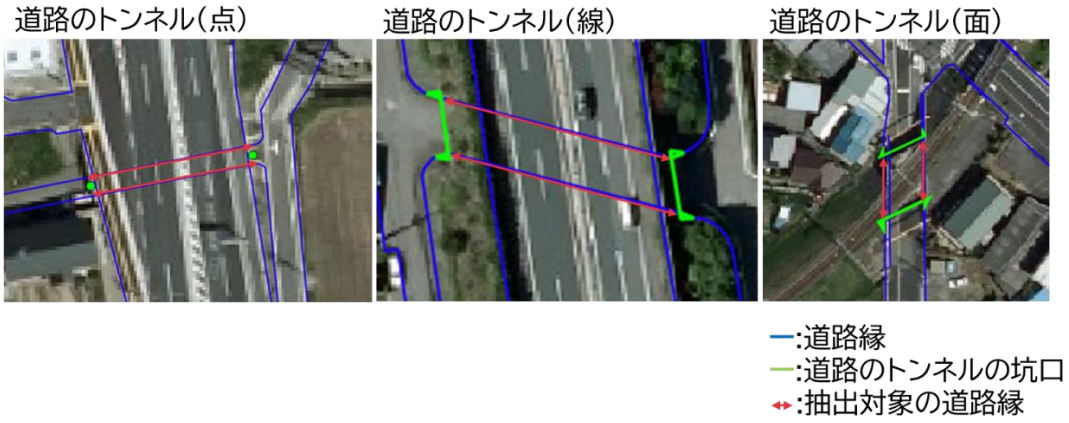


図 3-45　トンネルの図示タイプ

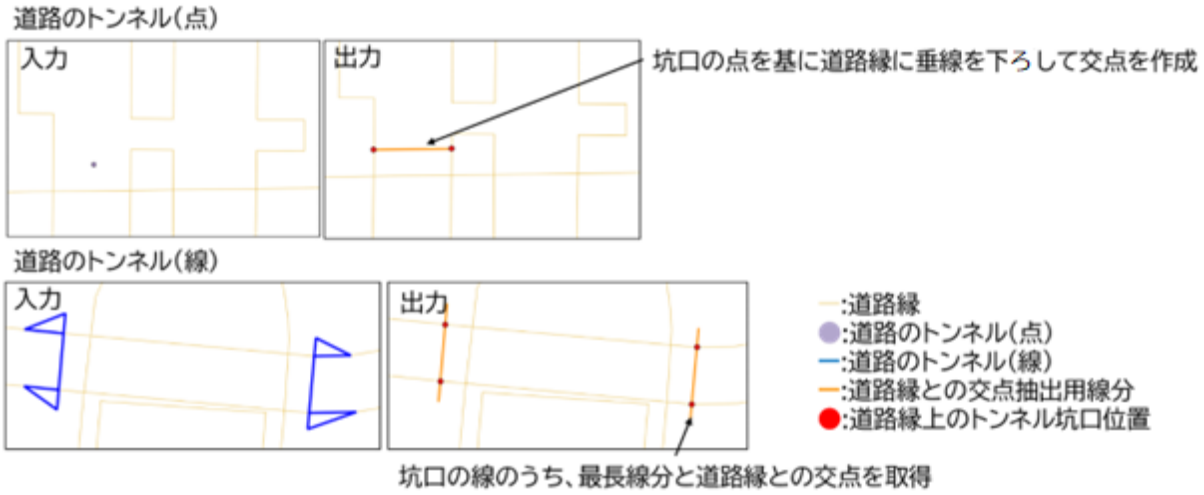


図 3-46　トンネルの取得

(10) エラーチェック機能

エラーチェック項目を

表 3-17 に示す。

表 3-17　エラーチェック項目

No.	エラーチェック項目	内容
1	モデル面の欠落確認	作成した道路面に欠落が存在していないか確認する
2	トポロジー不正確認	作成した道路面にトポロジー不正が存在していないか確認する
3	モデル面の内角確認	作成した道路面に鋭角(0-45 度)な内角が発生していないか確認する
4	車道交差部と交差点の一致確認	作成した車道交差部ポリゴンが、全ての交差点を内包しているか確認する
5	道路ポリゴンのはみ出し確認	作成した道路面が、分割前の道路ポリゴンからはみ出していないか確認する
6	車道交差部の重畳、包含確認	車道交差部ポリゴン同士で重畳、包含が発生していないか確認する
7	車道交差部の道路分割線数の確認	車道交差部の道路の分岐数と道路面の分割線数が同一であるか確認する
8	極小ポリゴンの確認	面積が小さいポリゴンが存在しないか確認する
9	車道交差部ポリゴン中心と交差点間距離の確認	車道交差部ポリゴンの中心と交差点間の距離が離れていないか確認する

1. モデル面の欠落確認

作成した道路モデルに欠落が存在していないか確認する。モデル面の欠落確認では、出力された道路モデルのほかに、入力範囲から街区ポリゴンを引いた道路ポリゴンを用いて検出する。モデルの欠落が発生している場合はエラーとし、位置情報を通知する。図 3-47 にモデル面の欠落検知を示す。

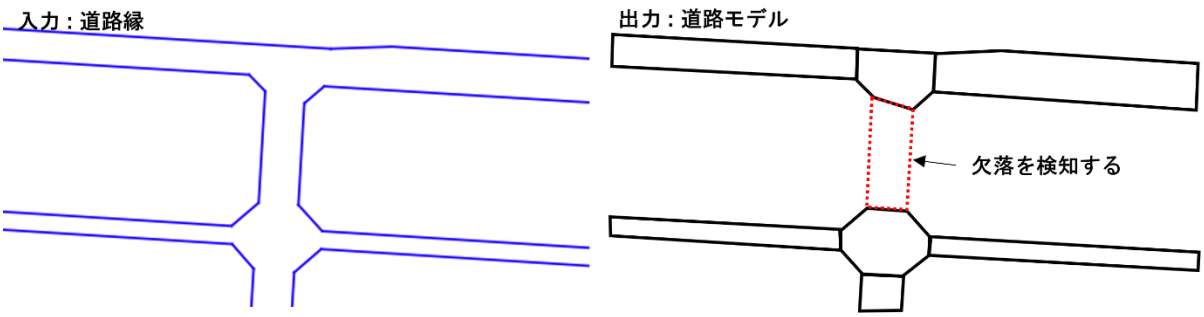


図 3-47　モデル面の欠落確認

2.　トポロジー不正確認

トポロジー不正確認では、モデル面に対して自己交差や自己接触などのトポロジー不正が発生していないか確認する。トポロジー不正が発生している場合はエラーとし、位置情報を通知する。

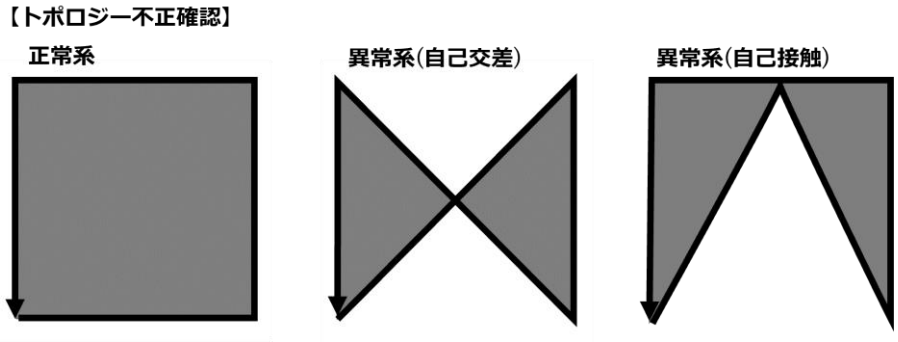


図 3-48　トポロジー不正確認

3.　モデル面の内角確認

作成した道路面に鋭角(0-45 度)な内角が発生していないか確認する。鋭角な内角が発生している場合はエラーとし、位置情報を通知する。

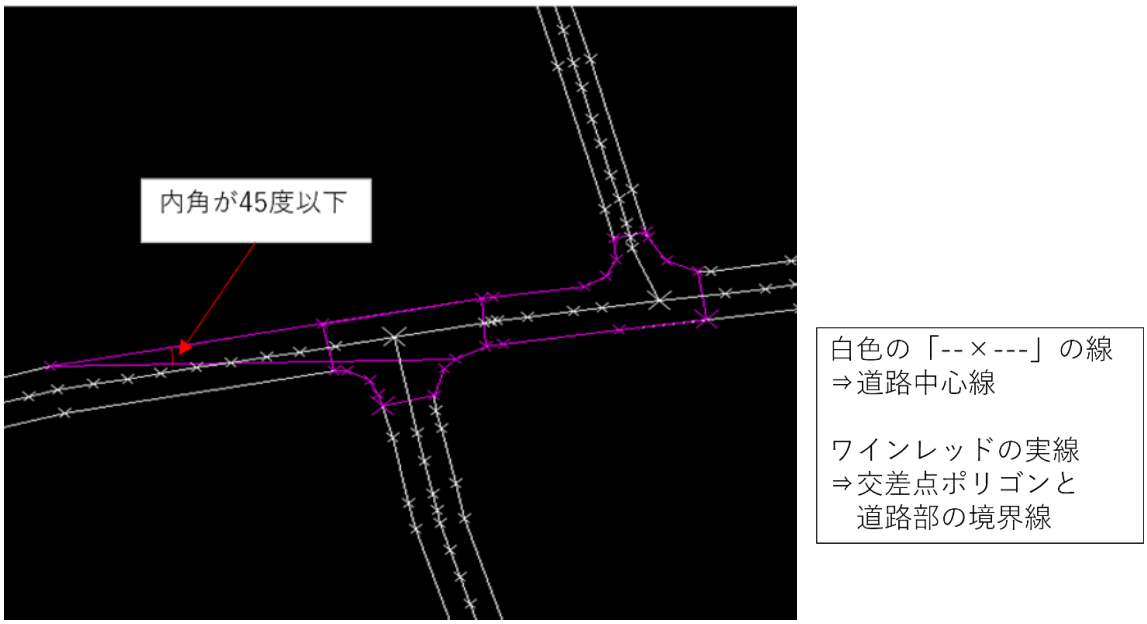


図 3-49　モデル面の内角確認

4.　車道交差部と交差点の一致確認

作成した車道交差部ポリゴンが、全ての交差点を内包しているか確認する。交差点を内包しない車道交差部ポリゴンが存在する場合及び車道交差部ポリゴンに内包されない交差点が存在する場合はエラーとし、位置情報を通知する。

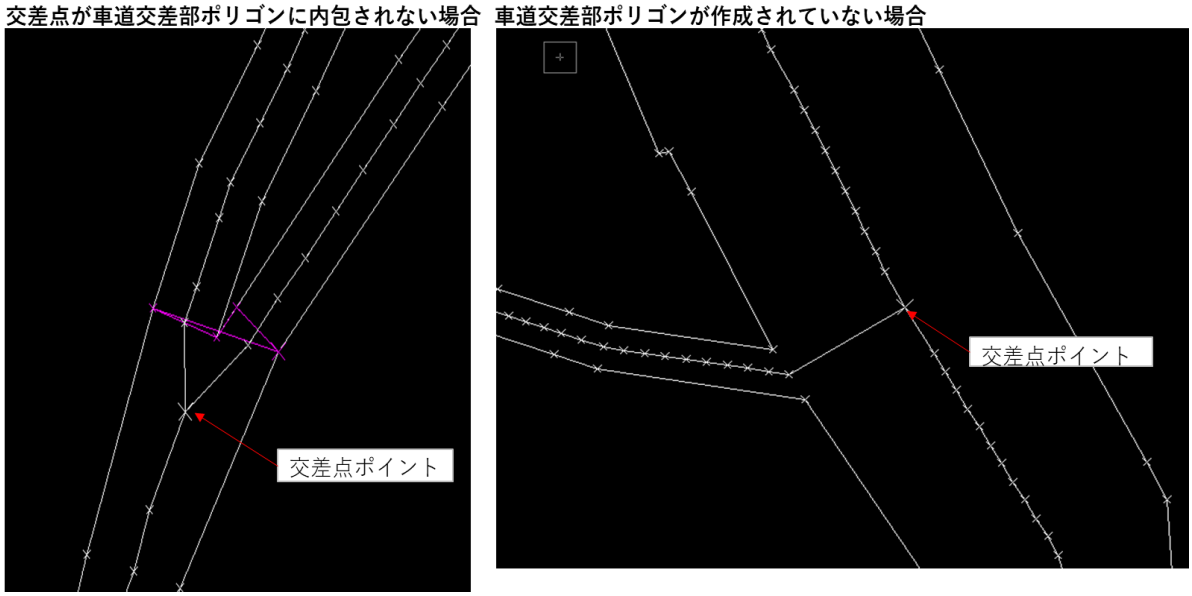


図 3-50　車道交差部と交差点の一致確認

5.　道路ポリゴンのはみ出し確認

作成した道路面が、分割前の道路ポリゴンからはみ出していないか確認する。はみ出している道路ポリゴンが存在する場合はエラーとし、位置情報を通知する。

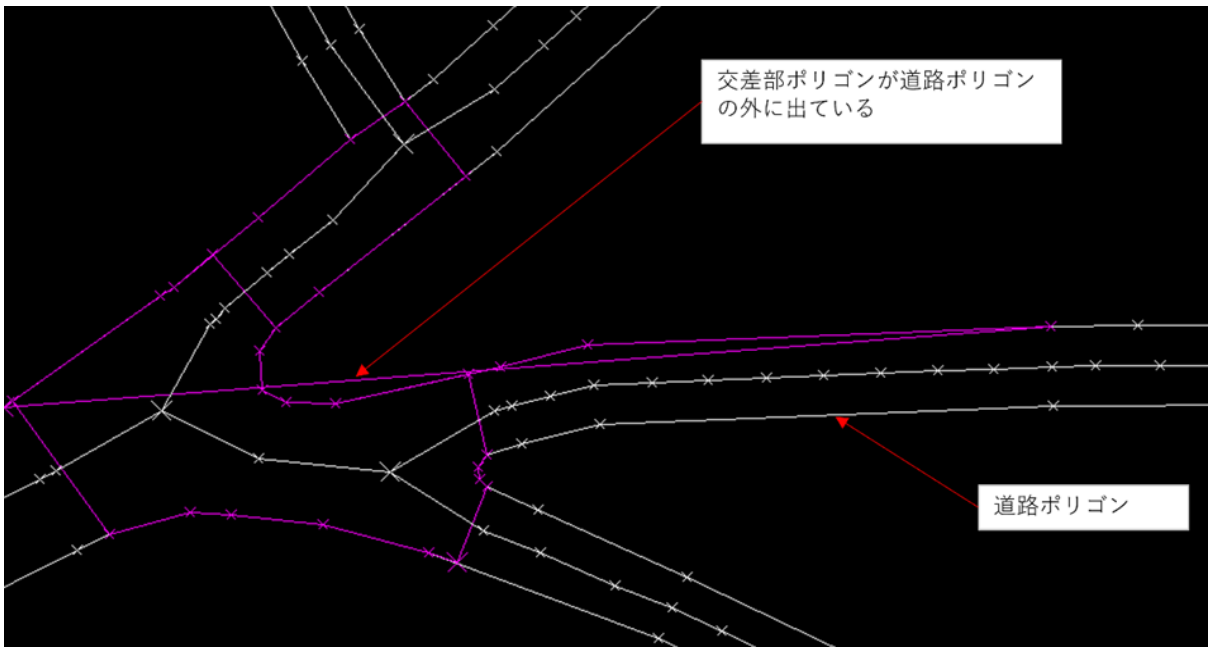


図 3-51　道路ポリゴンのはみ出し確認

6.　車道交差部の重畳確認

作成した車道交差部において、車道交差部同士で重畳が発生していないか確認する。重畳が発生している場合はエラーとし、位置情報を通知する。

7.　車道交差部の道路分割線数の確認

三叉路の場合は車道交差部の道路分割線数は 3 本、十字路の場合は車道交差部の道路分割線数は 4 本となるため、車道交差部の道路分岐数と車道交差部の道路分割線数は一致すると仮定し、車道交差部の道路分岐数と道路分割線数の比較を行う。

車道交差部の道路分岐数と道路分割線数が一致しない場合はエラーとし、位置情報を通知する。

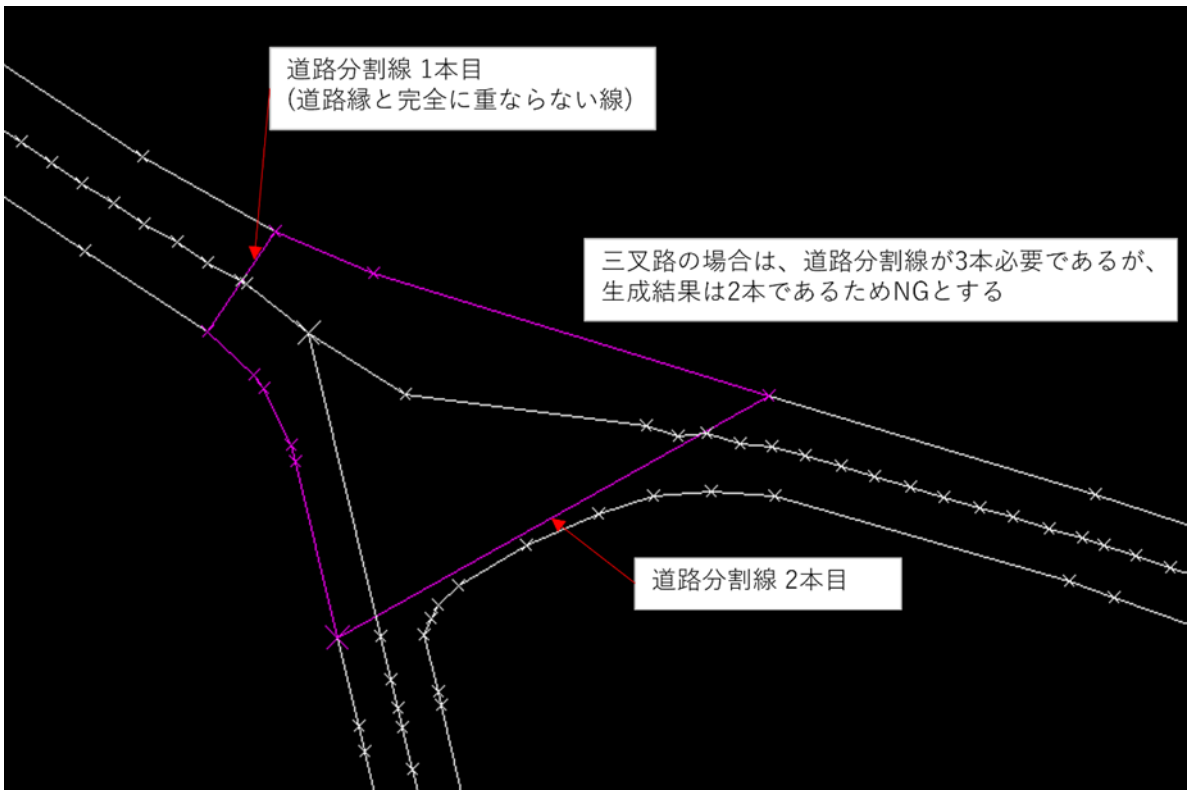


図 3-52　車道交差部の道路分割線数の確認

8.　極小ポリゴンの確認

作成した道路面において、決められたしきい値より面積が小さいポリゴンの確認を行う。面積が小さいポリゴンが存在する場合はエラーとし、位置情報を通知する。

9. 車道交差部ポリゴン中心と交差点間距離の確認

車道交差部ポリゴンの中心と交差点間の距離を確認する。2 点間の距離が決められたしきい値より離れている場合は、作成した車道交差部ポリゴンをエラーとし、位置情報を通知する。

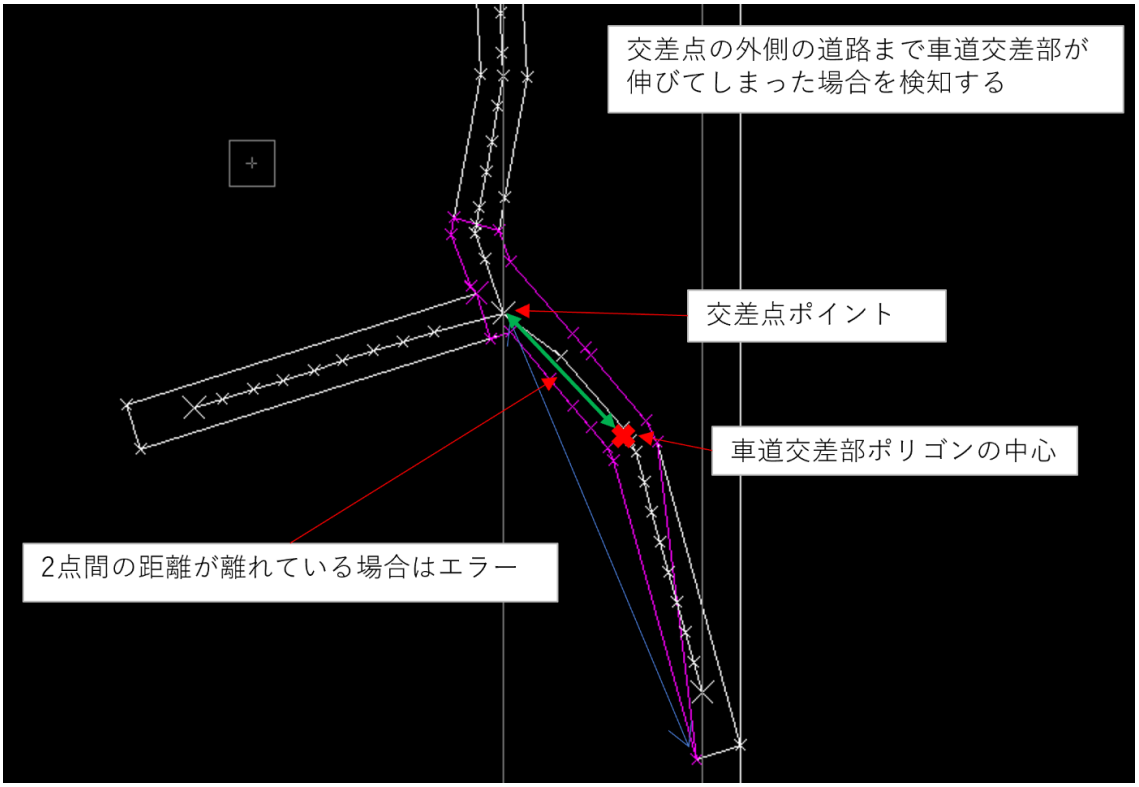


図 3-53 車道交差部ポリゴンの中心と交差点間距離の確認

(11) シェープファイル出力機能

パラメータファイルの OutputFolderPath で指定されたフォルダに道路面情報をシェープファイルとして出力する。シェープファイルはシングルポリゴンとする。

CityGML の uro:RoadStructureAttribute の uro:sectionType 属性に該当する、道路の形状属性(トンネル、高架橋、アンダーパス、交差部)を付与する。属性名は section とする。(シェープファイルの属性名には文字数制限（半角文字の場合 10 文字、全角文字の場合は 5 文字）がある。）

属性値はコードリスト（RoadStructureAttribute_sectionType.xml）から該当する値を設定する。

3.3.11.LOD1 道路モデル CityGML 変換ツール

3.3.11.1.機能一覧

LOD1 道路モデル CityGML 変換ツールの機能を表 3-18 に示す。各処理の詳細は 3.3.11.5 機能詳細に記載する。

表 3-18　機能一覧

No.	機能
1	パラメータファイルを読み込む。 パラメータファイルを用いて入出力データのパス、平面直角座標系の系番号を指定する。
2	マップファイルを読み込む。 シェープファイルの属性名と CityGML の属性名とをひも付けるマッピング情報を読み込む。マップファイルでの記載の有無により、付与する属性を決定する。したがって、マップファイルにて対応関係を持たない場合は該当属性を持たない CityGML へ変換される。
3	入力データ(シェープファイル)を読み込む。 属性付き LOD1 道路面情報が記載されたシェープファイルを読み込む。
4	属性を付与する。 マップファイルに記載されている CityGML の属性名とシェープファイルの属性名の対応関係を基にして、シェープファイルの属性値を CityGML 属性値として付与する。
5	座標を変換する。 入力シェープファイルの座標系が平面直角座標系であるため、CityGML の仕様に合わせ、座標系を「日本測地系 2011」における経緯度座標系に変換する。
6	基準地域メッシュに分割する。 作成した道路モデルを CityGML ファイルに出力する場合、JISX0410 地域メッシュコードに定められた基準地域メッシュ単位にファイルを出力する必要がある。
7	CityGML ファイルを出力する。 属性付き LOD1 道路面情報を CityGML ファイルに出力する。

CityGML 変換ツールは、シェープファイル形式の LOD1 道路面モデルデータを CityGML ファイルに変換するツールである。作成する CityGML の属性情報は、シェープファイルに記載されている属性情報を用いて付与する。CityGML の属性名とシェープファイルの属性名の対応付けに関しては、別途入力するマップファイルに対応付け情報が記載されているものとする。

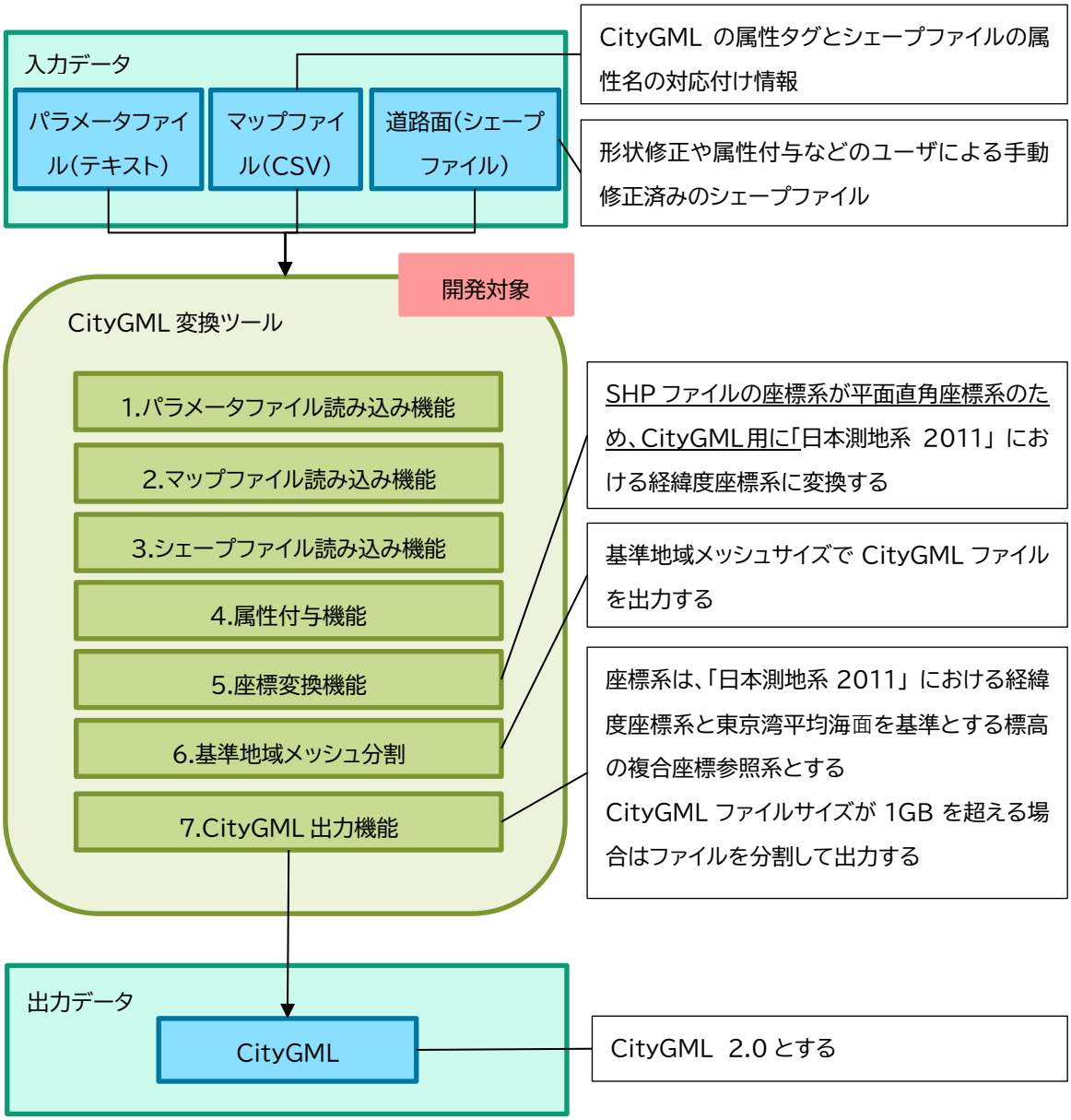


図 3-54　CityGML 変換ツールの機能概要

3.3.11.2.入力ファイル

入力ファイルを表 3-19 に示す。

表 3-19　入力ファイル

項目	形式	内容
道路面シェープファイル	シェープファイル	LOD1 道路面情報が記載されたシェープファイル
マップファイル	CSV	本ツールにて属性付与する対応付けを記載するファイル
パラメータファイル	テキスト	本ツールのパラメータを指定するファイル

(12) 道路面シェープファイル

道路 LOD1 自動作成ツールの出力ファイルを入力とする。道路面に GIS ソフトウェアで属性付与したシェープファイルも入力可能とする。仕様は「(1). シェープファイル」に記載する。

(13) マップファイル

ファイルフォーマットは CSV 形式.(*.csv)とし、ユーザがテキストエディタ等で容易に編集できることを想定する。なお、文字コードは Shift_JIS とする。

マップファイルのフォーマットは、1 行に 1 属性ずつ、変換対象とするか否かを指定するフラグと、マップファイル読み込み機能にて記載の属性名とシェープファイルのフィールド及びコードリストをカンマ区切りで記載するものとする。

［マップファイル　サンプル］

```
SettingAttrFlg,CityGMLAttrName,SHPAttrName
1,gml:description,desc
1,gml:name,name
1,core:creationDate,createDate
1,core:terminationDate,termDate
1,tran:class,class
1,tran:function,func
1,tran:usage,usage
1,uro:dmCode,dmCode
1,uro:meshCode,meshCode
1,uro:srcScale,srcScale
1,uro:geometrySrcDesc,geomDesc
1,uro:thematicSrcDesc,themDesc
1,uro:widthType,widthType
1,uro:width,width
1,uro:numberOfLanes,numOfLanes
1,uro:sectionType,sectType
1,uro:sectionID,sectID
1,uro:routeName,routeName
1,uro:weekday12hourTrafficVolume,12hTrafVol
1,uro:weekday24hourTrafficVolume,24hTrafVol
1,uro:largeVehicleRate,lgVehRate
1,uro:congestionRate,cgstnRate
1,uro:averageTravelSpeedInCongestion,ATSIC
1,uro:averageInboundTravelSpeedCongestion,AIBTSIC
1,uro:averageOutboundTraelSpeedInCongestion,AOBTSIC
1,uro:averageInboundTravelSpeedNotCongestion,AIBTSNC
1,uro:averageOutboundTravelSpeedNotCongestion,AOBTSNC
1,uro:observationPointName,obsPtName
1,uro:reference,ref
1,uro:surveyYear,survYear
```

(14) パラメータファイル

パラメータファイルの内容について表 3-20 に示す。なお、ファイルフォーマットはテキスト形式とし、文字コードは Shift_JIS とする。

表 3-20　パラメーター一覧

設定項目	内容
LOD1RoadSHPPath	道路面シェープファイル(*.shp)パス。 記載された同階層に、同名の.shp、.shx、.dbf、.prj の拡張子のファイルが存在するものとする。
JPZone	シェープファイルで使用している平面直角座標系の系番号(入力範囲:1 - 19)
MapFilePath	マップファイルパス
OutputFolderPath	出力フォルダパス

[パラメータファイル サンプル]

```
[Setting]
LOD1RoadSHPPath = C:/work/data/LOD1/road.shp
JPZone=6
MapFilePath= C:/work/map.csv
OutputFolderPath=C:/work/citygml
```

3.3.11.3.出力ファイル

出力ファイルを表 3-21 に示す。

表 3-21　出力ファイル

項目	形式	内容
道路 LOD1 CityGML ファイル	CityGML	CityGML 2.0 形式の LOD1 道路モデルの CityGML ファイル

(1) 道路 LOD1CityGML ファイル

ファイルフォーマットは CityGML 2.0(*.gml)とし、文字コードは UTF-8(BOM 付き)とする。

LOD1 道路面の形状情報と属性情報(CityGML の uro:RoadStructureAttribute の uro:sectionType 属性に該当する情報)を有する。1 ファイルは基準地域メッシュサイズとし、ファイルサイズは 1GB を超えないものとする。(図 3-55 に示すように、ファイルサイズが 1GB を超える場合は、2 分の 1 地域メッシュに分割する。2 分の 1 地域メッシュサイズに分割しても 1GB を超える場合は、4 分の 1 地域メッシュに分割する。)。分割する場合はファイル名の末尾に連番を付与する。

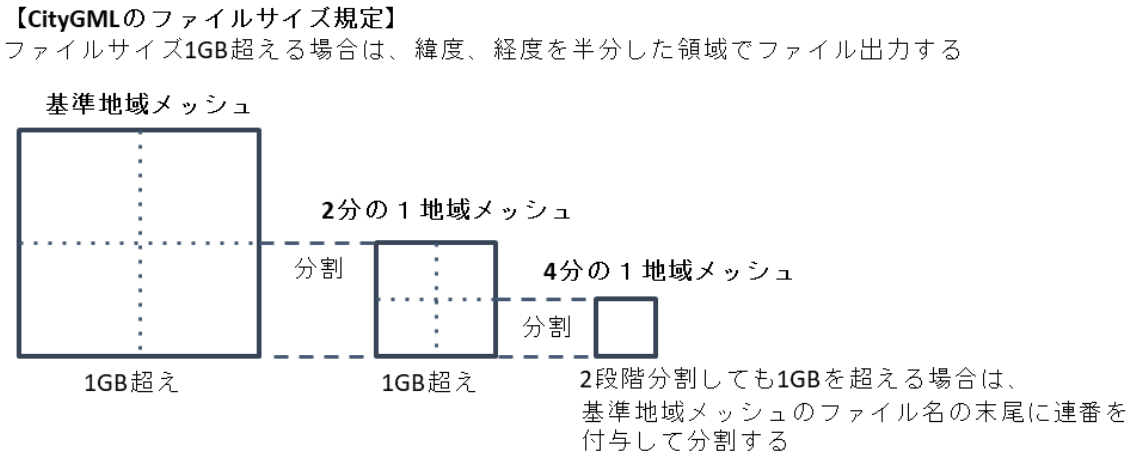


図 3-55　CityGML ファイルサイズ規定

CityGML ファイル内の座標系は、座標系は「日本測地系 2011」における経緯度座標系と東京湾平均海面を基準とする標高の複合座標参照系とする。

3.3.11.4.制約事項

CityGML に記載する属性情報は、入力シェープファイルの属性情報を流用するものとするが、マップファイルに記載がない属性がシェープファイルに存在する場合は、CityGML にその属性は記載されない。

3.3.11.5.機能詳細

(1) パラメータファイル読み込み機能

ツールの入力データパス、出力フォルダパス及びシェープファイルの平面直角座標系の系番号などのパラメータを記載したパラメータファイルを読み込む。

(2) マップファイル読み込み機能

シェープファイルの属性名と CityGML の属性名とをひも付けるマッピング情報を読み込む。

マッピング対象はマップファイルでの記載の有無により、付与する属性を決定する。したがって、マップファイルにて対応関係を持たない場合は該当属性を持たない CityGML へ変換される。

マップファイルは、1 行に 1 属性ずつ、変換対象とするか否かを指定するフラグ(SettingAttrFlg)を持つ。フラグが ON の場合はその行の設定を読み込み、フラグが OFF の場合はシェープファイルや CityGML の属性名が記載されていても設定を読み込まない。なお、フラグが ON の場合で、シェープファイルの属性又は CityGML の属性名が空欄の場合はエラー行として設定を読み込まない。

表 3-22　tran:Road クラスのマッピング対象属性

属性名	説明
gml:description	道路の概要
gml:name	道路を識別する名称 道路法に基づき路線が指定又は認定された路線名
core:creationDate	データが作成された日
core:terminationDate	データが削除された日
tran:class	交通分類 コードリスト(TransportationComplex_class.xml)から選択する。
tran:function	道路法における道路の区分及び建築基準法における道路の区分 コードリスト(Road_function.xml)から選択する。
tran:usage	道路の利用方法 コードリスト(Road_usage.xml)から選択する。
uro:tranDmAttribute	公共測量標準図式による図形表現に必要な情報
uro:tranDataQualityAttribute	作成した道路データ品質に関する情報 個々のデータのメタデータを記述する場合にのみ作成する。
uro:roadStructureAttribute	当該道路の道路構造に関する情報
uro:trafficVolumeAttribute	当該道路の通行する車両の量に関する情報

表 3-23　uro:DmAttributeクラスのマッピング対象属性

属性名	説明
uro:dmCode	公共測量標準図式の図式分類コード レイヤ番号(2 桁)とデータ項目(2 桁)からなる 4 桁の半角数字の列。コードリスト(Common_dmCode.xml)より選択する。
uro:meshCode	数値地形図データファイル仕様にもとづいて設定される図郭識別番号

表 3-24　uro:TransportationDataQualityAttribute クラスのマッピング対象属性

属性名	説明
uro:srcScale	元となるデータの地図情報レベル。コードリスト(TransportationDataQualityAttribute_srcScale.xml)から選択する。LOD1 と LOD2 のように、異なる LOD の幾何オブジェクトを持ち、それぞれの地図情報レベルが異なる場合は、最も高い地図情報レベルを記載する。
uro:geometrySrcDesc	幾何オブジェクトを作成する元となるデータの説明。コードリスト(TransportationDataQualityAttribute_geometrySrcDesc.xml)から選択する。道路オブジェクトに複数の LOD が含まれる場合は、最も高度な LOD について記述する。
uro:thematicSrcDesc	主題属性を作成する元となるデータの説明。コードリスト(TransportationDataQualityAttribute_thematicSrcDesc.xml)から選択する。

以下は本システムが LOD1 を対象としているため、対象外とする

uro:appearanceSrcDesc	テクスチャ画像を作成する基となるデータの説明。
uro:lodType	道路オブジェクトに適用された LOD3 の詳細な区分。

表 3-25　uro:RoadStructureAttribute クラスのマッピング対象属性

属性名	説明
uro:widthType	幅員の区分 コードリスト(RoadStructureAttribute_widthType.xml)から選択する。都市計画基礎調査で収集されている場合にのみ作成する。
uro:width	中央帯、車道、路肩、植樹帯、歩道等及び環境施設帯(環境施設帯の中の路肩、植樹帯、歩道等の部分を除いた部分)の幅員を合計した幅員 単位は m とする。
uro:numberOfLanes	上下線の合計(一方通行区間の場合を除く)の車線数 道路構造令第 2 条第 7 号の登坂車線、同第 2 条第 9 号の変速車線及び同第 2 条第 14 号の停車帯、及びゆずり車線は車線数には含めない。 交差点付近において、右左折のための車線が設けられている場合はこの数を含まない。 1 車線道路は道路構造令第 5 条 1 項ただし書きによって、車線により構成されない車道を持つ道路であるが、ここでは車線数=1 とする。1 車線道路は車道幅員が 5.5m 未満の場合とする。 道路構造が交差部の場合、この属性は作成しない。
uro:sectionType	道路構造の種別 コードリスト(RoadStructureAttribute_sectionType.xml)から選択する。

表 3-26　uro:TrafficVolumeAttribute クラスのマッピング対象属性

属性名	説明
uro:sectionID	交通量調査において、調査の単位となる交通調査基本区間に付与される番号 原則として「都道府県(2 桁)」+「道路種別(1 桁)」+「(路線番号(4 桁)」+「順番号(4 桁)」からなる 11 桁の番号。
uro:routeName	路線名
uro:weekday12hourTrafficVolume	平日 7 時～19 時までに通過する車両台数。単位は台とする。
uro:weekday24hourTrafficVolume	平日 7 時～翌朝 7 時又は 0 時～翌日 0 時までに通過する車両台数。単位は台とする。
uro:largeVehicleRate	自動車類交通量に対する大型車交通量の割合。単位は%とする。
uro:congestionRate	交通調査基本区間の交通容量に対する交通量の比。単位は%とする。
uro:averageTravelSpeedInCongestion	朝のラッシュ時間帯(7 時～9 時)又は夕方のラッシュ時間帯(17 時～19 時)において平均旅行速度を集計し、その遅い方の時間帯の旅行速度。都市計画基礎調査で収取されている場合にのみ作成する。単位は km/h とする。
uro:averageInboundTravelSpeedCongestion	朝のラッシュ時間帯(7 時～9 時)又は夕方のラッシュ時間帯(17 時～19 時)において上り線における平均旅行速度を集計し、その遅い方の時間帯の旅行速度。単位は km/h とする。
uro:averageOutboundTraelSpeedInCongestion	朝のラッシュ時間帯(7 時～9 時)又は夕方のラッシュ時間帯(17 時～19 時)において下り線における平均旅行速度を集計し、その遅い方の時間帯の旅行速度。単位は km/h とする。
uro:averageInboundTravelSpeedNotCongestion	昼間非混雑時(9 時～17 時)における上り線の平均旅行速度。単位は km/h とする。
uro:averageOutboundTravelSpeedNotCongestion	昼間非混雑時(9 時～17 時)における下り線の平均旅行速度。単位は km/h とする。
uro:observationPointName	交通量等を観測した地点の名称。
uro:reference	対象となる道路の区間を図上で域別する番号。
uro:surveyYear	調査が実施された年。必須とする。

(3) シェープファイル読み込み機能

属性付き LOD1 道路面情報が記載されたシェープファイルを読み込む。

(4) 属性付与機能

マップファイルに記載されている CityGML の属性名とシェープファイルの属性名の対応関係を基にして、シェープファイルの属性値を CityGML 属性値として付与する。マップファイルに記載がない属性がシェープファイルに存在している場合は、その属性は CityGML に付与しない。

(5) 座標変換機能

入力シェープファイルの座標系が平面直角座標系であるため、CityGML の仕様に合わせ、形状情報の座標系を「日本測地系 2011」における経緯度座標系に変換する。なお、シェープファイルで使用されている平面直角座標系の系番号は、パラメータファイルの JPZone に指定するものとする。

(6) 基準地域メッシュ分割機能

作成した道路モデルを CityGML ファイルに出力する場合、JISX0410 地域メッシュコードに定められた基準地域メッシュ単位にファイルを出力する必要がある。

現状、入力シェープファイルは基準地域メッシュよりも広範囲である行政界範囲を想定しているため、CityGML 出力時には作成した道路モデルがどの基準地域メッシュに属するか判断する必要がある。なお、道路モデルが基準地域メッシュの境界にまたがっている場合は、各メッシュに平面投影した形状が含まれる面積の割合を算出し、この割合が最大となるメッシュにモデルを出力する。面積計算の単位はm^2とし、小数点以下 2 桁(3 桁目で四捨五入)で比較する。面積が同じ場合は、メッシュ番号の小さい方に出力する。

(7) CityGML ファイル出力機能

属性付き LOD1 道路面情報を CityGML 2.0 ファイルに出力する。なお、文字コードは UTF-8(BOM 付き)とする。CityGML ファイル内の座標系は、座標系は「日本測地系 2011」における経緯度座標系と東京湾平均海面を基準とする標高の複合座標参照系とする。

属性値はマップファイルの内容を基にシェープファイルに入力されている属性値を適用する。

［ファイルサイズ制限］

CityGML ファイルのサイズが 1GB を超える場合はファイルを分割する必要がある。基準地域メッシュを分割する場合は、緯線、経線方向にそれぞれ 2 等分してできる 2 分の 1 地域メッシュに分割する。分割後のファイルにおいて、ファイルサイズが 1GB を超えるファイルが存在する場合は、サイズが超過しているファイルに対してのみ同様の分割処理を行い、4 分の 1 地域メッシュに分割する。4 分の 1 地域メッシュにしてもファイルサイズが 1GB を超える場合は、CityGML のファイル名称のオプションを使用してファイルを分割する。(図 3-55 参照)

3.4. LOD1 道路モデル自動作成ツールの検証

LOD1 道路モデル自動作成ツールを本章では本システムという。

3.4.1.検証概要

開発した LOD1 道路モデル自動作成ツールのモデル正確度を検証した。モデル正確度の検証には、一連の処理の出力結果が確認できる範囲として、岐阜県岐阜市の一部地域（下図、赤枠内）を検証対象地域として選定した。

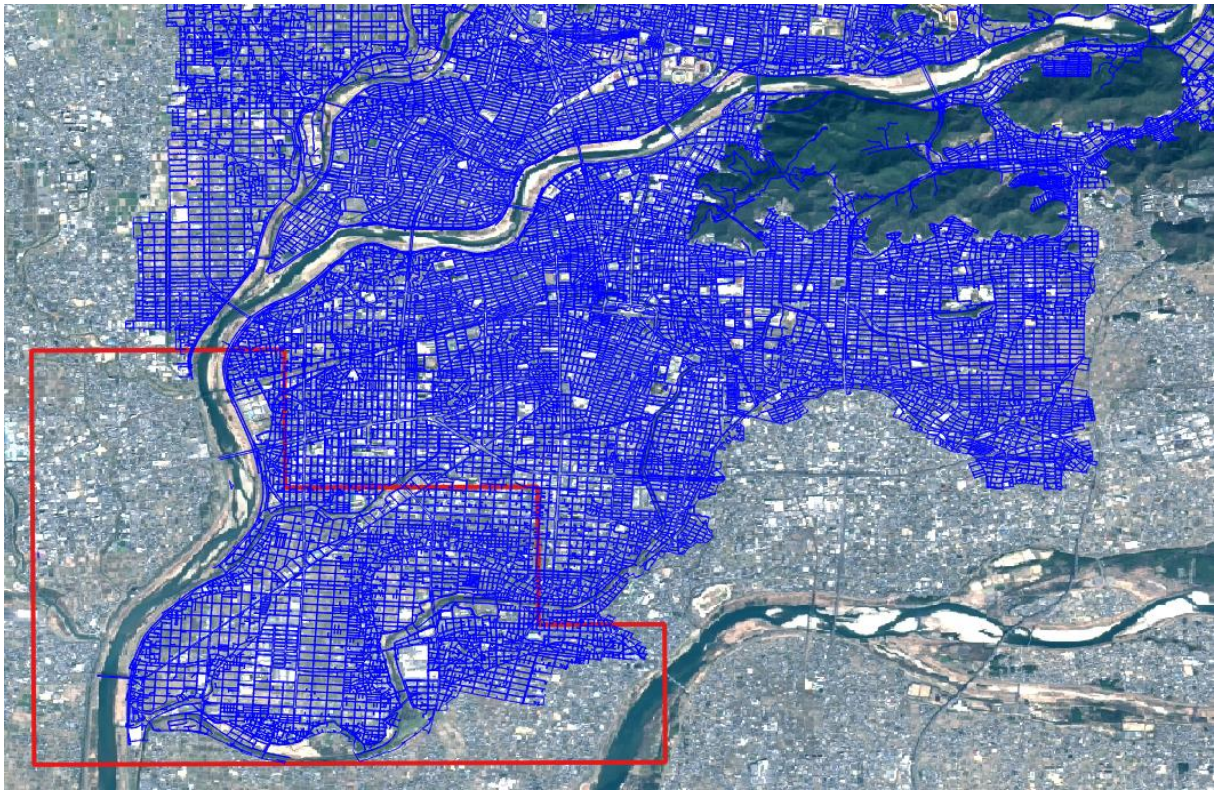


図 3-56　モデル正確度検証の対象地域

3.4.1.1. 評価基準

評価基準を表 3-27 に示す。

表 3-27　評価基準

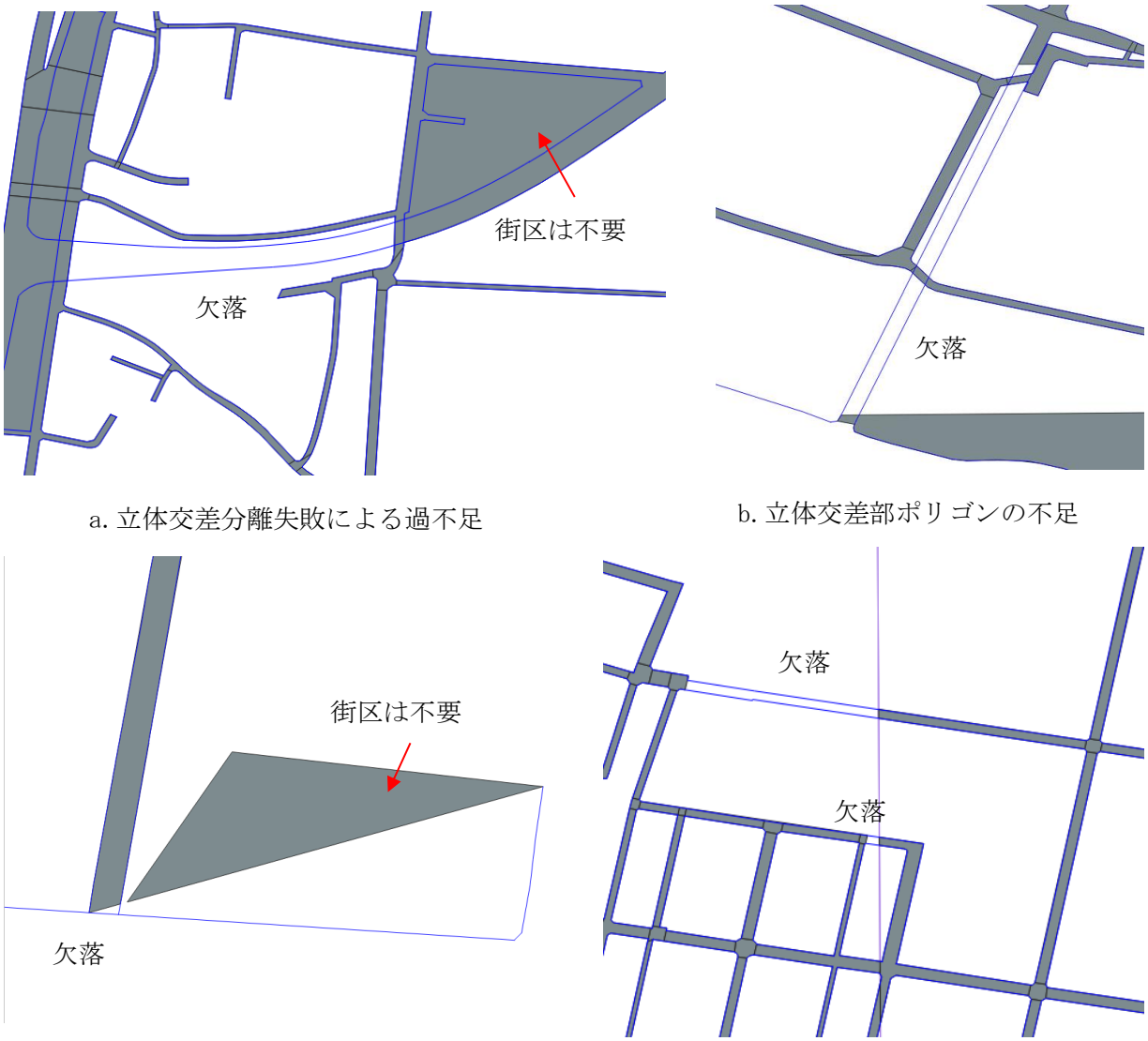
No.	評価基準
1	道路ポリゴンが過不足なく生成されているか
2	道路ポリゴンの形状は正確か
3	道路構造変化点（橋梁、トンネル）の分離は正確か
4	車道交差部の分離は正確か
5	属性は正しく付与されているか

3.4.1.2. 評価結果

評価基準ごとの評価結果を以下に示す。

① 道路ポリゴンの過不足

評価対象の道路ポリゴンにおいて、道路ポリゴンの欠落を確認した。道路ポリゴンの欠落は、立体交差が存在する地点、入力道路縁のデータ境界地点、及び、並列処理対象領域の境界地点で発生する可能性がある。道路ポリゴンの過不足が発生する原因の考察は、図 3-57 に記載する。



a. 立体交差分離失敗による過不足

b. 立体交差部ポリゴンの不足

c. データ境界による過不足

d. 処理対象領域境界での欠落

青 ： 道路縁

図 3-57 ポリゴンの過不足

② 道路ポリゴンの形状の正確性

検証対象地域において、道路構造変化点（橋梁、トンネル）及び車道交差部の形状の確認を行った。確認結果は以下のとおりである。

表 3-28　道路ポリゴンの形状の正確性の評価ランク

評価ランク	説明
A	形状が正しい
B	形状に軽微なずれが存在する
C	形状に大幅なずれが存在する
D	分割前の道路ポリゴンが生成できていない

表 3-29　評価地点数

項目	地点数
車道交差部	3135
橋梁	132
トンネル	15
総計	3282

表 3-30　道路ポリゴン形状の正確性検証結果

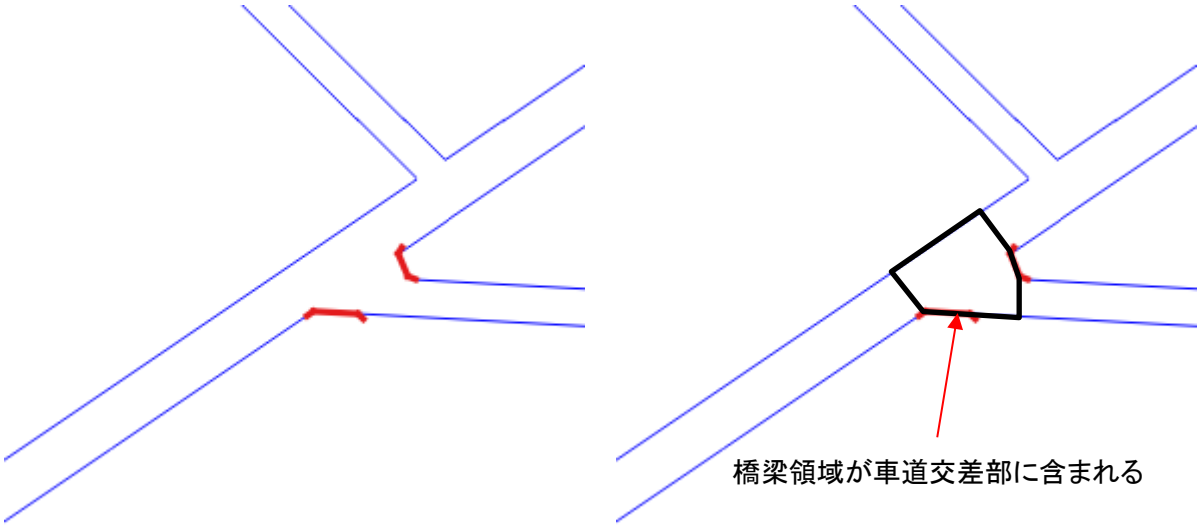
評価ランク	車道交差部	橋梁	トンネル	結果
A	671（20.44%）	27（20.45%）	0（0%）	698（21.27%）
B	787（23.98%）	6（4.55%）	0（0%）	793（24.16%）
C	1576（48.02%）	89（67.42%）	3（20%）	1668（50.82%）
D	101（3.08）	10（7.58%）	12（80%）	123（3.75%）

③　道路構造変化点（橋梁、トンネル）の分離の正確性

検証対象地域における道路構造変化地点は以下のとおりである。なお、橋梁において図 3-58 のように橋梁の領域が車道交差部の一部となる場合は、生成される道路ポリゴンは橋梁領域を含む車道交差部として分割され、橋梁のポリゴンが発生しない。そのため、評価対象から除外する。

表 3-31　道路構造変化地点数

項目	地点数
橋梁	132
トンネル	15
総計	147



a. 評価対象外の橋梁　　b. 車道交差部分割例

青 ： 道路縁、赤 ： 道路橋

図 3-58　評価対象外の橋梁

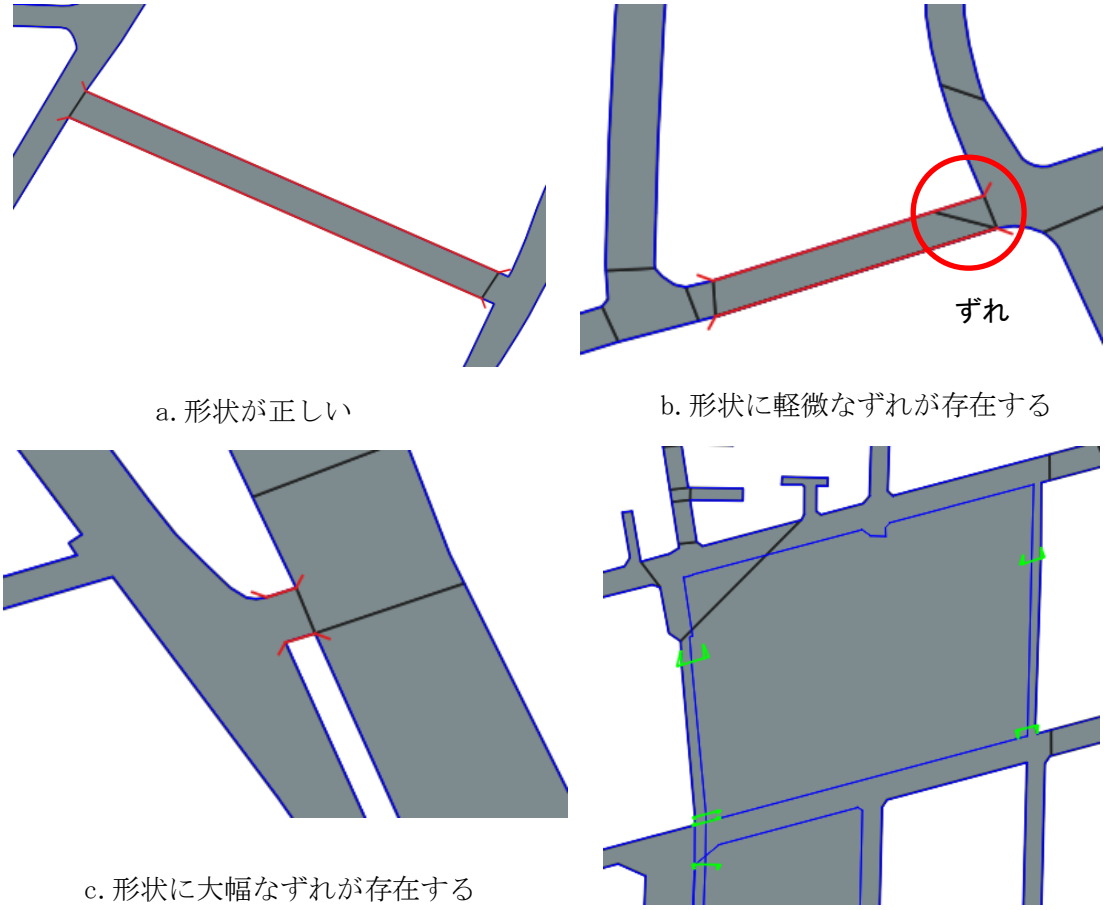
以下に、道路構造変化点の分離の正確性の結果を示す。

表 3-32　道路構造変化点の分離の評価ランク

評価ランク	説明
A	正しい形状で分割されている
B	分割されているが軽微なずれがある
C	形状に大幅なずれがあるが分割されている、又は分割されていない
D	分割前の道路ポリゴンが生成できていない

表 3-33　道路構造変化点の分離結果

評価ランク	橋梁	トンネル	総計
A	27（20.45%）	0（0%）	27（18.37%）
B	6（4.55%）	0（0%）	6（4.08%）
C	89（67.42%）	3（20%）	92（62.59%）
D	10（7.58%）	12（80%）	22（14.97%）



a. 形状が正しい

b. 形状に軽微なずれが存在する

c. 形状に大幅なずれが存在する
又は分割されていない

d. 分割前道路ポリゴンの作成失敗

青 ： 道路縁、赤 ： 道路橋、緑 ： トンネル

図 3-59　道路構造変化点の分離結果例

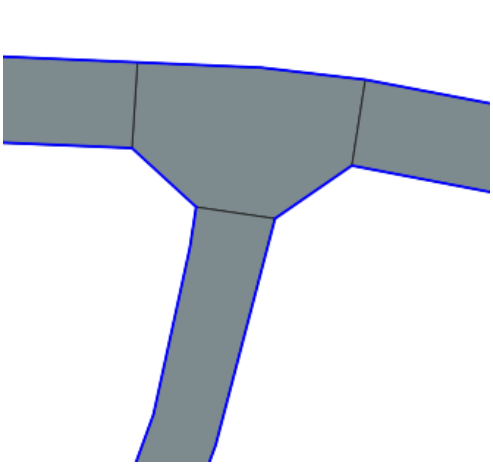
④　車道交差部の分離の正確性

以下に、車道交差部の分離の正確性確認結果を示す。なお、評価対象地点数は 3135 地点である。

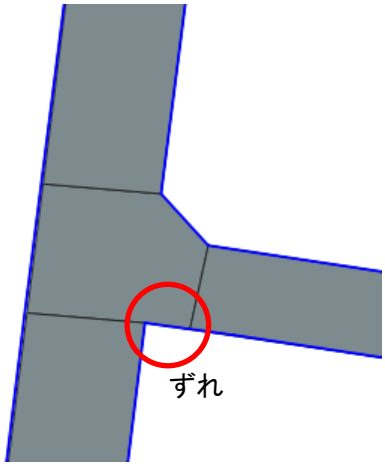
評価ランクに関しては、道路構造変化点の分離の評価ランク（表 3-32）と同一基準を使用している。

表 3-34　車道交差部の分離結果

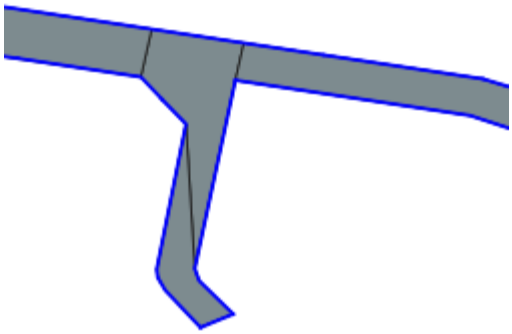
評価ランク	車道交差部
A	671（21.4%）
B	787（25.1%）
C	1576（50.27%）
D	101（3.22%）



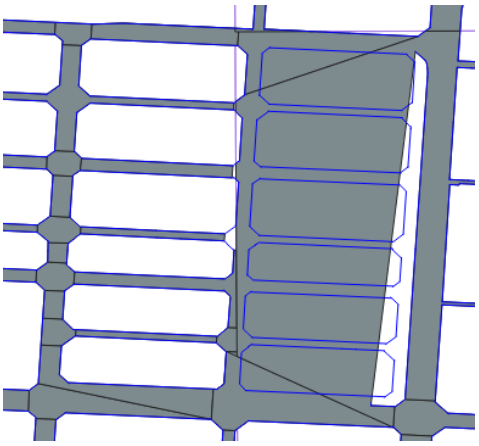
a. 形状が正しい



b. 形状に軽微なずれが存在する



c. 形状に大幅なずれが存在する



d. 分割前道路ポリゴンの作成失敗

青 ： 道路縁

図 3-60　車道交差部の分離結果例

⑤　属性付与の正確性

属性付与の正確性確認では、車道交差部、橋梁、トンネルに対して、形状の正否関係なく、属性が正しく付与されているかの確認を実施した。以下に、属性付与の正確性確認結果を示す。なお、評価地点数は表 3-29 と同様である。

表 3-35　属性付与結果

項目	属性付与が正しい地点数
車道交差部	1773（56.56%）
橋梁	30（22.73%）
トンネル	0（0%）
総計	1803（54.94%）

3.4.1.3. モデル作成例

モデル作成例を図 3-61、図 3-62 に示す。

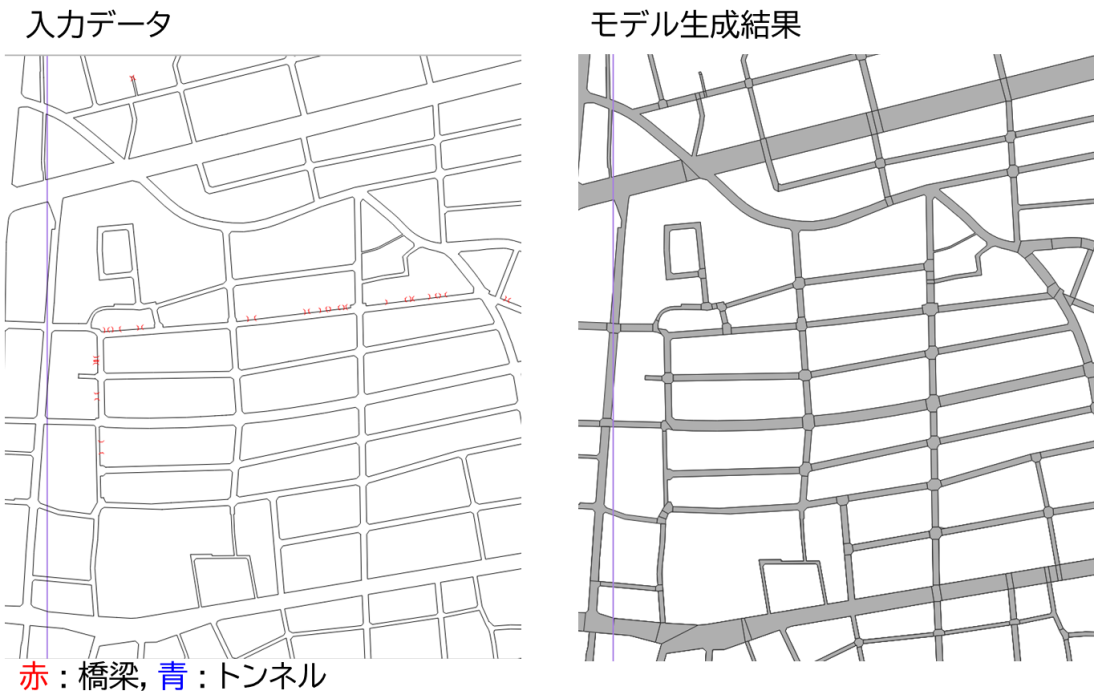


図 3-61　モデル作成例

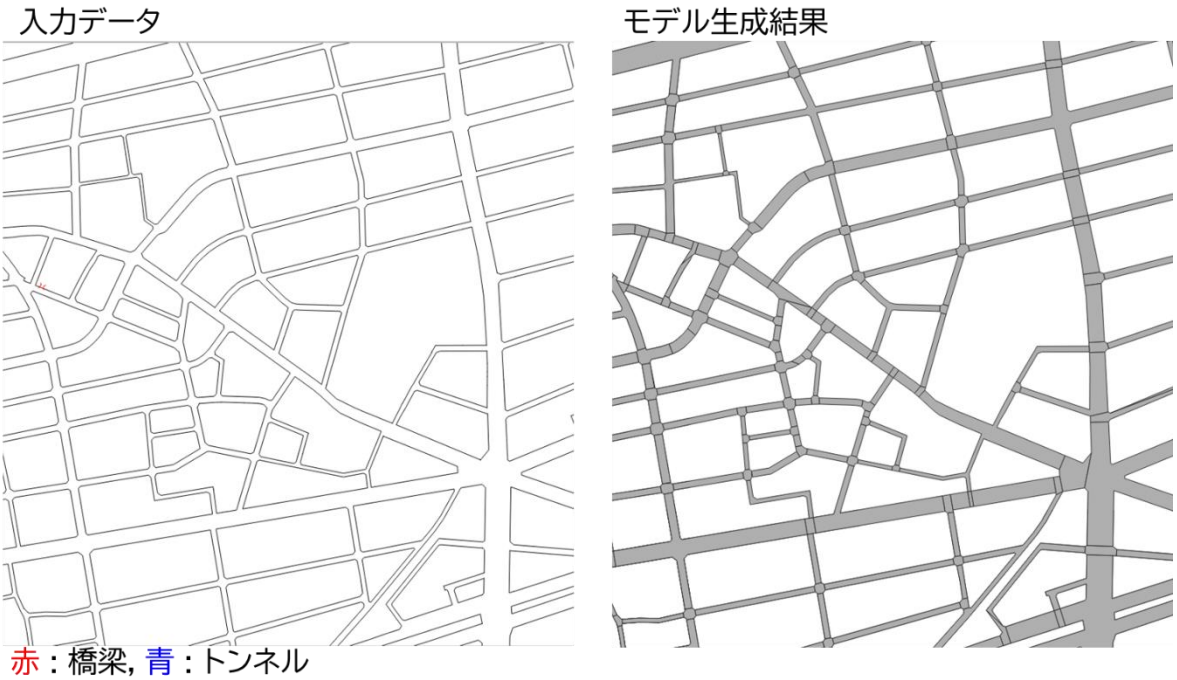


図 3-62　モデル作成例

3.4.2.データ整備事業者による検証

LOD1 道路自動作成ツールの使いやすさを検証することを目的として､データ整備事業者(以降、検証者）の協力の下で検証を行った。

実際に OSS として公開予定であるプログラムを検証者に共有し、検証者のデータでモデルの作成を行った。

3.4.2.1. 試行環境

試行に利用した PC スペックを表 3-36 に示す。

表 3-36　試行に利用した PC スペック

OS	Windows 10 Pro 64 ビット
CPU	Intel® Core™ i7-11800H プロセッサー 8 コア / 16 スレッド / 2.30GHz [最大 4.60GHz] / 24MB キャッシュ
GPU	NVIDIA GeForce RTX 3060 Laptop GPU / GDDR6 6GB
メモリ	32GB メモリ [16GB×2 (DDR4-3200) / デュアルチャネル]
ストレージ	SSD 512GB NVM Express SSD (M.2 PCI Express 接続)

各検証者が試行時に使用したデータを表 3-37 に示す。

表 3-37　試行データ

自治体名	埼玉県松伏町	埼玉県加須市	山梨県富士吉田市
試行範囲の面積	16km^2	133km^2	3km^2
利用データ	DM データをシェープファイルに変換したデータ		
モデル作成処理時間	約 1 時間 30 分	約 9 時間 30 分	約 30 分

3.4.2.2. 試行の観点

試行において表 3-38 の観点でヒアリングした。

表 3-38　試行の観点

項目	定義
品質	作成した道路モデルの形状、交差部等の分離状況を評価
効率性／処理速度	既往の手作業を含む手順と比較し、データ生産が効率化するかを評価
利用しやすさ	開発したツールの使いやすさを評価

3.4.2.3. 試行による評価結果

試行による評価結果を以下に示す。複数事業者による評価であるため、相反する評価結果もある。

品質評価結果から、精度上、正確にできる割合が少ないことが問題であり、実用には精度改善が必要であることがわかった。特に道路面の生成、構造変化点での分離に課題がある。

表 3-39　品質評価

No.	観点	評価
1	道路ポリゴンが過不足なく生成されているか	街区ポリゴンが作成されたり、道路ポリゴンが作成されなかったり過不足がある。
		一部面が作成されていない、道路縁まで面が作成されていない等、複数個所で過不足が見受けられた。
2	道路ポリゴンの形状は正確か	道路ポリゴンが正確にできていない箇所がある。
		細い道路や脇道が多く密集している範囲に形状エラーが多く見受けられた。
3	道路構造変化点（橋梁、トンネル）の分離は正確か	全く異なる箇所で分離されている箇所がある。
4	車道交差部の分離は正確か	全く異なる箇所で分離されている箇所がある。
		問題なく分離されている。
5	属性は正しく付与されているか	分離が正確にできていないため、結果的に属性付与が正しくできていない。
6	自由意見	3 市町を対象に施行した結果、3～4 割は上手くモデルが生成されている印象。 エラーの種類については、目視検査での修正に時間がかかるものもあったため、エラー箇所を通知する機能があると作業工数の減少が見込める。

効率性においては、前述の道路ポリゴン生成、道路構造分離作業での精度上の課題解決が必要である。

表 3-40　効率性評価

No.	観点	評価
1	道路ポリゴン生成作業	人工削減できない。
		人工は変わらない 一部において高精度のポリゴンが生成できていたが、形状エラーのポリゴンも多く見受けられた。
2	車道交差部分離作業	人工削減できない。
		人工削減できる。 精度よくポリゴンを生成できている箇所は、上手く車道交差部の分離が行えていた。
3	道路構造変化点分離作業	人工削減できない。

		道路構造変化点に形状エラーが多く見受けられたため、分離ができていない。
4	チェック作業	目視検査で認識できる不備もあれば、GIS 処理でも検知が難しい箇所も多くあり、チェック作業に工数がかかる印象を受けた。
5	属性付与作業	人工は変わらない。 一部形状エラーがあるため、正しく属性が付与されていない箇所がある。
6	データ形式変換作業	人工は変わらない。
7	自由意見	チェック作業に多くの工数がかかったため、チェック作業を簡易にできると効率性が向上すると考える。

GUI の方が使いやすいという意見、利用しやすいという意見があった。入出力フォーマットは今回の仕様で問題ないといえる。

表 3-41　利便性評価

No.	観点	評価
1	ツールの動作環境は利用しやすいか	コマンドラインでの実行は利用しにくい。
		利用しやすい
2	ユーザインタフェースは利用しやすいか	コマンドライン実行ではなく GUI 環境の方が良い。
		問題なく利用できる
3	入力データは利用しやすいデータ形式であるか	汎用的なシェープファイルのため、問題ない。
		問題なく利用できる
4	出力データは利用しやすいデータ形式であるか	汎用的なシェープファイルのため、問題ない。
		利用しやすい
5	自由意見	パラメータファイル作成等が面倒に感じた。

3.4.3.モデル作成失敗要因分析

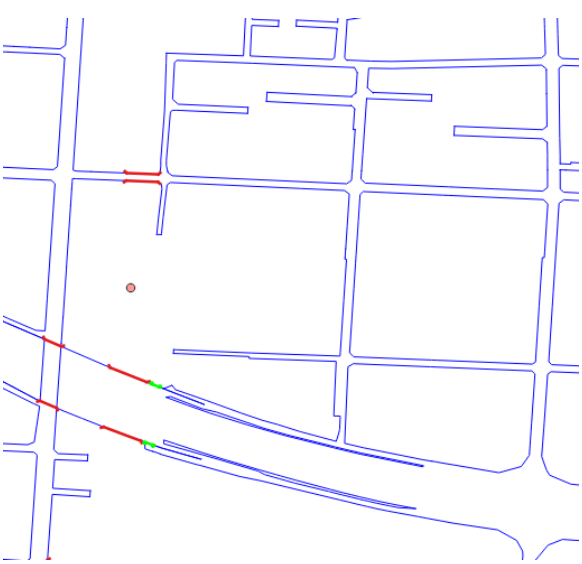
作成失敗の要因分析と対策検討を行った。（表 3-42）

表 3-42　作成失敗の要因分析と対策検討

No.	課題	対策検討
1	立体交差分離の失敗による道路ポリゴンの過不足	立体交差の分離処理の改良が必要
2	立体交差部ポリゴンの不足	立体交差の分離処理で分離した道路橋及びトンネル部分の道路ポリゴン作成処理の追加が必要
3	データ境界による道路ポリゴンの過不足	街区ポリゴン作成時にデータ境界部分における街区ポリゴンの作成方法の改良が必要
4	処理対象領域境界での道路ポリゴンの欠落	車道交差部の分割処理の改良が必要
5	車道交差部の分割	車道交差部の分割処理の見直しが必要
6	道路ポリゴンの作成	入力道路縁をクラスタリングしてから、道路縁全体の外形ポリゴンを作成する必要がある

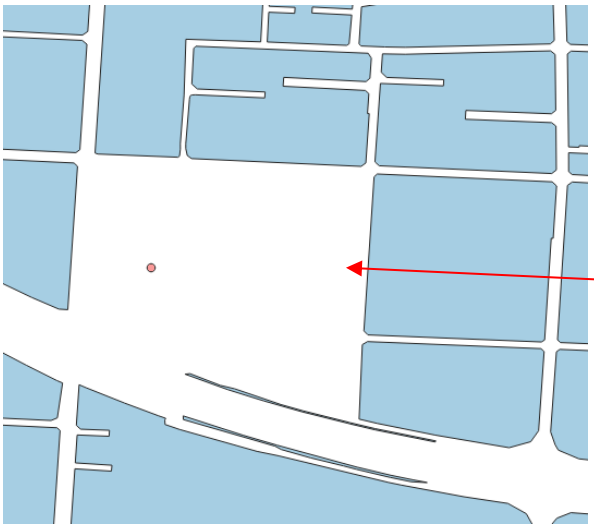
① 立体交差分離失敗による道路ポリゴンの過不足

LOD1 道路自動作成ツールでは、道路縁の入力範囲ポリゴンと街区ポリゴンの差分を取得することで道路縁から道路ポリゴンを作成している。道路ポリゴンの作成に必要な街区ポリゴンは、道路縁をグラフ（ノードとエッジの集合）とみなし、深さ優先探索を行うことで作成している。深さ優先探索は、道路縁に立体交差が残っていると予期せぬ結果となり自己交差などの不備がある街区ポリゴンを作成してしまう。道路ポリゴンの作成には、不備のない街区ポリゴンを使用するため、不備が発生した街区部分は道路ポリゴンから除去されずに残る。課題の解消には、立体交差の分離処理の改良が必要である。



a. 入力データ

b. 立体交差除去結果



c. 街区ポリゴン生成結果
（正常ポリゴンのみ）

立体交差の分離に失敗したため
街区ポリゴンの作成に失敗

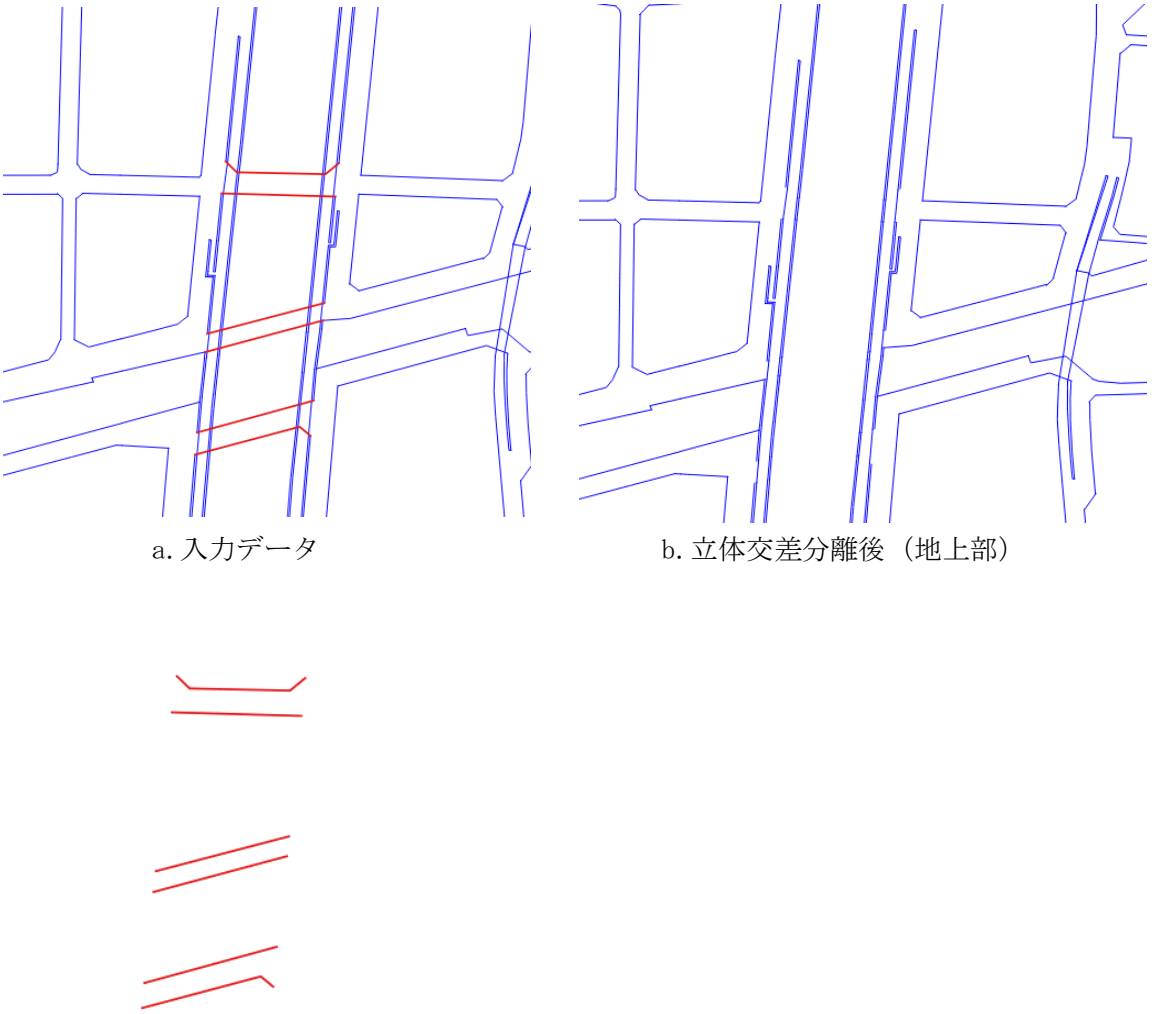
青 ： 道路縁、赤 ： 道路橋、緑 ： トンネル

図 3-63　街区生成失敗原因

② 立体交差部ポリゴンの不足

LOD1 道路自動作成ツールでは、立体交差が発生している道路縁を階層ごと（地上部、高架部、トンネル）に分離した後、地上部の道路縁を使用して道路ポリゴンの作成を行っている。現状、高架部とトンネル部分の道路ポリゴンの作成を行っていないため、立体交差が発生している道路橋、トンネル部分において道路ポリゴンの欠落が発生する。

課題の解消には、未作成部分である道路橋とトンネル部分の道路ポリゴンの作成が必要となるが、立体交差分離後の道路橋とトンネル部分の道路縁は、地上部の道路縁とは異なり細切れの状態となるため、地上部で使用している道路ポリゴンの作成方法と同一の方法では道路ポリゴンが作成できないと考える。立体交差分離後の道路橋とトンネル部分の道路ポリゴンの作成は、作成方法を別途検討する必要がある。



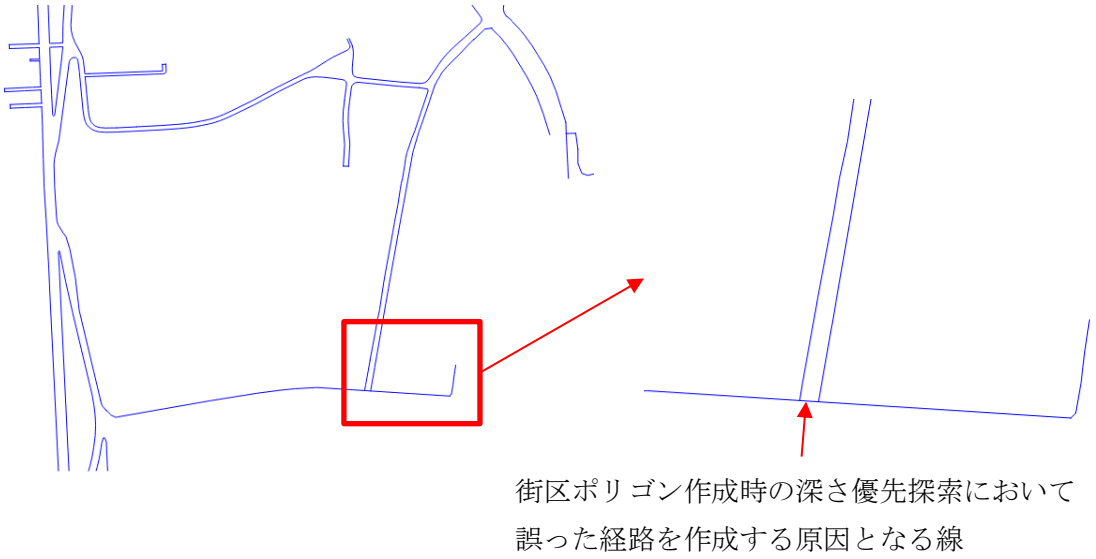
a. 入力データ　　b. 立体交差分離後（地上部）

c. 立体交差分離後（高架部）　　青 ： 道路縁、赤 ： 道路橋

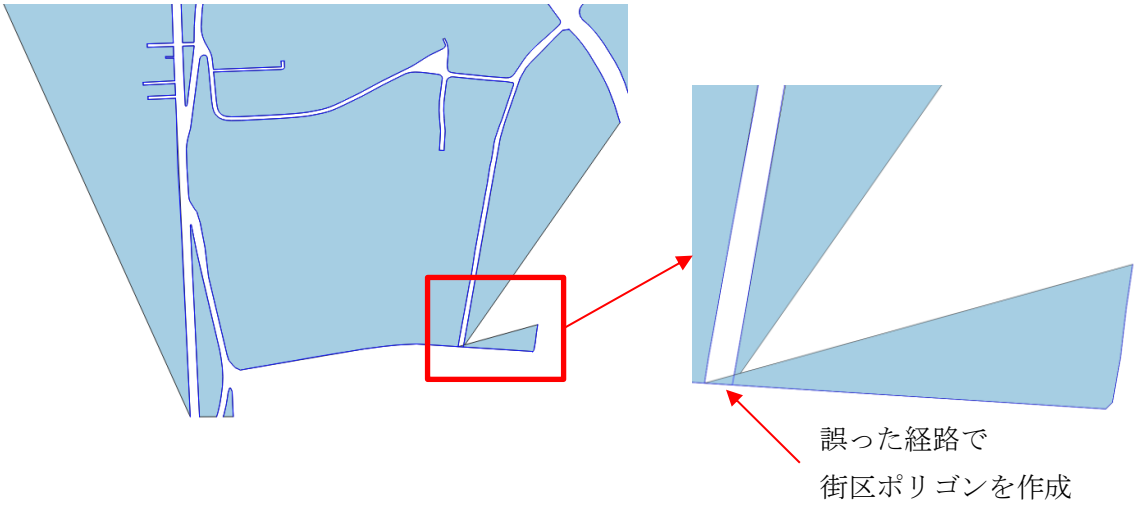
図 3-64　立体交差部ポリゴンの不足原因

③　データ境界による道路ポリゴンの過不足

LOD1 道路自動作成ツールでは、道路縁の入力範囲ポリゴンと街区ポリゴンの差分を取得することで道路縁から道路ポリゴンを作成している。街区ポリゴンを作成する際に使用する道路縁のデータ境界部分では、街区ポリゴンを作成する際に悪影響を及ぼす線が発生している場合があり、この不要線を除去することで街区ポリゴンの作成精度が向上すると考える。



a.　街区ポリゴン作成用の道路縁



b. 街区ポリゴンの作成結果

青 ： 道路縁

図 3-65　データ境界部分における街区生成失敗原因

④　処理対象領域境界での道路ポリゴンの欠落

LOD1 道路自動作成ツールでは処理時間を短縮するために、入力データ範囲を一定間隔で分割した領域（以降、小領域とする。）に対して並列に道路モデル生成処理を実施している。なお、小領域の境界上で道路が不要に分割されないように小領域の周囲 500m を含む領域を道路モデル生成対象とし、小領域の境界上に位置する道路モデルに対して小領域内の道路モデルとして採用するか否かを取捨選択している。この処理仕様のため、小領域の境界上に位置する道路ポリゴンは、隣接する小領域ごとに生成しているが、生成結果が小領域ごとに異なる場合に道路ポリゴンの欠落が発生する。なお、小領域の境界上に位置する道路ポリゴンが小領域ごとに形状が異なる原因

は、車道交差部の分割に失敗するためである。そのため、課題の解消には車道交差部分割の精度向上が必要となる。

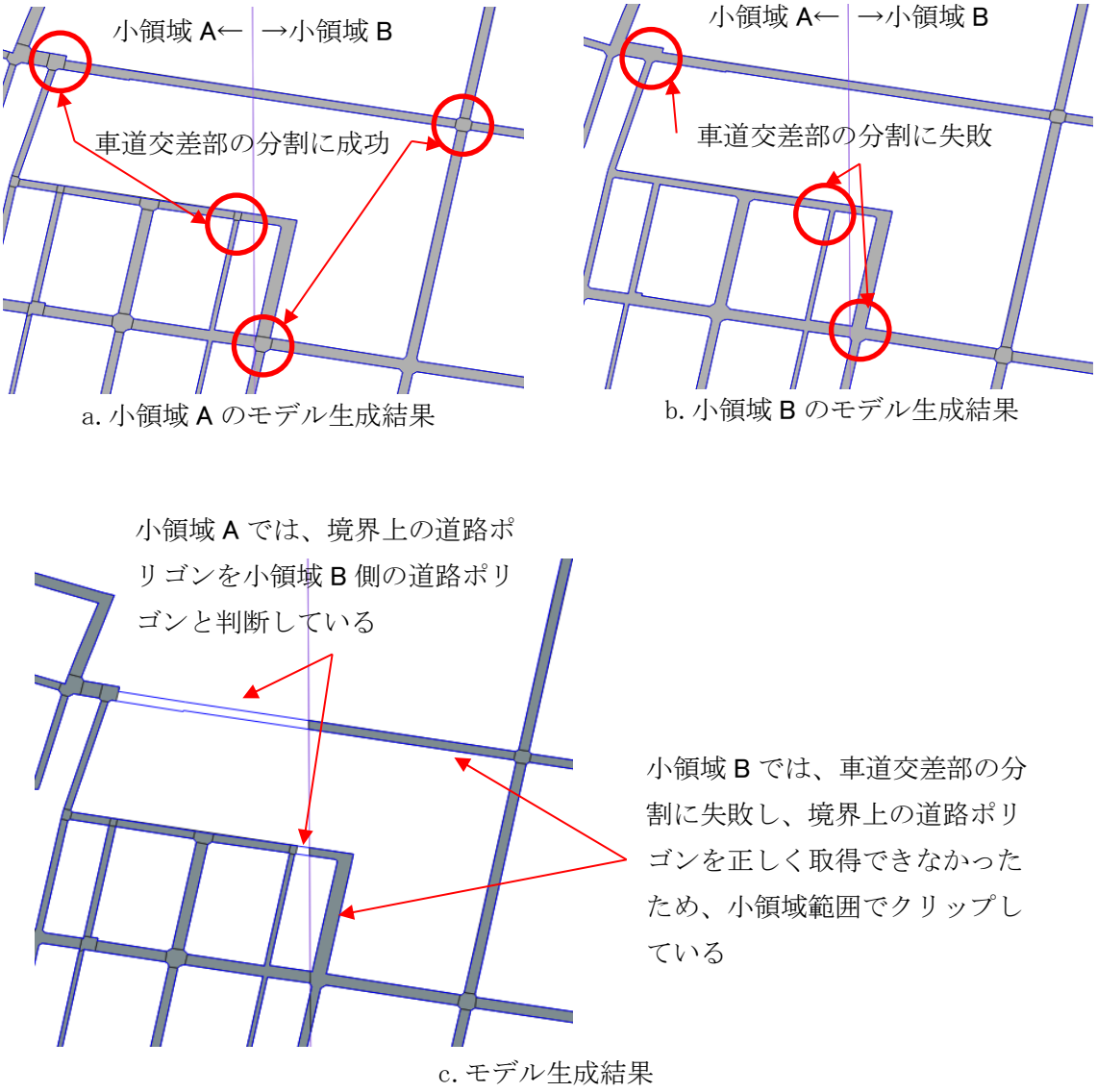


図 3-66　処理対象領域境界での道路ポリゴンの欠落原因

⑤　車道交差部の分割

道路ポリゴンの形状の正確性の検証では、大幅なずれがある道路ポリゴンが約 50%存在する。なお、評価対象地点の大半が車道交差部のため、車道交差部の分割処理を改善することで道路ポリゴンの形状の正確性が向上すると考える。

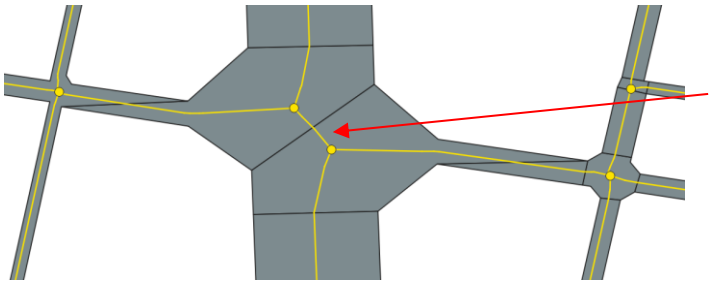
車道交差部の分割では、以下の課題が存在している。

(ア)車道交差部の統合

大きい車道交差部では、1 つの車道交差部が複数の車道交差部に分割されて道路ポリゴンが作成される場合がある。車道交差部は入力道路縁から作成した道路中心線の分岐点を基準として分

割するが、現状の道路中心線の作成方法の関係上、1 つの車道交差部に対して道路中心線の分岐が複数発生する場合がある。現状、道路中心線の分岐点が複数発生する問題の対応として、近傍距離にある分岐点を統合する処理を実装しているが、近傍分岐点と判定するための距離しきい値の設定値を超える場合は分岐点が統合されずに残存する。

この課題の対応策としては、道路中心線の分岐点の統合方法の改良又は、作成された車道交差部ポリゴンに対して隣接している車道交差部ポリゴンは統合して 1 つの車道交差部ポリゴンとする対応が必要と考える。



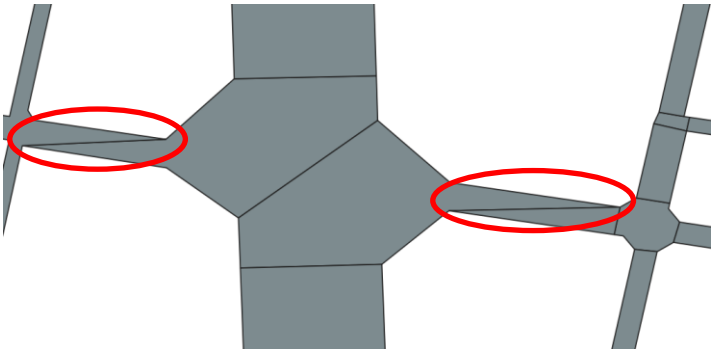
大きい交差点では、道路中心線の分岐点が複数発生する場合が存在する
分岐点ごとに車道交差部を分割するため、1 つの交差点が 2 つに分割される

青 ： 道路縁、黄 ： 道路中心線

図 3-67　車道交差部の統合課題

(イ)車道交差部の分割処理

車道交差部の分割処理において、分割線が下図のような斜線になる現象を確認している。この課題は車道交差部の分割処理に不備があるものと考え、処理の見直しを実施する必要がある。



車道交差部の分割線の形状に問題があるため、車道交差部分割処理の見直しが必要

図 3-68　車道交差部の分割処理課題

(ウ)車道交差部の補正

車道交差部の形状において、理想形状と比較して軽微なずれが発生している場合がある。現状の車道交差部の分割処理に下図のような分割線の補正処理を追加する必要がある。

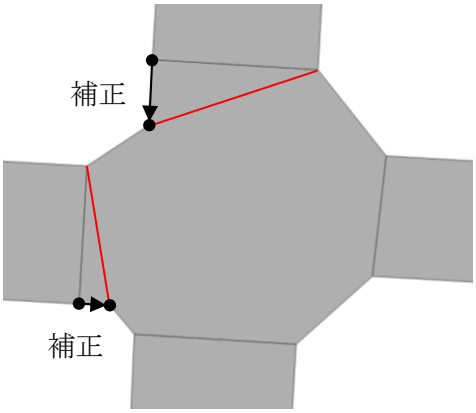
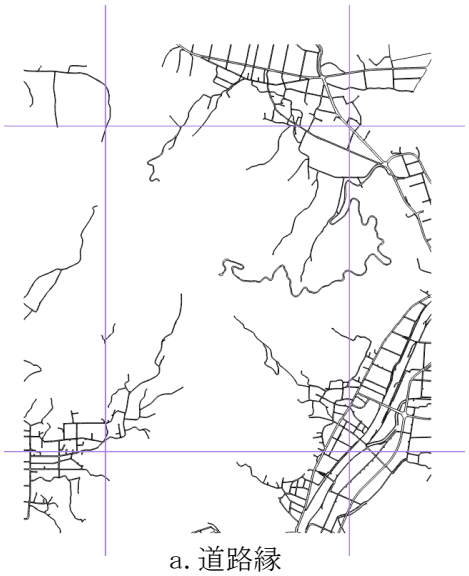


図 3-69　車道交差部の分割線の補正

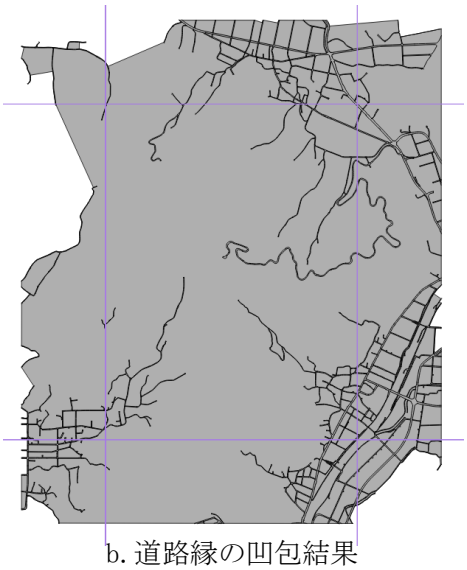
⑥　道路ポリゴンの作成

車道交差部等で分割する前の道路ポリゴンにおいて、図 3-70　a のように道路モデル生成範囲の中央付近において、道路の密度が低い場合に街区ポリゴンが作成できず、分割前道路ポリゴンの形状が実際の道路ポリゴンと大きく乖離する現象を確認している。

この課題の対策案として、道路縁をクラスタリングし、クラスタごとに道路縁の凹包結果を取得することで、道路の密度が低い地域のポリゴンを作成しないようにする方法を提案する。

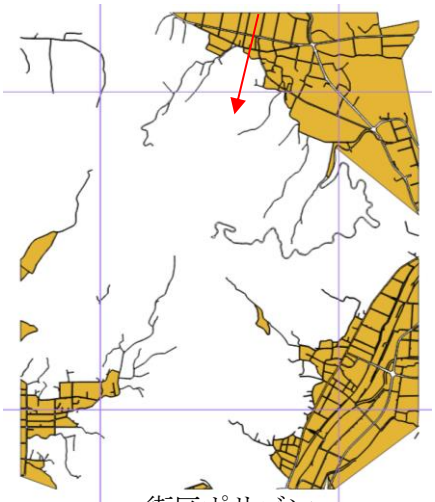


a. 道路縁

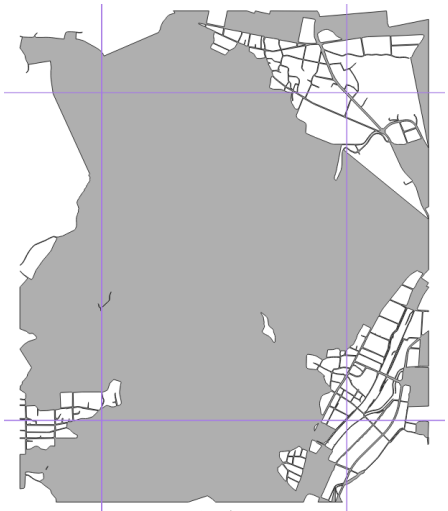


b. 道路縁の凹包結果

街区ポリゴンが作成できない部分

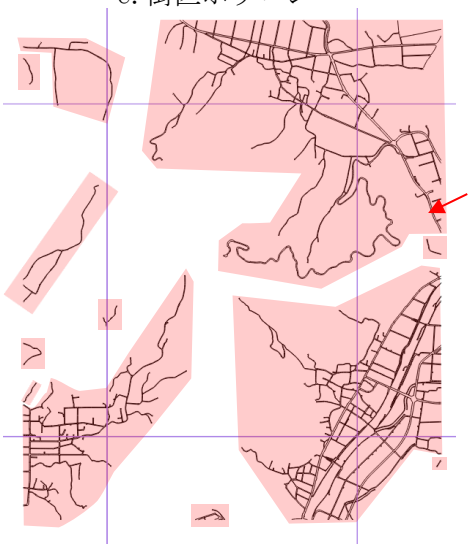


c. 街区ポリゴン



d. 分割前道路ポリゴン

(b-c のポリゴン)



クラスタ

道路縁をクラスタリングし、クラスタごとに分割前道路ポリゴンを作成する必要がある

e. 対策

図 3-70　分割前道路ポリゴン作成の課題

3.5. LOD2 道路モデル自動作成ツールの開発

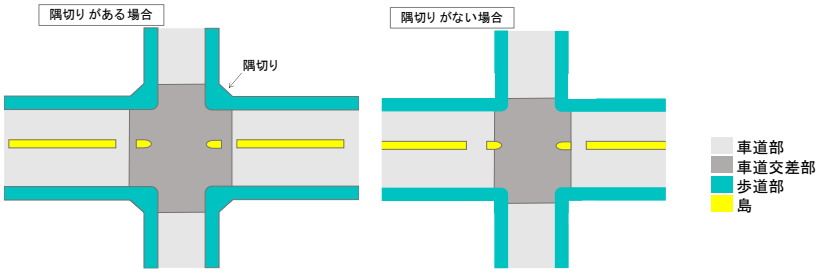
LOD2 道路モデル自動作成ツールを本章では本システムという。

3.5.1.設計概要

3.5.1.1. LOD2 道路モデルの仕様

本システムでは「3D 都市モデル標準製品仕様書」「3D 都市モデル標準作業手順書」に従い、LOD2 道路モデルを作成する。LOD2 道路モデルでは、道路の形状を面により表現し、面を車道部、車道交差部、歩道部及び島に区分する仕様となっている。

表 3-43　LOD2 道路モデルの仕様[10]

	LOD2
取得例	
説明	道路縁により囲まれた範囲を面として取得し、面を以下に区分する。 4 車道部 5 車道交差部 6 歩道部 7 島 高さは 0 とする。

3.5.1.2. 本システムの開発目的

LOD2 道路モデルにおいて、車道交差部は LOD1 で分離されているが、歩道部と島は LOD2 の作成工程での分離となる。この道路面の分離は航空写真から判読する必要があり、人手による作業が行われているため、作業のコストや期間が課題となっている。本開発において、AI を活用したツールを開発することで道路モデル作成を効率化することを目的とする。

3.5.1.3. 設計方針

- 本システムの開発、運用コストを減らすため、オープンソースソフトウェアを積極的に利用する。
- 本システムの開発成果をオープンソースソフトウェアとして Project PLATEAU GitHub に公開する。

[10] 国土交通省都市局 Project PLATEAU,
3D 都市モデル標準製品仕様書 ver.3.5（2024.3）p.160. 表 4-18 引用

3.5.1.4. 前提、制約条件

- 本システムの入力航空写真(オルソ)は、下記の制約を想定とする。
  - ワールドファイルの座標系が全て同じであること
  - 座標系が平面直角座標系であること
  - ワールドファイルの回転パラメータ(2、3 行目)が 0 であること
  - 複数のオルソ画像を入力とする場合、領域に重複がないこと
  - 画像の色は RGB 形式であること
  - 地上解像度が 12cm から 16cm であること
    （後述する AI モデルの学習には 12cm から 16cm までの地上解像度のオルソ画像が使用されていたため、この範囲外のオルソ画像を入力すると、認識精度が低下する可能性がある）

- CityGML の入出力データに関して、上記全ては PLATEAU のマニュアル(https://www.mlit.go.jp/plateau/libraries/)を満たすものとする。
  - 3D 都市モデル標準製品仕様書　V3.5
  - 3D 都市モデル標準作業仕様書　V3.5
  - 3D 都市モデル整備のための測量マニュアル　V3.0

3.5.1.5. 機能、性能

本システムの主な機能を以下に示す。

- 航空写真（オルソ）や道路ポリゴンデータ等から CityGML 形式及びシェープファイル形式の LOD2 道路モデルを出力する。
- LOD1 の道路面を車道部、車道交差部、歩道部、島に分離する。

3.5.2.AI 技術の利用

LOD2 道路モデルの作成では画像のセグメンテーションを行い、車道部、島、歩道部、オクルージョンを分離する必要がある。この分離作業には、AI 画像セマンティックセグメンテーションという技術が有用と考えられている。

AI 画像セマンティックセグメンテーションは、画素ごとに特定のカテゴリに分類する技術であり、同じカテゴリの画素は 1 つの連続領域となる。（図 3-71）

図 3-71　セマンティックセグメンテーション

出典：https://www.superannotate.com/blog/guide-to-semantic-segmentation

本章では､AI 画像セマンティックセグメンテーション用モデルの選定と学習について記載する。

3.5.2.1. AI モデルの選定

AI セマンティックセグメンテーション用モデルには、コンボルーションベースモデル（CNN）とトランスフォーマーベースモデル（Transformer）がある。

コンボルーションベースモデルの代表例としては、図 3-72 に示されている UNet がある。UNet では、Up sampleing による高解像度特徴マップの生成と Short-Cut の導入により精度向上が達成され、セマンティックセグメンテーション用モデルの基本的な構造の一つとなっている。

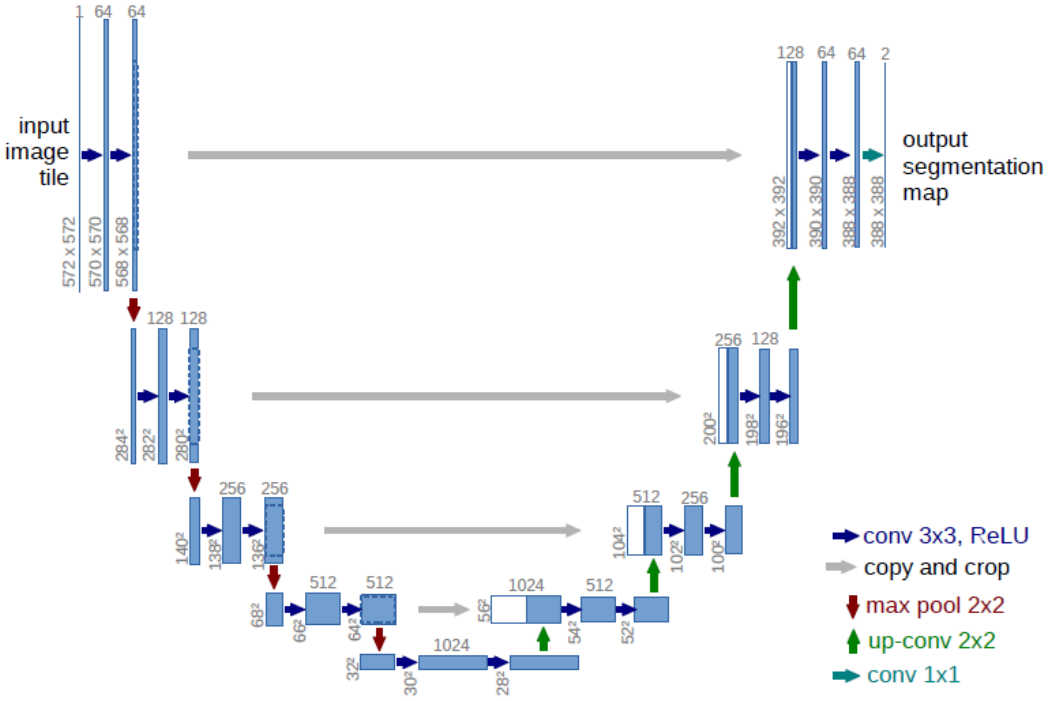


図 3-72　UNet の構成図

出典：https://arxiv.org/abs/1505.04597

近年、トランスフォーマーベースモデルやコンボルーションとトランスフォーマーを組み合わせたハイブリッドモデルが大規模なデータセットで高い認識精度を達成できることが報告されている。2022 年に発表された Mask2Former はその一つであり、その構成を図 3-73 に示す。Mask2Former は、ResNet をバックボーンとして利用し、低解像度特徴マップを抽出してから Up Sampling して特徴マップのピラミッドを生成し、さらに Transformer を用いて全域特徴と関係を抽出する。Mask2Former の改善ポイントとしては、Transformer の Decoder に Masked Attention を使用して Transformer ベース手法の収束が遅い問題を改善し、さらにマルチスケール特徴マップを使用することや最適化手法を改善することで小さいオブジェクトの認識性能を向上させた。

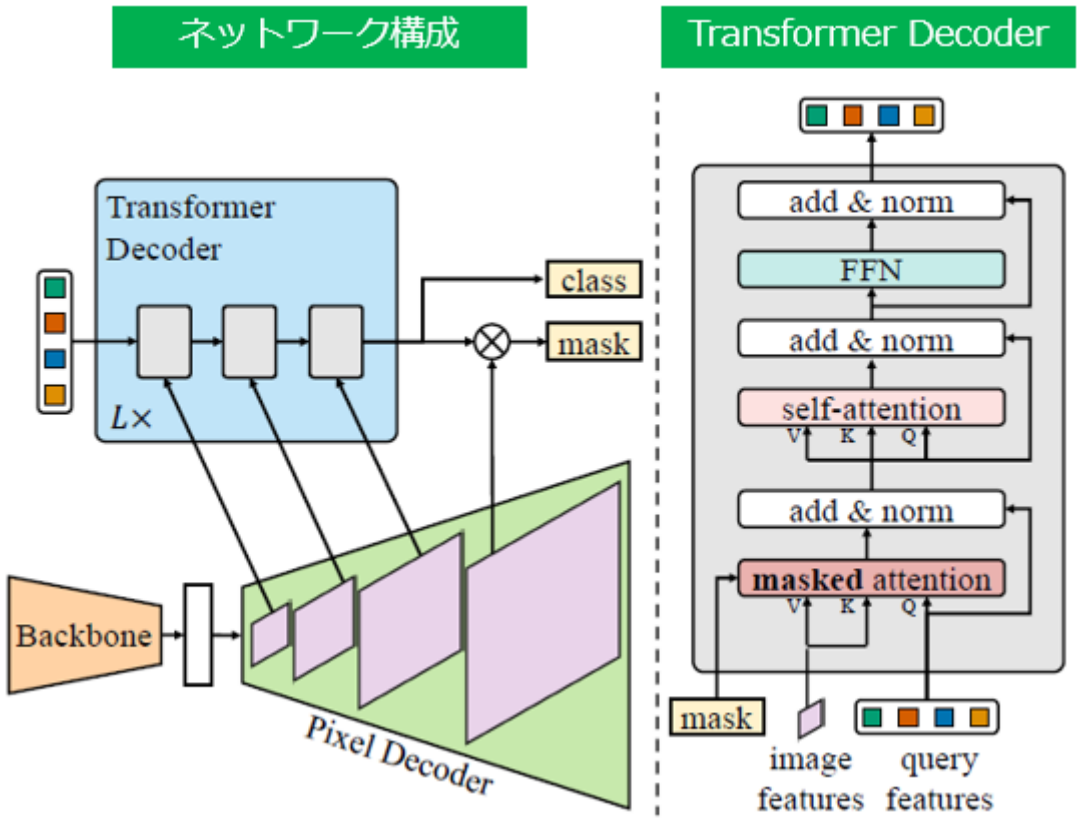


図 3-73　Mask2Former の構成図

出典：https://arxiv.org/abs/2112.01527

Mask2Former と他のモデルの比較結果を表 3-44 に示す。公開データセットの ADE20K（図 3-74 左）と Cityscapes（図 3-74 右）において Mask2Former がもっとも良い精度が得られることが示されている。そのため、本システムの AI セマンティックセグメンテーション用モデルには Mask2Former を選定した。

表 3-44　モデルの比較

モデル	モデル詳細	パラメータ数(M)	ADE20K	Cityscapes
SETR	SETR-MLA	310.57	48.64 / 50.28	
	SETR-PUP	318.31		79.34 / 82.15
Swin	Swin-L	234	53.5	
Segmenter	Seg-L/Mask	307	53.63	81.3
SegFormer	SegFormer-B5	81.4	51.0 / 51.8	82.4 / 84.0
PVT	PVT v1	65.1	44.8	
	PVT v2	85.7	48.7	
Twins	Twins-PCPVT	91.5	48.6 / 49.8	
	Twins-SVT	133	48.8 / 50.2	
DPT	DPT-Hybrid	123	49.02	
HRFormer	HRFormer-B	50.3	46.3 / 47.6	81.9 / 82.6
Mask2Former	Mask2Former-Swin-L-FaPN	(参考:200-400)	**56.4 / 57.7**	
	Mask2Former-Swin-B	(参考:200-400)		**83.3 / 84.5**

図 3-74　ADE20K と Cityscapes

出典：https://arxiv.org/abs/2305.03273

ADE20K は、セマンティックセグメンテーションのために設計された広範なシーンカテゴリーを含む画像データセットである。ADE20K は MIT のコンピュータビジョン研究グループによって作成された。このデータセットには約 20,000 枚の画像が含まれており、各画像にはオブジェクトや部屋などの細かいセグメントがラベル付けされている。主な用途は、物体認識やシーン解析、セマンティックセグメンテーションなどの機械学習タスクのためのトレーニングと評価である。

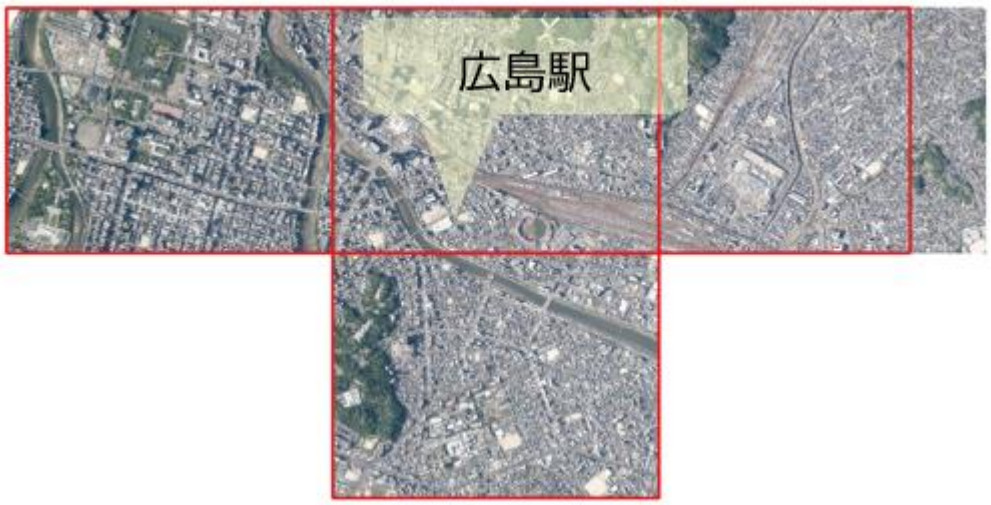
Cityscapes は、都市部のシーンを対象としたデータセットである。このデータセットは、車の前方に設置されたカメラから撮影された画像を含み、セマンティックセグメンテーションやインスタンスセグメンテーションなどのコンピュータビジョンのタスクに利用される。Cityscapes には約 5,000 枚の高解像度画像が含まれており、画像のピクセルレベルでのセマンティックラベルが付けられている。このデータセットは、自動運転や都市環境の理解など、都市部での AI 技術の開発や評価に役立つ。

3.5.2.2. AI モデルの学習

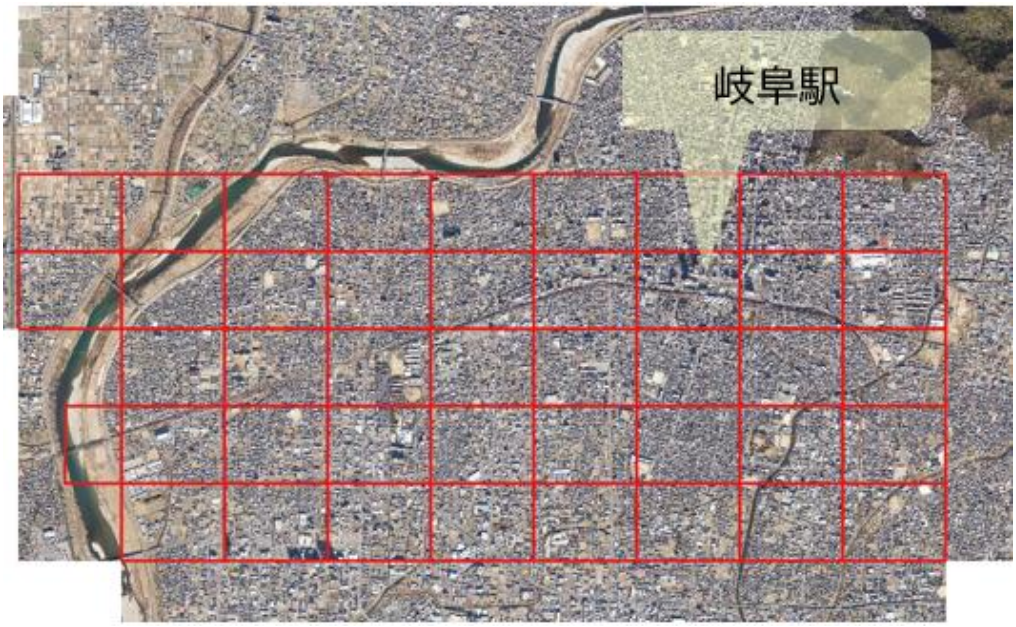
AI セマンティックセグメンテーション用モデルの学習・評価用データの取得地域は広島市と岐阜市の一部とした（表 3-45、図 3-75）。広島市では、広島駅周辺の交通網が発達している地域を選択した。また、広島市では林地が沢山含まれていることも特徴である。一方、岐阜市では、農地や山地を含むより広範囲な地域を選択した。

表 3-45　学習・評価用地域の選定

選定地域	事業者	解像度	作成面積
広島市	国際航業	16.0cm	約 2000*1500*4 ㎡
岐阜市	アジア航測	12.0cm	約 800*600*48 ㎡



(a) 広島市



(b) 岐阜市

図 3-75　学習・評価用地域の選定

教師ラベルは、オルソ画像をもとに道路の車道部、歩道部、島部、オクルージョン領域を目視で確認し、異なる色（ラベル）で塗りつぶして作成した。作成した教師ラベルの例を図 3-76 に示す。

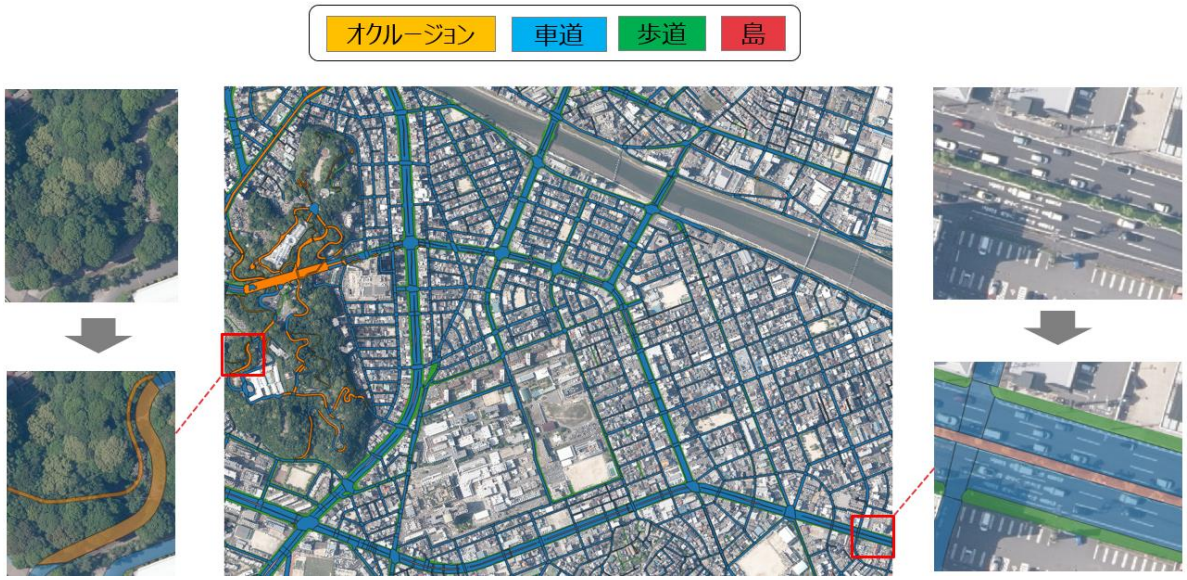


図 3-76　学習データのイメージ

作成した学習データをトレーニングデータとテストデータに分割し、トレーニングデータを使用してモデルを学習し、テストデータを使用してモデルを評価した。トレーニングデータとテストデータの分割方法については 3.6.2.1 に、評価指標について 3.6.2.2 の(1)定量評価に、そして評価結果については 3.6.2.3 の(1)定量評価結果に記載する。

3.5.3.システム設計

3.5.3.1 では本システムのシステム構成、3.5.3.2 ではシステムのインタフェース、3.5.3.3 ではハードウェア環境、実装言語と利用したライブラリを含むソフトウェア環境、3.5.3.4 では本システムのユースケースについて記載する。

3.5.3.1. システム構成

システム構成を図 3-77 に示す。

本システムは、①CityGML 入力機能、②LOD2 モデル生成機能、③CityGML 出力機能の 3 つの機能から構成される。更に②LOD2 モデル生成機能は、道路の隣接関係の判定機能、推論用データの生成機能、セグメンテーション機能、ノイズ除去機能、オクルージョンの再分類機能、ベクトル化機能のサブ機能から構成される。各機能は独立していて、入力されたデータは順番に処理される。

本システムの入力データには、道路 LOD1 CityGML（又は道路 LOD1 シェープファイル）、航空写真（オルソ画像）が含まれる。

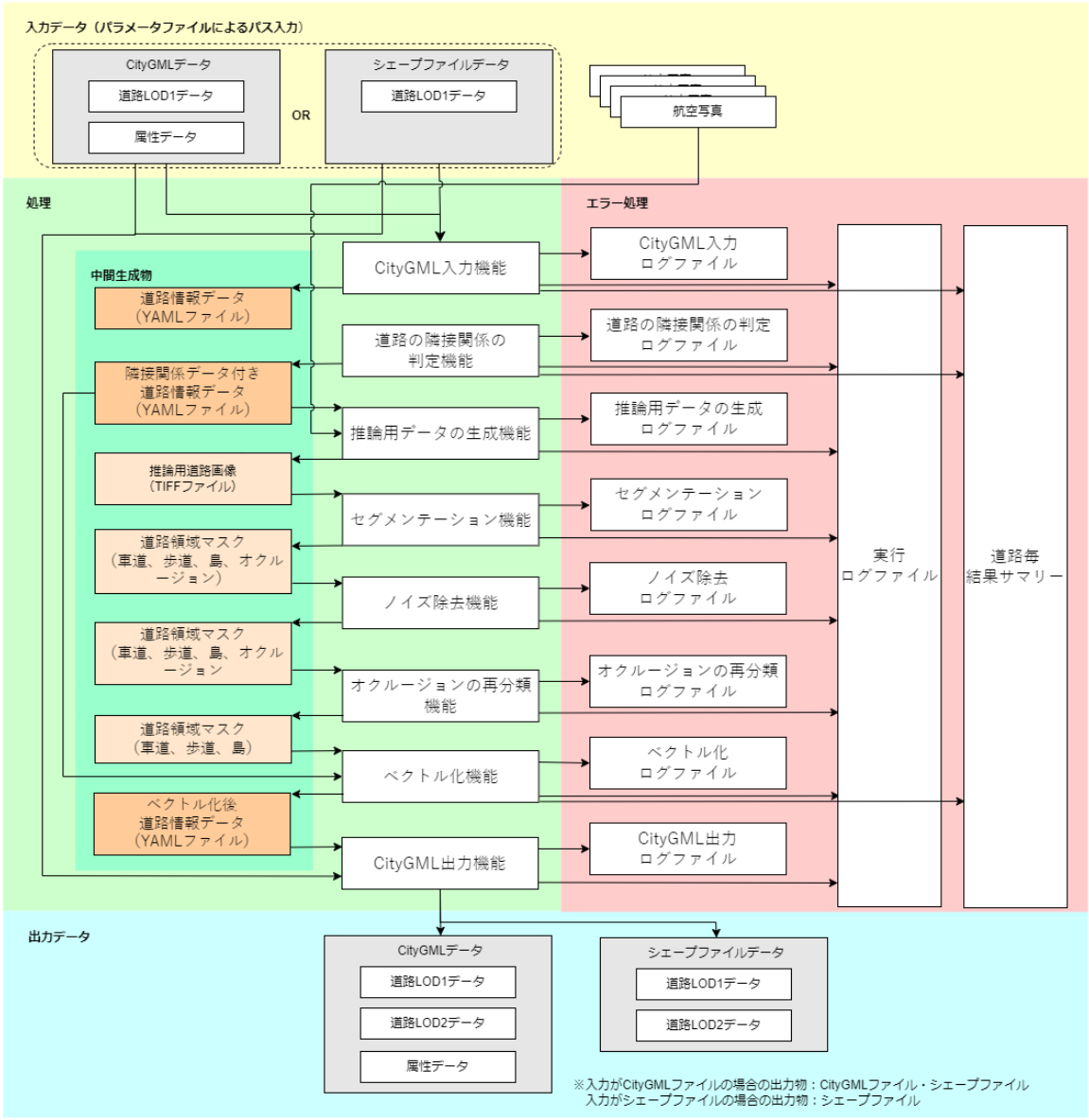


図 3-77　システム構成図

3.5.3.2. システムインタフェース

(1) 外部インタフェース

本システムの外部システムとのインタフェースについて記載する。

- 写真測量ソフト

  航空写真（原画像）から、航空写真（オルソ）を生成する。

(2) 内部インタフェース

本システムの内部モジュール間のインタフェースについて記載する。

- 道路情報データ(YAML ファイル)

  YAML ファイル形式の道路情報データで、各モジュール間のデータ授受を行う。

・ 道路領域マスク画像(TIFF ファイル)
入力した航空写真データをもとに作成した TIFF ファイル形式の道路領域マスク画像で、各モジュール間のデータ授受を行う。

実行時に出力される中間ファイル及び出力ファイルについて表 3-46 に記載する。実行時引数として渡した出力ディレクトリに以下の８ディレクトリに分けてファイルが出力される。

表 3-46　中間ファイル及び出力ファイル

ディレクトリ名	扱い	内容
road_info	中間出力	オルソ画像の図郭ごとに CityGML ファイル又はシェープファイルから抽出した道路情報が書かれている YAML ファイル
road_info_with_neighbors	中間出力	road_info ディレクトリの内容に道路の隣接情報を追記した YAML ファイル
inference_input	中間出力	推論時の入力に使用するデータ道路領域のマスク画像と LOD1 道路領域でマスクされたオルソ画像が TIFF 画像として出力される
inference_result	中間出力	道路の各領域（車道部、島、歩道部、オクルージョン）を示すマスク画像
inference_result_noiseless	中間出力	推論結果から小さい領域を消した画像ファイル
inference_result_without_occlusion	中間出力	推論結果から小さい領域を消しオクルージョンの再分類を行った、道路の各領域（車道部、島、歩道部、オクルージョン）を示すマスク画像
vectorized	中間出力	ベクトル化後のデータ YAML 形式で出力される
lod2_citygml	最終出力	ベクトル化後のシェープファイルと、生成した LOD2 の情報が追記された CityGML ファイル

(3) ユーザインタフェース

本システムの実行は、コマンド プロンプトにて下記のコマンドにて行うものとする。

```
> python AutoCreateRoadLod2.py param.json
```

AutoCreateRoadLod2.py: 本システムの Python コード

param.json: 設定パラメータファイル

本システムが表示するシステムメッセージを図 3-78 に記載する。

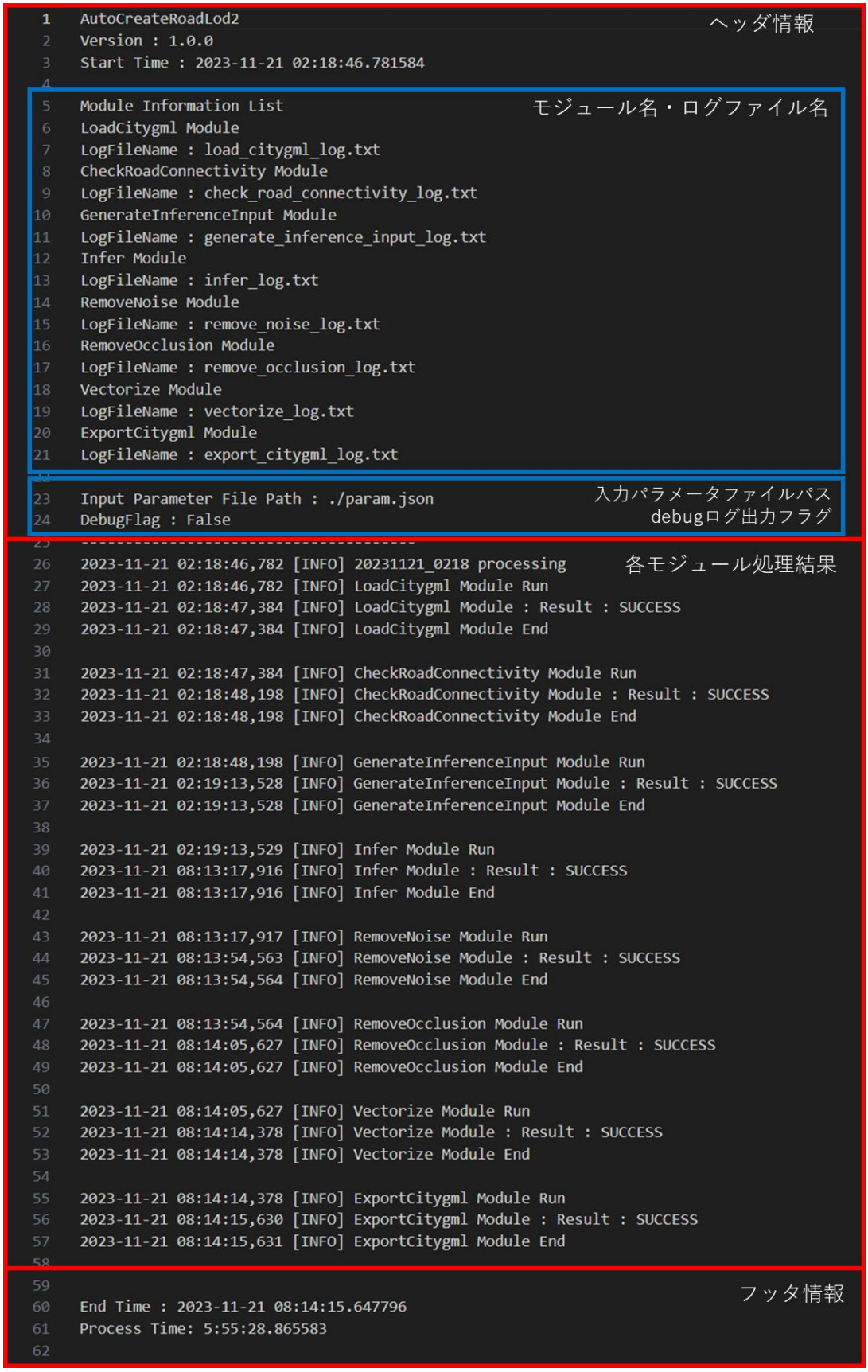


図 3-78　システムメッセージ出力例(main_log.txt、標準出力)

システムメッセージは、標準出力とファイルに出力する。

ファイルの出力先は、設定パラメータファイルの"OutputLogDir"で指定したフォルダ内の"outputlog_YYYYMMDD_HHMMSS"フォルダに出力する。ファイル構成は以下のとおりである。

〈ファイル構成例〉

LogFolder（"OutputLogDirh"で指定されたフォルダ）

└ outputlog_20221011_110120（システム実行ごとの日時フォルダ）

└ main_log.txt

3.5.3.3. 動作環境

(1) ハードウェア、OS 環境

本システムの推奨環境を以下に示す。

推奨環境:

CPU: Intel® Core™ i7 以上

Memory: 16GB 以上

GPU： NVIDIA Quadro RTX 5000 以上

GPU Memory: 16GB 以上

(2) ソフトウェア環境

本システムの使用言語は、Python(バージョン 3.9)である。

表 3-47 に使用ライブラリの一覧を示す。

表 3-47　使用ライブラリ一覧

ライブラリ名	バージョン	ライセンス	使用用途
tqdm	4.65.0	MIT License、 Mozilla Public License 2.0	プログレスバーの表示
shapely	2.0.1	BSD License (BSD 3-Clause)	幾何計算
numpy	1.25.0	BSD License (BSD-3-Clause)	数値計算
Pillow	10.0.0	Historical Permission Notice and Disclaimer	画像の読み書き
PyYAML	6.0.1	MIT License	YAML ファイルの読み書き
pyproj	3.6.0	MIT License	座標系の変換
pyshp	2.3.1	MIT License	シェープファイルの読み書き
lxml	4.9.3	BSD License (BSD-3-Clause)	XML ファイルの読み書き
torch	2.0.1	BSD License (BSD-3)	機械学習
torchvision	0.15.2	BSD	機械学習
lightning	2.0.6	Apache Software License (Apache 2.0)	機械学習
opencv-python	4.7.0.72	Apache Software License (Apache 2.0)	画像の読み書き、画像処理
opencv-contrib-python	4.8.0.74	Apache Software License (Apache 2.0)	画像処理
networkx	3.1	BSD License	グラフ構造の管理
shapelysmooth	0.1.1	Public Domain (Unlicense)	幾何のスムージング処理
geopandas	0.13.2	BSD License (BSD 3-Clause)	シェープファイルの書き出し
sortedcontainers	2.4.0	Apache Software License (Apache 2.0)	ソート済みコレクションの管理

ライブラリ名	バージョン	ライセンス	使用用途
openmim	0.3.9	Apache 2.0 license	OpenMMLab ライブラリの管理
mmcv	2.0.1	Apache Software License	機械学習
mmsegmentation	1.1.1	Apache Software License (Apache License 2.0)	機械学習
mmdet	3.1.0	Apache Software License (Apache License 2.0)	機械学習

3.5.3.4. ユースケース

本システムを用いた LOD2 道路モデル生成の手順は以下のとおりである。

(1) 入力データの準備

- 航空写真(オルソ画像)を準備する。
- 道路情報データに関しては、航空写真の撮影エリアに該当する LOD1 CityGML ファイルを用意する（例：G 空間情報センターからダウンロードする）。

(2) パラメータファイルの作成

- 本システムに入力する設定パラメータファイルを作成する。
  パラメータファイルの記載内容については、「3.5.5.1 設定パラメータ」を参照。

(3) システムの実行

- (2)で作成したパラメータファイルを指定して、本システムを実行する。
  実行方法及びシステム終了時のシステムメッセージの表示例については、「3.5.3.2 システムインタフェース」を参照。

3.5.4.モジュール設計

3.5.4.1. CityGML 入力モジュール

(1) 概要

道路 LOD1 データ入力ファイルを読み込み、本システムに必要な道路情報データを取得する。

(2) 入力データ

表 3-48 に CityGML 入力モジュールの入力データの一覧を示す。

No.2、3 に関しては No.1 の指定に従いどちらか一方のみの記載を用いる。

表 3-48 CityGML 入力モジュールの入力データの一覧

No.	データ名	入力元	説明
1	入力ファイル形式指定	設定パラメータファイル	・設定パラメータファイルの"InputType"キー が指定する入力ファイルの形式 ・「1：CityGML ファイル 2：シェープファイル」とする
2	CityGML 入力ファイルパス	設定パラメータファイル	・設定パラメータファイルの"CityGMLDir"キー が指定するフォルダ内に存在する CityGML ファイルのパス
3	シェープファイル入力ファイルパス	設定パラメータファイル	・設定パラメータファイルの"ShapeDir"キー が指定するフォルダ内に存在するシェープファイルのパス
4	航空写真（オルソ）入力ファイルパス	設定パラメータファイル	・設定パラメータファイルの"OrthoDir"キー が指定するフォルダ内に存在する航空写真オルソ画像(ワールドファイル付き)ファイルのパス
5	航空写真（オルソ）の EPSG コード	設定パラメータファイル	・設定パラメータファイルの"OrthoEPSG"キー が指定する、オルソ画像のワールドファイルに記載された座標の EPSG コード
6	CityGML の EPSG コード	設定パラメータファイル	・設定パラメータファイルの“CityGmlEPSG”キー が指定する、CityGML に記載されている座標の EPSG コード

(3) 出力データ

表 3-49 に道路 LOD1 データ入力モジュールの出力データの一覧を示す。

表 3-49　CityGML 入力モジュールの出力データの一覧

No.	データ名	出力先	説明
1	道路情報	[road_info] フォルダ	・道路の外部境界の XY 座標列と道路 ID 番号が含まれる ・ファイルフォーマットは YAML ファイル形式
2	ログメッセージ	道路LOD1データ入力ログファイル	・道路 LOD1 データ入力のログメッセージを出力する ・道路 LOD1 データ入力に失敗した際のエラーメッセージを主に出力する ・ログメッセージは、設定パラメータファイルの"OutputLogDir"で指定したフォルダ内の”outputlog_YYYYMMDD_HHMMSS “フォルダに load_citygml_log.txt として出力する ・出力例を図 3-79 に示す

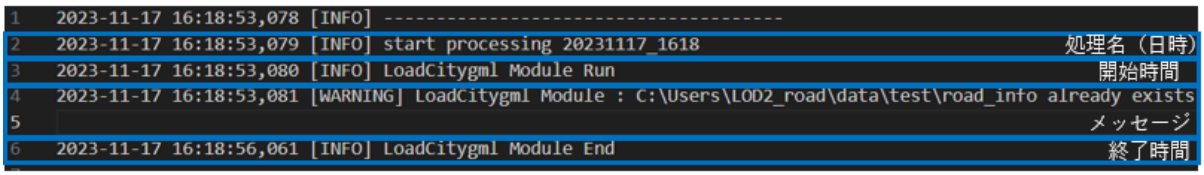


図 3-79　道路 LOD1 データ入力ログ出力例

(4) 処理内容

道路 LOD1 データ入力ファイルを読み込み、本システムに必要な道路情報データを取得する。取得する道路情報データは以下のとおりである。

- 道路 ID
  CityGML ファイルの
  "/core:CityModel/core:cityObjectMember/tran:Road[@gml:id]"タグから取得する。
- 道路の外部境界
  CityGML ファイルの
  "/core:CityModel/core:cityObjectMember/tran:Road/tran:lod1MultiSurface/gml:MultiSurface/gml:surfaceMember/gml:Polygon"タグから取得する。

(5) 例外処理

表 3-50 に道路 LOD1 データ入力モジュールの例外処理の一覧を示す。

表 3-50　道路 LOD1 データ入力モジュールの例外処理の一覧

No.	エラー名称	発生条件	処理内容
1	ファイル読み込みエラー	入力 CityGML ファイル又は入力シェープファイルが存在しない	エラーログを出力し、処理を終了する。

3.5.4.2. 道路の隣接関係の判定モジュール

(1) 概要

車道部と車道交差部を区別して出力するために、各道路が車道交差部であるかの判定を行う。

(2) 入力データ

表 3-51 に道路の隣接関係の判定ジュールの入力データを示す。

表 3-51　道路の隣接関係の判定モジュールの入力データの一覧

No.	データ名	入力元	説明
1	道路情報	[road_info]フォルダ	・道路の外部境界の XY 座標列と道路 ID 番号が含まれる ・ファイルフォーマットは YAML ファイル形式

(3) 出力データ

表 3-52 に道路の隣接関係の判定モジュールの出力データを示す。

表 3-52　道路の隣接関係の判定モジュールの出力データの一覧

No.	データ名	出力先	説明
1	道路情報（隣接関係データ付き）	[road_info_with_neighbors]フォルダ	・道路情報に隣接関係が付与された列挙データ ・ファイルフォーマットは YAML ファイル形式
2	ログメッセージ	道路の隣接関係の判定ログファイル	・道路の隣接関係の判定時のログメッセージを出力する ・道路の隣接関係の判定に失敗した際のエラーメッセージを主に出力する ・ログメッセージは、設定パラメータファイルの"OutputLogDir"で指定したフォルダ内の"outputlog_YYYYMMDD_HHMMSS"フォルダに model_element_generation_log.txt として出力する ・出力例を図 3-80 に示す

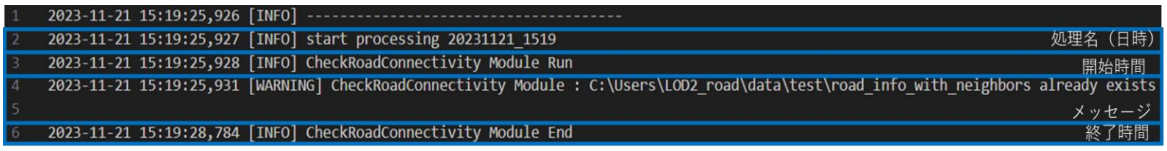


図 3-80　道路の隣接関係の判定/ログ出力例

(4) 処理内容

2 つの道路の隣接は、外部境界の少なくとも 1 つの辺の両端点の座標が一致することと定義して処理を行う。各道路の外部境界の辺ごとに他の道路で同じ位置の辺を持つ道路が存在するかを調べ、同じ辺を持つ道路同士は隣接していると判断することで隣接関係を列挙する。求められた隣接関係は CityGML の読み込みで出力した YAML ファイルに追記する。

(5) 例外処理

表 3-53 に道路の隣接関係の判定モジュールの例外処理の一覧を示す。

表 3-53　道路の隣接関係の判定モジュールの例外処理の一覧

No.	エラー名称	発生条件	処理内容
1	ファイル読み込みエラー	入力ファイル読み込みに失敗	エラーログを出力し、モジュールの処理を終了する。

3.5.4.3. 推論用データの生成モジュール

(1) 概要

道路の図形情報とオルソ画像から、次工程の道路セグメンテーションの推論時に使用する画像データを作成する。

(2) 入力データ

表 3-54 に推論用データの生成モジュールの入力データの一覧を示す。

表 3-54　推論用データの生成モジュールの入力データの一覧

No.	データ名	入力元	説明
1	道路の隣接関連データ	[road_info_with_neighbors] フォルダ	・道路の隣接関係の列挙データ ・ファイルフォーマットは YAML ファイル形式
2	航空写真（オルソ）入力ファイルパス	設定パラメータファイル	・設定パラメータファイルの"OrthoFolderPath"キー が指定するフォルダ内に存在する航空写真オルソ画像(ワールドファイル付き)ファイルのパス

(3) 出力データ

表 3-55 に推論用データの生成モジュールの出力データの一覧を示す。

表 3-55　推論用データの生成モジュールの出力データの一覧

No.	データ名	出力先	説明
1	推論用道路画像	[inference_input] フォルダ	・画像内の道路の領域を示すマスク画像データ ・ファイルフォーマットは TIFF ファイル形式
2	ログメッセージ	推論用データの生成ログファイル	・推論用データの生成のログメッセージを出力する ・ログメッセージは、設定パラメータファイルの"OutputLogDir"で指定したフォルダ内の"outputlog_YYYYMMDD_HHMMSS"フォルダに check_road_connectivity_log.txt として出力する ・記述例を図 3-81 に示す

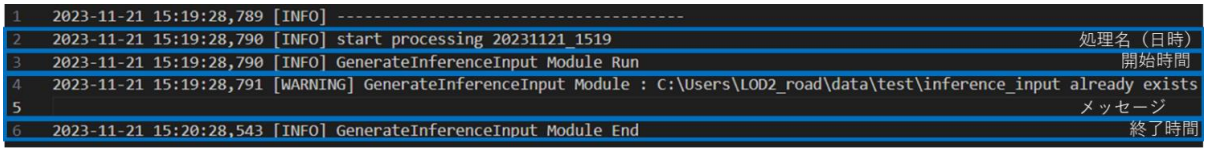


図 3-81　推論用データの生成ログ出力例

(4) 処理内容

本処理では、道路の図形情報とオルソ画像から、推論時に使用するデータを作成する。 ツールの 3 ステップ目がこの処理にあたる。

この処理は、以下のステップからなる。

1. 道路の座標をオルソ画像の UV 座標に変換し、画像内の道路の領域を示すマスク画像を作成する
2. オルソ画像の道路外の領域を黒く塗りつぶす

① 画像内の道路の領域を示すマスク画像を作成

まず、道路ポリゴンの座標を画像の UV 座標に変換する。

オルソ画像のワールドファイルの座標系での座標を(x, y)としたとき、 画像の UV 座標(u, v)は、

$$u = \frac{x - C}{A}$$
$$v = \frac{y - F}{E}$$

で求まる。ただし、ワールドファイルの値を 1 行目から、A, D, B, E, C, F とし、D = 0, B = 0 とする。

その後、オルソ画像と同じ大きさの画像にマスク領域を描画する。その例を図 3-82 に示す。このマスク画像は、道路外の領域を黒く塗りつぶす処理、推論結果から道路上のもののみを抽出する処理に使用する。

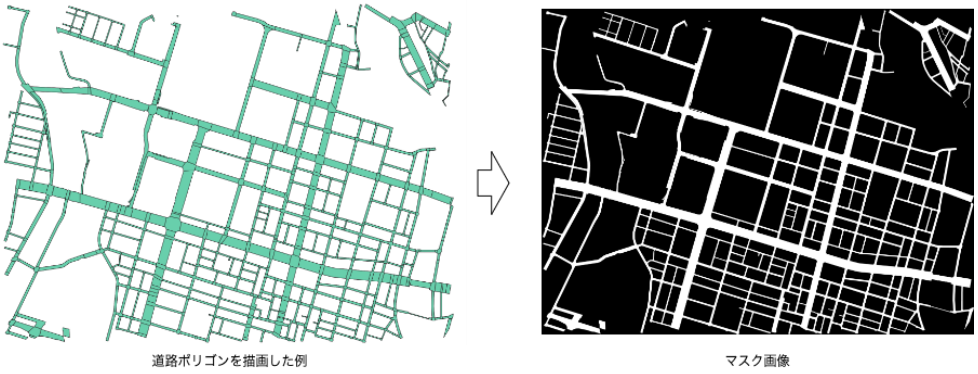


図 3-82 マスク画像生成例

② オルソ画像の道路外の領域を黒く塗りつぶす

作成したマスク画像を使用して道路外を黒く塗りつぶす。その例を図 3-83 に示す。

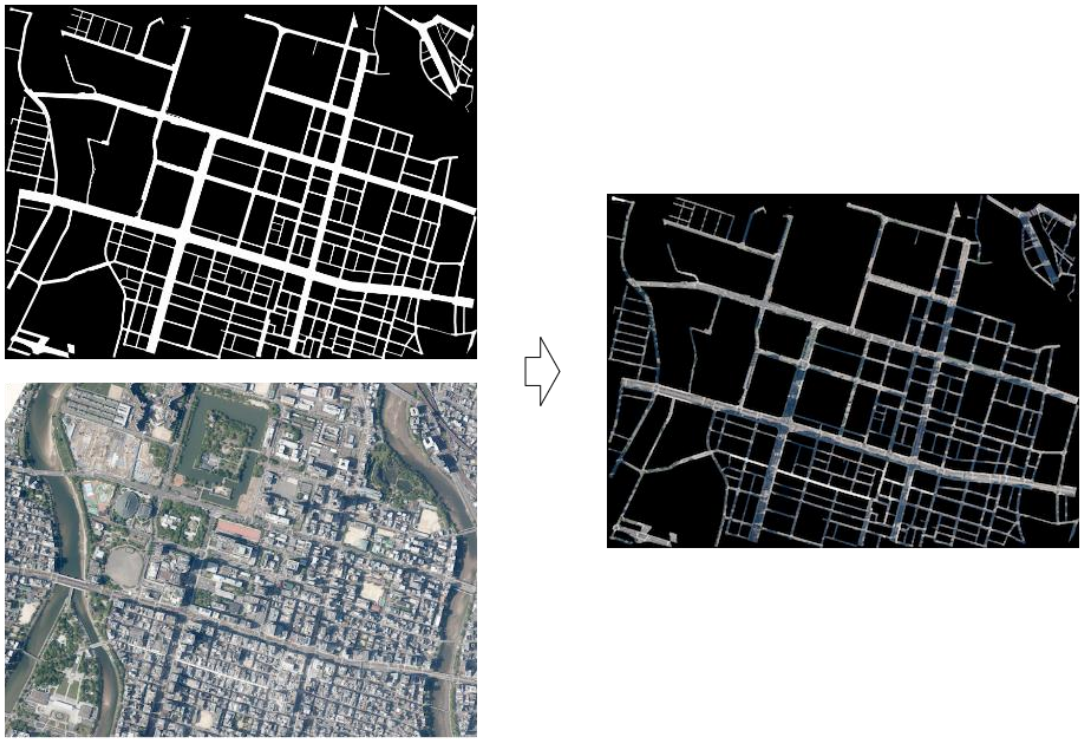


図 3-83　道路外の黒塗りの例

(5) 例外処理

表 3-56 に推論用データの生成モジュールの例外処理の一覧を示す。

表 3-56　推論用データの生成モジュールの例外処理の一覧

No.	エラー名称	発生条件	処理内容
1	ファイル読み込みエラー	入力ファイル読み込みに失敗	エラーログを出力し、モジュールの処理を終了する。

3.5.4.4. セグメンテーションモジュール

(1) 概要

道路の領域に対し、機械学習モデルを使用して車道部(車道交差部を含む)、歩道部、島、オクルージョンの 4 クラスに分類する。

(2) 入力データ

表 3-57 にセグメンテーションモジュールの入力データの一覧を示す。

表 3-57　セグメンテーションモジュールの入力データの一覧

No.	データ名	入力元	説明
1	推論用道路画像	[inference_input]フォルダ	・画像内の道路の領域を示すマスク画像データ ・ファイルフォーマットはTIFFファイル形式
2	【任意】推論を実行するデバイス	設定パラメータファイル	・設定パラメータファイルの"Device"キー が指定する推論を実行するデバイス ・CPU のみの環境では[cpu]を指定する ・デフォルトは[cuda]
3	【任意】推論時のタイルの 1 辺のサイズ	設定パラメータファイル	・設定パラメータファイルの"InferenceSize"キー が指定する、推論を行う際のタイルの 1 辺のサイズ(pixel) ・デフォルトは[256]
4	【任意】推論時のタイルで推論結果を使用しない外側の幅	設定パラメータファイル	・設定パラメータファイルの"InferencePadding"キー が指定する、推論を行う際のタイルで推論結果を使用しない外側の幅(pixel) ・デフォルトは[64]

(3) 出力データ

表 3-58 にセグメンテーションモジュールの出力データの一覧を示す。

表 3-58　セグメンテーションモジュールの出力データの一覧

No.	データ名	出力先	説明
1	セグメンテーション後の道路領域マスク	[inference_result]フォルダ	・道路領域マスク画像を車道部、歩道部、島、オクルージョンに分類したデータ

			・ファイルフォーマットは TIFF ファイル形式
2	ログメッセージ	セグメンテーションログファイル	・セグメンテーションのログメッセージを出力する ・セグメンテーションに失敗した際のエラーメッセージを主に出力する ・ログメッセージは、設定パラメータファイルの"OutputLogDir"で指定したフォルダ内の"outputlog_YYYYMMDD_HHMMSS"フォルダに infer_log.txt として出力する ・出力例を図 3-84 に示す

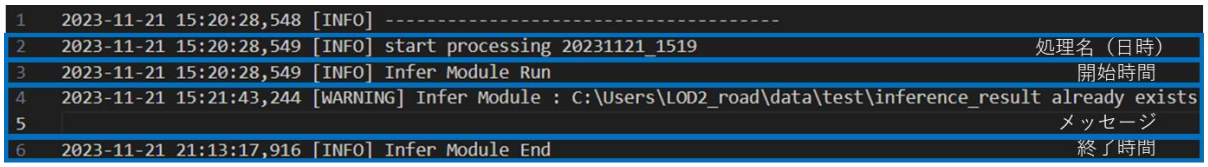


図 3-84　セグメンテーションモジュールログ出力例

(4) 処理内容

本処理では、道路の領域に対し機械学習モデルを使用して車道部(車道交差部を含む)、歩道部、島、オクルージョンの 4 クラスに分類する。ツールの 4 ステップ目がこの処理にあたる。

推論には、セグメンテーションモデルである Mask2Former を使用する。

推論は、図郭全体をまとめて入力として与えた場合、GPU の使用メモリが多く 16GB VRAM の GPU 上でも実行することはできなかったため、入力前に細かく切り分け、切り分けたタイルごとに推論し、推論結果を再度つなぎ合わせる形で図郭全体の推論を行う。この際、画像の一部がオーバーラップするように切り取り、切り取ったタイルの中心部分のみ使用する。その処理例を図 3-85 に示す。

なお、タイルの大きさや、余白の大きさは、実行時のオプションで変更することができる。

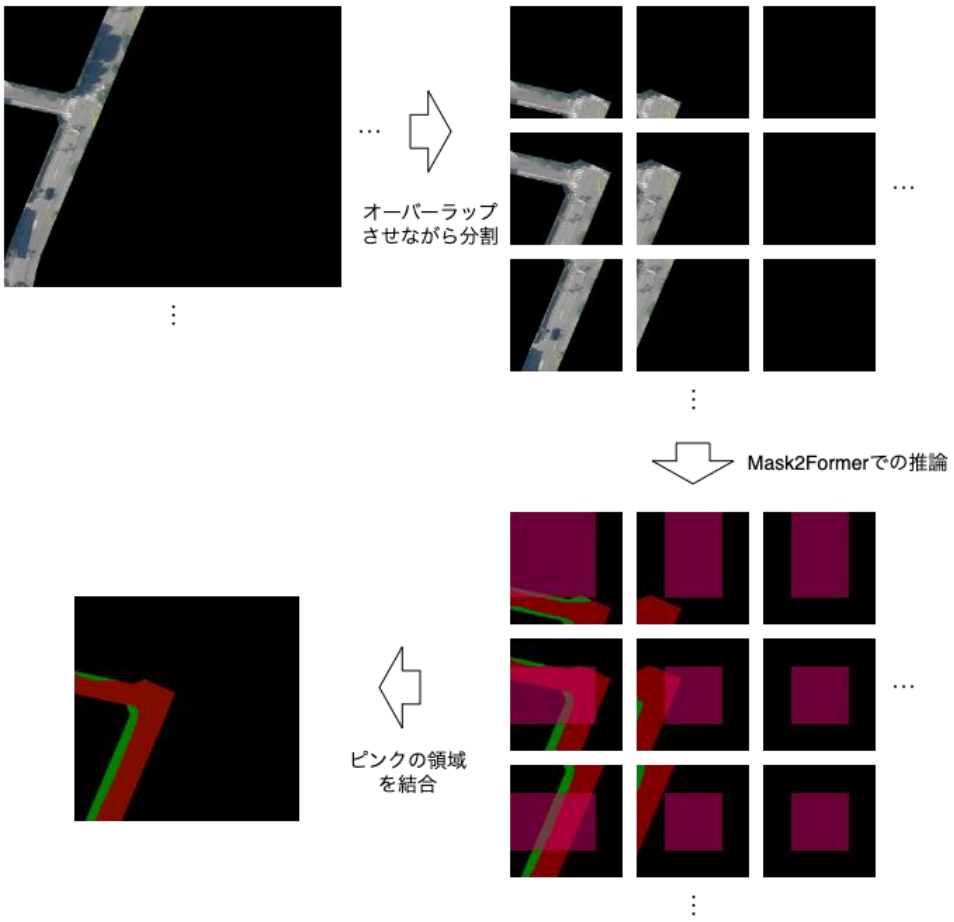


図 3-85　推論処理の例

(5) 例外処理

表 3-59 にセグメンテーションモジュールの例外処理の一覧を示す。

表 3-59　セグメンテーションモジュールの例外処理の一覧

No.	エラー名称	発生条件	処理内容
1	ファイル読み込みエラー	入力ファイル読み込みに失敗	エラーログを出力し、モジュールの処理を終了する。

3.5.4.5. ノイズ除去モジュール

(1) 概要

セマンティックセグメンテーションの結果から小さい領域(ノイズ)を除去する。

(2) 入力データ

ノイズ除去モジュールの入力データの一覧を表 3-60 に示す。

表 3-60　ノイズ除去モジュールの入力データの一覧

No.	データ名	入力元	説明
1	セグメンテーション後の道路領域マスク	[inference_result]フォルダ	・道路領域マスク画像を車道部、歩道部、島、オクルージョンに分類したデータ ・ファイルフォーマットはTIFFファイル形式
2	【任意】小さな領域の除去で除去する領域のピクセル数のしきい値	設定パラメータファイル	・設定パラメータファイルの"NoiseThreshold"キー が指定する、小さな領域の除去で除去する領域のピクセル数のしきい値 ・デフォルトは[200]

(3) 出力データ

ノイズ除去モジュールの出力データの一覧を表 3-61 に示す。

表 3-61　ノイズ除去モジュールの出力データの一覧

No.	データ名	出力先	説明
1	ノイズ除去後の道路領域マスク	[inference_result_noiseless]フォルダ	・道路領域マスク画像（車道部、歩道部、島、オクルージョン）からノイズを除去したデータ ・ファイルフォーマットは TIFF ファイル形式
2	ログメッセージ	ノイズ除去ログファイル	・ノイズ除去のログメッセージを出力する ・ノイズ除去に失敗した際のエラーメッセージを主に出力する ・ログメッセージは、設定パラメータファイルの"OutputLogDir"で指定したフォルダ内の"outputlog_YYYYMMDD_HHMMSS"フォルダに remove_noise_log.txt として出力する ・出力例を図 3-86 に示す

```
1  2023-11-16 13:32:35,475 [INFO] ------------------------------------
2  2023-11-16 13:32:35,475 [INFO] start processing 51324357_tran_6668_op.gml        処理名（日時）
3  2023-11-16 13:32:35,475 [INFO] RemoveNoise Module Run                           開始時間
4  2023-11-16 13:32:35,931 [WARNING] RemoveNoise Module : C:\Users\LOD2_road\data\test\remove_noise already exists
5                                                                                   メッセージ
6  2023-11-16 13:33:13,061 [INFO] RemoveNoise Module End                           終了時間
```

図 3-86　ノイズ除去ログ出力例

(4) 処理内容

本処理では、セマンティックセグメンテーションの結果から小さい領域(ノイズ)を除去する。ツールの 5 ステップ目がこの処理にあたる。

小さい領域が存在する状態では、道路の区分け数が増加することや、ベクトル化の処理で悪影響を及ぼす可能性があるため、現実に存在する可能性がないと考えられるほど小さな領域を除去し、その領域を周囲のラベルに置き変える処理を行う。

なお、本業務では小さい領域のしきい値を 200 ピクセル(解像度 16cm の場合で 5.12 m^2) としている。この値は実行時のオプションで変更することができる。ノイズの例を図 3-87 に示す左図の黄色枠内にノイズが存在し、周辺のラベルに置き換えをしている。

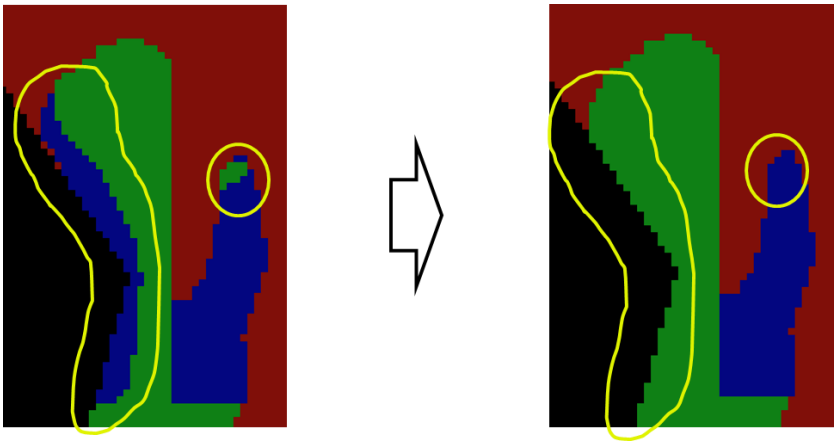


図 3-87　ノイズ除去前（左図）と除去後（右図）

ノイズ除去の処理は、以下のステップからなる。

1. 4 方向連結でピクセル数がしきい値未満の領域を抽出する
2. 各ノイズの周上のラベルを求める
3. 複数のノイズが連結している場合があるため、再度 4 方向連結のグルーピングを行う
4. ノイズ領域にラベルを連結成分ごとに割り当てる

なお、各ノイズの周上のラベルを求める処理は、高速に処理を行うため、厳密な個数は求めていない。

① しきい値未満の領域を抽出

画像の連結成分ごとの位置と面積は、OpenCV[11] の connectedComponentsWithStats 関数を使用することで求めることができる。

connectedComponentsWithStats は 2 値化された画像にのみ対応している。そのため、車道の連結成分を求める場合には、車道を 255, 車道以外を 0 とした画像に対してこの処理を行う。車道以外の歩道、島、オクルージョンも同じように処理を行うことで、全てのラベルの連結成分の面積と位置を求めることができる。

上記の処理で求められた連結成分のうち、しきい値(デフォルトは 200 ピクセル)未満の面積のものをノイズ領域とする。なお、この処理で 8 方向連結ではなく、4 方向連結を使用している。これは、ベクトル化の処理の際に 4 方向の連結性で区切られるためである。

② 各ノイズ領域の周上のラベルを求める

ノイズ領域に割り当てるラベルを決定するために、まず周上のラベルを求める。ただし、厳密に各領域の周上を数えるのは処理に時間が掛かるため、上下左右に 1 ピクセルずらしたものをマスク画像で切り出し、上下左右の結果を組み合わせることで数える。その例を示す。なお、同じ位置のラベルについては、右、左、下、上の優先順位で適用される。

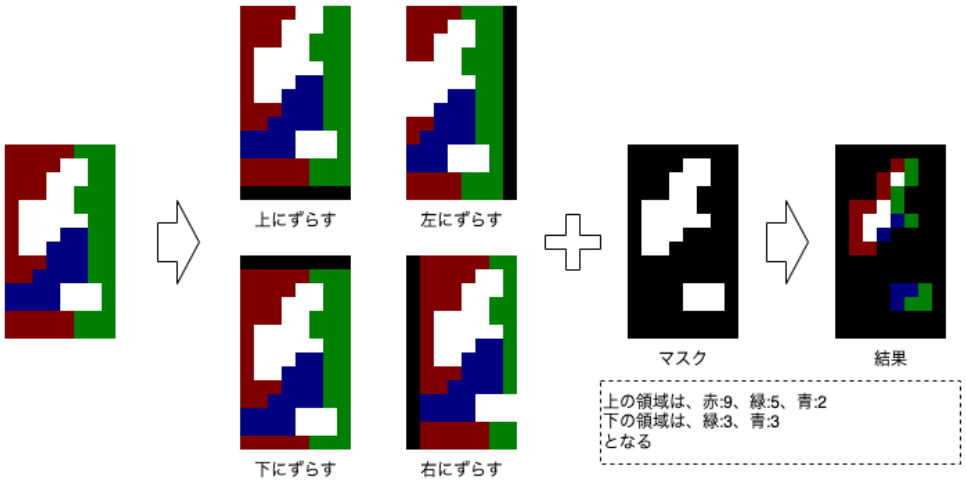


図 3-88 ノイズラベルの算出

③ ノイズ領域のグルーピング

隣接しているノイズ領域を 1 つにまとめるために、再度 OpenCV の connectedComponentsWithStats 関数を使用して、ノイズ領域の連結成分を求める。この処理は、複数のノイズ領域が隣接していることで周囲にノイズ以外のピクセルが存在しないノイズ領域が発生する可能性があるためである。

ここで同じ連結成分上とみなされた領域は以降の処理で同じラベルが割り当てられることになる。

[11] https://opencv.org

④　ノイズ領域にラベルを連結成分ごとに割り当てる

ノイズ領域の連結成分ごとに、前のステップで求めた周上のラベルの最頻値を求め、連結成分内の全てのピクセルに適用する。なお、同数の場合は、車道部、歩道部、島、オクルージョンの順に優先する。

なお、道路自体が小さいことで、ノイズ領域のみの孤立した領域が存在する可能性がある。その場合は、その領域の元のラベルの最頻値を割り当てる。

(5) 例外処理

ノイズ除去モジュールの例外処理の一覧を表 3-62 に示す。

表 3-62　ノイズ除去モジュールの例外処理の一覧

No.	エラー名称	発生条件	処理内容
1	ファイル読み込みエラー	入力ファイル読み込みに失敗	エラーログを出力し、モジュールの処理を終了する。

3.5.4.6. オクルージョンの再分類モジュール

(1) 概要

セマンティックセグメンテーションでオクルージョンと分類された領域を、車道部、歩道部、島に機械的に再分類する。

(2) 入力データ

オクルージョンの再分類モジュールの入力データの一覧を表 3-63 に示す。

表 3-63　オクルージョンの再分類モジュールの入力データの一覧

No.	データ名	入力元	説明
1	ノイズ除去後の道路領域マスク	[inference_result_noiseless] フォルダ	・道路領域マスク画像（車道部、歩道部、島、オクルージョン）からノイズを除去したデータ ・ファイルフォーマットは TIFF ファイル形式

(3) 出力データ

オクルージョンの再分類モジュールの出力データの一覧を表 3-64 に示す。

表 3-64　オクルージョンの再分類モジュールの出力データの一覧

No.	データ名	出力先	説明
1	オクルージョンの再分類後の道路領域マスク	[inference_result_without_occlusion] フォルダ	・道路領域マスク画像（車道部、歩道部、島、オクルージョン）のうち、オクルージョンを再分類したデータ ・ファイルフォーマットは TIFF ファイル形式
2	ログメッセージ	オクルージョンの再分類ログファイル	・オクルージョンの再分類のログメッセージを出力する ・オクルージョンの再分類に失敗した際のエラーメッセージを主に出力する ・ログメッセージは、設定パラメータファイルの"OutputLogDir"で指定したフォルダ内の"outputlog_YYYYMMDD_HHMMSS"フォルダに remove_occlusion_log.txt として出力する ・出力例を図 3-89 に示す

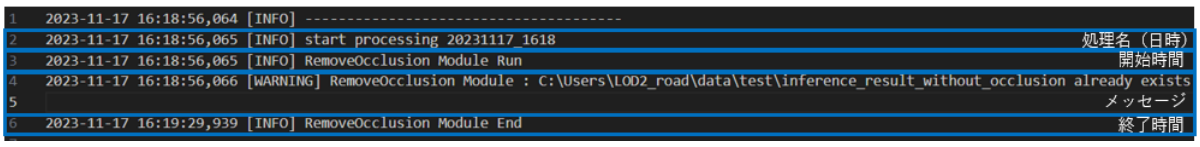


図 3-89　オクルージョンの再分類ログ出力例

(4) 処理内容

本処理では、セマンティックセグメンテーションでオクルージョンと分類された領域を、車道部、歩道部、島に機械的に再分類する。ツールの 6 ステップ目がこの処理にあたる。

オクルージョンの領域の再分類は、オクルージョンの周囲の分類が正しいという前提で処理を行う。オクルージョンの再分類は以下のステップで行う。

1. 道路縁のピクセルからの距離を求める
2. オクルージョン領域の再分類
3. 残ったオクルージョン領域のラベル付け

① 道路縁のピクセルからの距離を求める

道路内の各ピクセルについて、最も近い道路縁のピクセルからの距離を求める。この処理は OpenCV の `distanceTransform` 関数を使用して行う。

② オクルージョン領域のラベル付け

オクルージョンの周囲のピクセルからオクルージョン内を探索する要領でオクルージョン領域のラベル付けを行う。

オクルージョン領域にラベルを適用する際には、適用先のピクセルと、その適用するラベルの元となったピクセルと比較して score を算出し、score が小さいものを採用する手法を採っている。score は次の式で算出する。なお道のりの重み w は 0.1 としている。

$$\text{score} = \left| \text{道路縁からの距離}_{\text{適用元}} - \text{道路縁からの距離}_{\text{適用先}} \right| + \textit{反映元から反映先の道のり} \cdot w$$

道路外からの距離の差については、歩道部や島の領域は道路縁からの距離に依存していることが多いため採用している。また、反映元から反映先の道のりについては、使用しない場合に図 3-90 のようにオクルージョン領域の両側からの適用により縞模様になるケースが発生したため、これを防ぐために採用している。

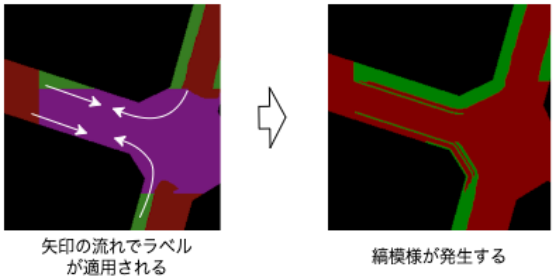


図 3-90　オクルージョン再分類において、道のりを score に含めない場合の縞模様発生ケース

本処理の詳しい流れは以下のとおりである。

1. オクルージョン領域に隣接する非オクルージョンのピクセルを列挙する。

2. 1 で列挙した非オクルージョンのピクセルごとに、隣接しているオクルージョンピクセルにそのピクセルを適用する際の score を求め、score、適用元の座標、適用先の座標、の組を優先度付きキューに追加する

3. 優先度付きキューが空になるまで以下の処理を繰り返す

（ア）優先度付きキューで最も score が小さい要素を取り出す

（イ）現時点で、適用先の座標がオクルージョンの場合、適用元座標と同じラベルに置き変える

（ウ）置き変えた場合、隣接しているオクルージョンピクセルに今回使用した適用元ピクセルを適用する際の score を求め、score、適用元の座標、適用先の座標、の組を優先度付きキューに追加する

③ 残ったオクルージョン領域のラベル付け

前ステップの処理では、オクルージョン領域に隣接しているピクセルを参照してオクルージョン領域の再分類を行っている。そのため、 連結している道路の全ての領域がオクルージョン領域の場合は、再分類を行うことができない。そのような場合には車道部を割り当てる。

(5) 例外処理

オクルージョンの再分類モジュールの例外処理の一覧を表 3-65 に示す。

表 3-65 オクルージョンの再分類モジュールの例外処理の一覧

No.	エラー名称	発生条件	処理内容
1	ファイル読み込みエラー	入力ファイル読み込みに失敗	エラーログを出力し、モジュールの処理を終了する。

3.5.4.7. ベクトル化モジュール

(1) 概要

本処理では、ラスタデータであるセマンティックセグメンテーションの結果と、ベクターデータの道路の座標を用いて、車道部、歩道部、島の領域に区分けされたベクター形式のデータを作成する。

(2) 入力データ

ベクトル化モジュールの入力データの一覧を表 3-66 に示す。

表 3-66　ベクトル化モジュールの入力データの一覧

No.	データ名	入力元	説明
1	道路情報（隣接関係データ付き）	[road_info_with_neighbors]フォルダ	・道路情報に隣接関係が付与された列挙データ ・ファイルフォーマットは YAML ファイル形式
2	オクルージョンの再分類後の道路領域マスク	[inference_result_without_occlusion]フォルダ	・道路領域マスク画像（車道部、歩道部、島、オクルージョン）のうちオクルージョンを再分類したデータ ・ファイルフォーマットは TIFF ファイル形式
3	航空写真（オルソ）の EPSG コード	設定パラメータファイル	・設定パラメータファイルの"OrthoEPSG"キー が指定する、オルソ画像のワールドファイルに記載された座標の EPSG コード

(3) 出力データ

ベクトル化モジュールの出力データの一覧を表 3-67 に示す。

表 3-67　ベクトル化モジュールの出力データの一覧

No.	データ名	出力先	説明
1	道路情報（ベクトル化後データ） YAML ファイル	[vectorized]フォルダ	・道路情報をベクトル化したデータ ・ファイルフォーマットは YAML ファイル形式
2	道路情報（ベクトル化後データ）シェープファイル	[result_output]フォルダ	・道路情報をベクトル化したデータ ・ファイルフォーマットはシェープファイル形式

3	ログメッセージ	CityGML 出力ログファイル	・CityGML 出力のログメッセージを出力する ・CityGML 出力に失敗した際のエラーメッセージを主に出力する ・ログメッセージは、設定パラメータファイルの"OutputLogDir"で指定したフォルダ内の"outputlog_YYYYMMDD_HHMMSS"フォルダに vectrize_log.txt として出力する ・出力例を図 3-91 に示す

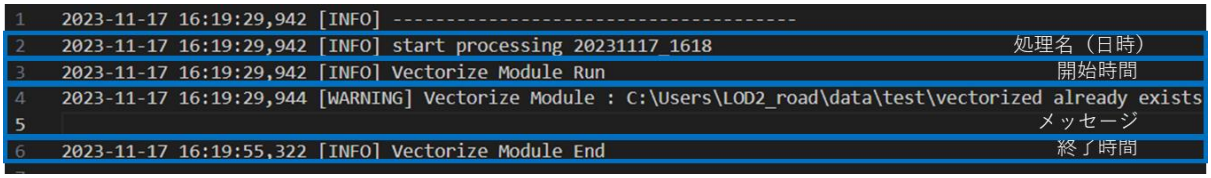


図 3-91　ベクトル化ログ出力例

(4) 処理内容

本処理では、ラスタデータであるセマンティックセグメンテーションの結果と、ベクターデータの道路の座標を用いて、車道部、歩道部、島の領域に区分けされたベクター形式のデータを作成する。ツールの 7 ステップ目がこの処理にあたる。

ベクトル化の処理は、以下のステップからなる。

1. オクルージョン再分類処理の結果から車道部、歩道部、島の境界線を求める
2. 境界線を分割する
3. 境界線のスムージング処理を行う
4. 画像の UV 座標から、オルソ画像の座標系に変換する
5. 道路ごとに以下の処理を行う
    1. 境界線で道路ポリゴンを分割する
    2. 分割された領域ごとにセマンティックセグメンテーションの結果と照らし合わせて、ラベルを決定する

① セマンティックセグメンテーションの結果(後処理済み)から車道部、歩道部、島の境界線を求める

セマンティックセグメンテーションの結果の各ピクセルで、左右上下のピクセルと比較し、ラベルが異なる場合は、その間を境界とする線を引く。これを行うと、ピクセルの格子点を頂点、引いた境界を辺としたグラフが作成される。その例を図 3-92 に示す。 ただし、道路外との境界には元の道路ポリゴンの座標を使用するため、この処理では道路外との境界は抽出しない。

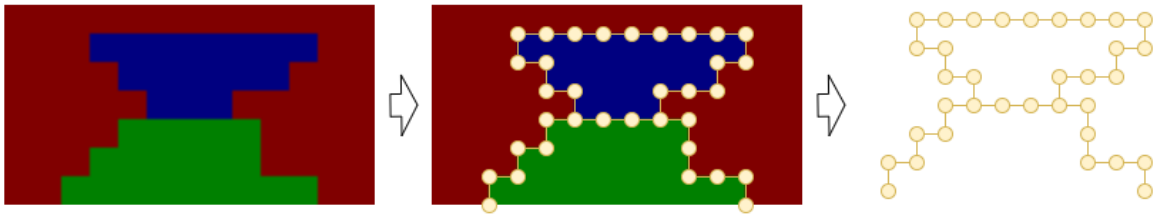


図 3-92　境界線の抽出例

② 境界線を分割する

前ステップで作成したグラフにスムージング処理を行う単位に分解する。分解に当たって以下のいずれかを満たす頂点は必ず端点となるように行う。

- 次数が 1 の頂点
- 次数が 3 以上の頂点
- 道路縁にある頂点

また、上記の分割では、分岐と道路縁上の頂点が含まれていないループ状の境界線は考慮されない。そのため、端点が存在しない連結成分は各々1 つのループとして扱う。

分割する例を図 3-93 に示す。

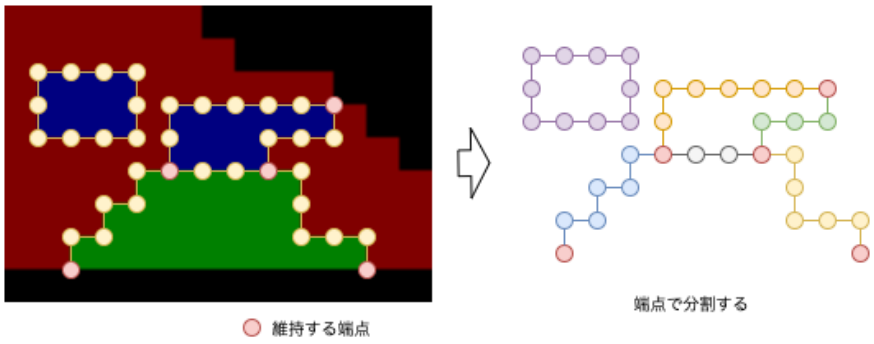


図 3-93　境界線の分割

③ 境界線のスムージング処理を行う

スムージング処理は、前ステップで分割された単位で処理を行う。処理は 2 段階からなる。

1 段階目では、前ステップの境界線抽出処理によって発生するジャギーを消すために、OpenCV の `approxPolyDP` 関数を使用して、少ない頂点での近似を行う。例を図 3-94 に示す。 2 段階目では、折れ線の角度を緩やかにするために、shapelysmooth[12]の `chaikin_smooth` 関数を使ってコーナーカットアルゴリズムを適用する。例を図 3-95 に示す。

スムージング処理で端点の座標が移動した場合には接続関係が変化してしまうため、端点の座標は変化しないように処理を行う。

[12] https://github.com/philipschall/shapelysmooth

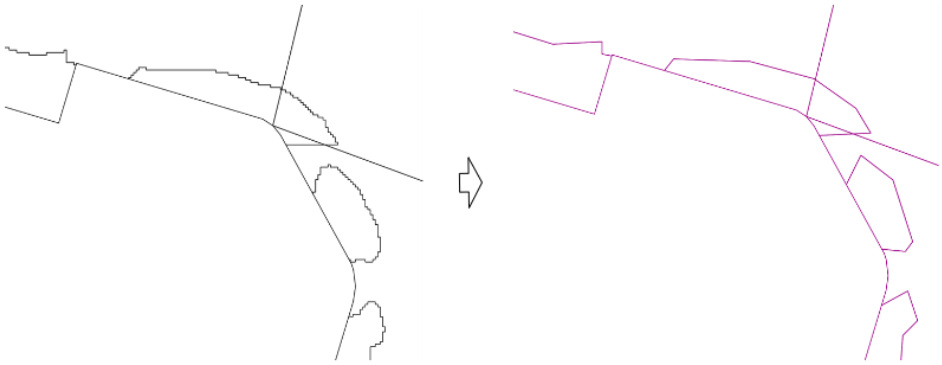


図 3-94　境界線のジャギー除去

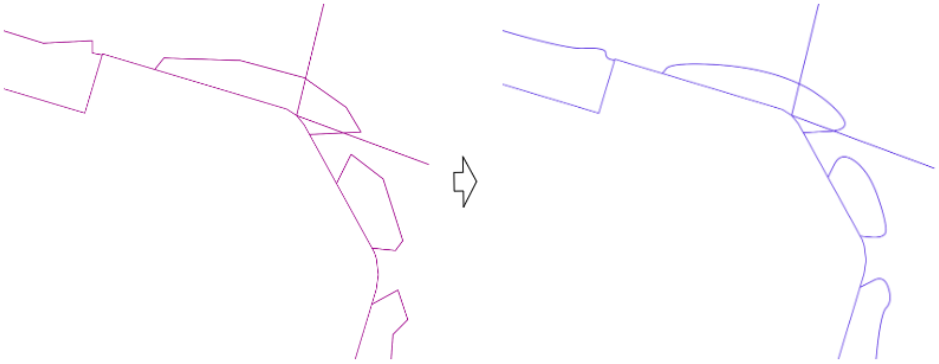


図 3-95　境界線のコーナーカット

④　画像の UV 座標から、オルソ画像の座標系に変換する

対応するオルソ画像のワールドファイルの値を使用して座標系を変換する。この処理は、この後の道路を分割する処理で道路ポリゴンの座標系と合わせるために行う。

画像の UV 座標を(u, v)としたとき、オルソ画像の座標系での座標(x, y)は、次の式で求まる。

$$x = u \cdot A + C$$
$$y = v \cdot E + F$$

ただし、ワールドファイルの値を 1 行目から、A, D, B, E, C, F とし、D = 0, B = 0 とする。

⑤　道路ごとに境界線で分割

道路ごとに境界線で分割する処理は、以下の流れで行う。

1. 境界線から、道路内の線を抽出する
2. 境界線の端と道路ポリゴンの辺との間に隙間がある可能性があるため、境界線を延長する
3. 境界線と道路ポリゴンの辺から囲われた領域ごとにポリゴンを生成する
4. 2 の境界線延長による影響で道路ポリゴン外にポリゴンが作成される可能性があるため除去する

2 の延長については、ラスタ画像から境界線を抽出していることによって発生する道路ポリゴンとの間の隙間への対応である。道路ポリゴンと境界線のずれを図 3-96 示す。完全に交差していない場合には領域が分割されていないと判断されて、ポリゴン生成時にその境界が無視されるため、延長することで完全に交差するようにする。

3 の処理は shapely[13]の polygonize 関数を使用することで実現している。

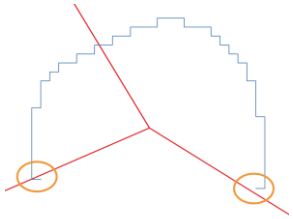


図 3-96　境界線と道路ポリゴンのずれ

⑥　分割された領域ごとのラベル付け

各領域のラベルは、セグメンテーションの結果を基に決められる。

セグメンテーションの結果はラスタ形式のため、各領域を一度画像の UV 座標に変換しマスク画像を作成する。 マスク画像をセグメンテーションの結果と合わせ、マスク内の最頻値をその領域のラベルとする。

領域の面積が特に小さい場合、まれに座標変換の誤差により 1 ピクセルもラベルが存在しないことがある。この場合は、その領域に最も近いラベルを採用する。

ベクトル化処理を行った結果の例として、03PE962-RGB-16cm の推論結果のベクトル化を行った際の入出力を図 3-97 に示す。

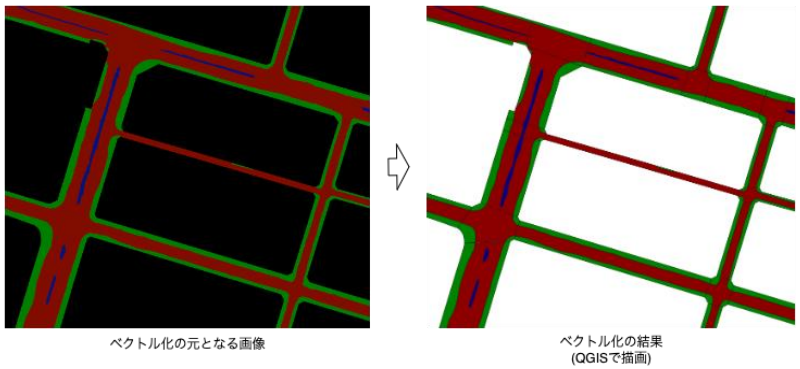


図 3-97　ベクトル化の結果

(5) 例外処理

ベクトル化モジュールの例外処理の一覧を表 3-68 に示す。

表 3-68　ベクトル化モジュールの例外処理の一覧

[13] https://github.com/shapely/shapely

No.	エラー名称	発生条件	処理内容
1	ファイル読み込みエラー	入力ファイル読み込みに失敗	エラーログを出力し、モジュールの処理を終了する。

3.5.4.8. CityGML 出力モジュール

(1) 概要

LOD1 CityGML データに、前述までの処理で作成した LOD2 道路情報データを追加し、LOD1 と LOD2 を含む CityGML データを作成する。

(2) 入力データ

CityGML 出力モジュールの入力データの一覧を表 3-69 に示す。

表 3-69 CityGML 出力モジュールの入力データの一覧

No.	データ名	入力元	説明
1	CityGML 入力ファイルパス	設定パラメータファイル	・設定パラメータファイルの"CityGMLFolderPath"キー が指定するフォルダ内に存在する CityGML ファイルのパス
2	道路情報（ベクトル化後データ）	[vectorized] フォルダ	・道路情報をベクトル化したデータ ・ファイルフォーマットは YAML ファイル形式
3	航空写真（オルソ）の EPSG コード	設定パラメータファイル	・設定パラメータファイルの“OrthoEPSG”キー が指定する、オルソ画像のワールドファイルに記載された座標の EPSG コード
4	CityGML の EPSG コード	設定パラメータファイル	・設定パラメータファイルの"CityGmlEPSG"キー が指定する、CityGML に記載されている座標の EPSG コード

(3) 出力データ

CityGML 出力モジュールの出力データの一覧を表 3-70 に示す。

表 3-70 CityGML 出力モジュールの出力データの一覧

No.	データ名	出力先	説明
1	LOD2 CityGML	[result_output] フォルダ	・入力された CityGML データに LOD2 道路情報データを追記したデータ
2	ログメッセージ	CityGML 出力ログファイル	・CityGML 出力のログメッセージを出力する

			・CityGML 出力に失敗した際のエラーメッセージを主に出力する ・ログメッセージは、設定パラメータファイルの"OutputLogDir"で指定したフォルダ内の"outputlog_YYYYMMDD_HHMMSS"フォルダに export_citygml_log.txt として出力する ・出力例を図 3-98 に示す

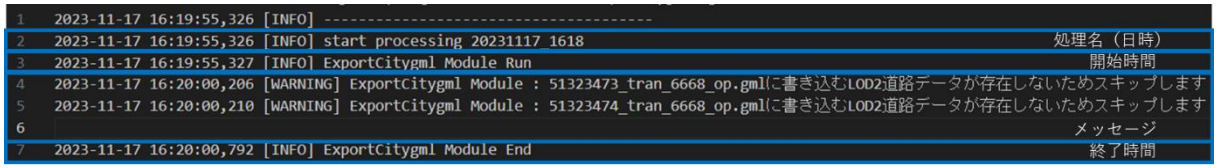


図 3-98　CityGML 出力ログ出力例

(4) 処理内容

本ツールで生成した LOD2 道路情報を、入力の CityGML ファイルに追記し、出力する。ツールの 8 ステップ目がこの処理にあたる。

まず、`tran:TrafficArea` 又は `tran:AuxiliaryTrafficArea` を、LOD1 道路の `tran:Road` の子として追加する。

(5) 例外処理

CityGML 出力モジュールの例外処理の一覧を表 3-71 に示す。

表 3-71　CityGML 出力モジュールの例外処理の一覧

No.	エラー名称	発生条件	処理内容
1	ファイル読み込みエラー	入力ファイル読み込みに失敗	エラーログを出力し、モジュールの処理を終了する。
2	対応道路データなし	入力 CityGML に書き込まれた LOD2 道路データがない	次の CityGML の処理に移る。

3.5.5.ファイル仕様

本システムにて使用されるファイルの一覧を表 3-72 に示す。

表 3-72　使用ファイルの一覧

No.	ファイル名	概要
1	設定パラメータ	実行時に使用するパラメータ
2	航空写真（オルソ画像）	航空写真のオルソ画像
3	CityGML 入力ファイル	道路形状（LOD1）、属性
4	道路 LOD1 シェープファイル	道路 LOD1 ポリコンを含めたシェープファイル
5	CityGML 出力ファイル	道路形状（LOD1、LOD2）、属性
6	道路 LOD2 シェープファイル	道路 LOD2 ポリゴンを含めたシェープファイル
7	実行ログ	実行履歴を記録する
8	モジュールログ	各モジュールのエラー内容を記録する
9	モデル化結果サマリー	各道路の実行結果を一覧で記録する

3.5.5.1. 設定パラメータ

本システムの実行時に使用するパラメータの仕様を表 3-73 に示す。

表 3-73　設定パラメータファイルの仕様

ファイル形式	JSON
ファイル名	param.json
格納フォルダ	任意
入力先	システム全般
出力元	-
特記事項	文字コードは UTF-8 とする。

本システムの実行時に使用する設定パラメータを表 3-74 に示す。

表 3-74　本システムの設定パラメータ

No.	キー名	値形式	説明
1	OrthoDir	文字列	推論に使用するオルソ画像(ワールドファイル付き)が保存されているディレクトリのパス
2	InputType	整数値	入力ファイルに CityGML ファイル・シェープファイルのどちらを使うかを指定する １：CityGML ファイル ２：シェープファイル
3	CityGMLDir	文字列	LOD1 CityGML が保存されているディレクトリのパス
4	ShapeDir	文字列	LOD1 シェープファイルが保存されているディレクトリのパス
5	OutputDir	文字列	途中結果と最終結果を出力するディレクトリのパス
6	OrthoEpsg	整数値	オルソ画像のワールドファイルに記載された座標の EPSG コード
7	CityGmlEpsg	整数値	CityGML に記載されている座標の EPSG コード
8	【任意】Device	文字列	推論を実行するデバイス。CPU のみの環境では cpu を指定する
9	【任意】InferenceSize	整数値	推論を行う際のタイルの 1 辺のサイズ(pixel)
10	【任意】InferencePadding	整数値	推論を行う際のタイルで推論結果を使用しない外側の幅(pixel)
11	【任意】NoiseThreshold	整数値	小さな領域の除去で除去する領域のピクセル数のしきい値
12	【任意】StartWith	文字列	途中から実行したい場合の開始ステップ名 (load_citygml, check_road_connectivity, generate_inference_input, infer, remove_noise, remove_occlusion, vectorize, export_citygml) output_dir にそのステップの直前までの結果が含まれている必要がある
13	【任意】EndWith	文字列	途中で実行を止めたい場合の最後のステップ名 (load_citygml,check_road_connectivity, generate_inference_input, infer,

			remove_noise, remove_occlusion, vectorize, export_citygml)
14	OutputLogDir	文字列	ログのフォルダパス。 未記入又は存在しない場合は、本システムの Python コードと同階層のログフォルダ"output_log"にログファイルを作成し、処理を中止する。
15	DebugLogOutput	真偽値	デバッグレベルのログを出力するかどうかのフラグ。 True 又は false で値を指定する。 未記入又は真偽値以外の値が入力された場合は、エラーメッセージを表示し、処理を中止する。

設定パラメータファイルの記載例を図 3-99 に示す。

図 3-99　設定パラメータファイルの記載例

3.5.5.2. 航空写真（オルソ画像）

航空写真測量で撮影したオルソ画像（中心投影）ファイルの仕様を表 3-75 に示す。

表 3-75　航空写真（オルソ画像）ファイルの仕様

ファイル形式	TIFF
ファイル名	XXX.tif、「XXX」はファイルの識別子となる。
格納フォルダ	「設定パラメータ」にて指定する。
入力先	
出力元	-
特記事項	-

航空写真（オルソ画像）格納フォルダの構成例を図 3-100 に示す。

図 3-100　航空写真（オルソ画像）格納フォルダの構成例

3.5.5.3. CityGML 入力ファイル

CityGML 入力ファイルの仕様を表 3-76 に示す。

表 3-76　CityGML 入力ファイルの仕様

ファイル形式	GML
ファイル名	設定パラメータにて指定する。
格納フォルダ	設定パラメータにて指定する。
入力先	CityGML 入力モジュール、CityGML 出力モジュール
出力元	-
特記事項	・道路情報に関して以下が記載されるものとする。 1. gml:id 2. gml:Polygon

3.5.5.4. LOD1 道路シェープファイル

LOD1 道路シェープファイル入力ファイルの仕様を表 3-77 に示す。

表 3-77　LOD1 道路シェープファイル入力ファイルの仕様

ファイル形式	シェープファイル形式
ファイル名	設定パラメータにて指定する。
格納フォルダ	設定パラメータにて指定する。
入力先	CityGML 入力モジュール、CityGML 出力モジュール
出力元	-
特記事項	・道路情報に関して以下が記載されるものとする。属性情報は id のみ使用する。 1. Polygon 2. id(任意)

3.5.5.5. CityGML 出力ファイル

CityGML 出力ファイルの仕様を表 3-78 に示す。

表 3-78　CityGML 出力ファイルの仕様

ファイル形式	GML
ファイル名	設定パラメータにて指定する。
格納フォルダ	設定パラメータにて指定する。
入力先	-
出力元	CityGML 出力モジュール
特記事項	道路情報に関して LOD0、LOD1、LOD2 が記載されるものとする。

3.5.5.6. LOD2 道路シェープファイル

LOD2 道路シェープファイル出力ファイルの仕様を表 3-79 に示す。

表 3-79　LOD2 道路シェープファイル出力ファイルの仕様

ファイル形式	シェープファイル形式
ファイル名	設定パラメータにて指定する。
格納フォルダ	設定パラメータにて指定する。
入力先	-
出力元	CityGML 出力モジュール
特記事項	・道路情報に関して LOD1、LOD2 が記載されるものとする。 ・入力データに id が含まれていなかった場合、任意の id を指定する。

3.5.5.7. 実行ログ

本システムの実行ログを格納するファイルの仕様を表 3-80 に示す。出力内容は、指定パラメータ内容、処理開始時刻、処理終了時刻、処理時間等とする。

表 3-80　実行ログファイルの仕様

ファイル形式	テキスト形式
ファイル名	main_log.txt
格納フォルダ	設定パラメータにて指定する。
入力先	-
出力元	システム全般
特記事項	

3.5.5.8. モジュールログ

モジュールログを格納するファイルの仕様を表 3-81 に示す。各モジュールの動作ログを出力する。

表 3-81　モジュールログファイルの仕様

ファイル形式	テキストファイル
ファイル名	・ 道路 LOD1 データ入力モジュール load_citygml_log.txt ・ 道路の隣接関係の判定モジュール check_road_connectivity_log.txt ・ 推論用データの生成モジュール generate_inference_input_log.txt ・ セグメンテーションモジュール infer_log.txt ・ ノイズ除去モジュール remove_noise_log.txt ・ オクルージョンの再分類モジュール remove_occlusion_log.txt ・ ベクトル化モジュール vectorize_log.txt ・ CityGML 出力モジュール export_citygml_log.txt
格納フォルダ	設定パラメータにて指定する。
入力先	-
出力元	各モジュール
特記事項	

3.5.5.9. 結果サマリー

結果サマリーを格納するファイルの仕様を表 3-82 に示す。各道路の作成結果を CSV ファイル形式で出力する。

表 3-82　結果サマリーファイルの仕様

ファイル形式	CSV
ファイル名	road_create_result.csv
格納フォルダ	設定パラメータにて指定する。
入力先	-
出力元	システム全般
特記事項	・パラメータファイル入力エラー又は CityGML 入力の結果が ERROR の場合は出力しない。 ・サマリーファイルの項目は以下とする。 ・No. ・ファイル名 ・道路 ID ・最終結果 ・CityGML 読み込み結果 ・LOD2 モデルの作成結果 ・　文字コードは UTF-8 BOM 付きとする。

結果サマリーファイルの記載例を図 3-101 に示す。

図 3-101　モデル化結果サマリーファイル出力例

3.6. LOD2 道路モデル自動作成ツールの検証

3.6.1.検証概要

開発した LOD2 道路モデル自動作成ツールの有効性を検証するため、生成したモデルの正確度と費用削減効果についての評価及びデータ整備事業者による検証を行った。

3.6.2.モデル正確度評価

開発した LOD2 道路モデル自動作成ツールのモデル正確度検証の検証を行った。自動作成ツールから生成した LOD2 道路モデルと手動作成による LOD2 道路モデルのポリゴンデータを比較検証した。3.6.2.1 に評価対象地域の選定、3.6.2.2 に評価方法、3.6.2.3 に評価結果、3.6.2.4 にモデル生成例を記載する

3.6.2.1. 対象地域の選定

評価対象地域は、岐阜県岐阜市（以下、「岐阜市」）と広島県広島市（以下、「広島市」）から選定し、岐阜市では 8 図郭（800x600 ㎡）、広島市では 1 図郭（2000x1500 ㎡）を検証した。

各都市の検証範囲は、図 3-102、図 3-103 の青い枠の図郭を示す。

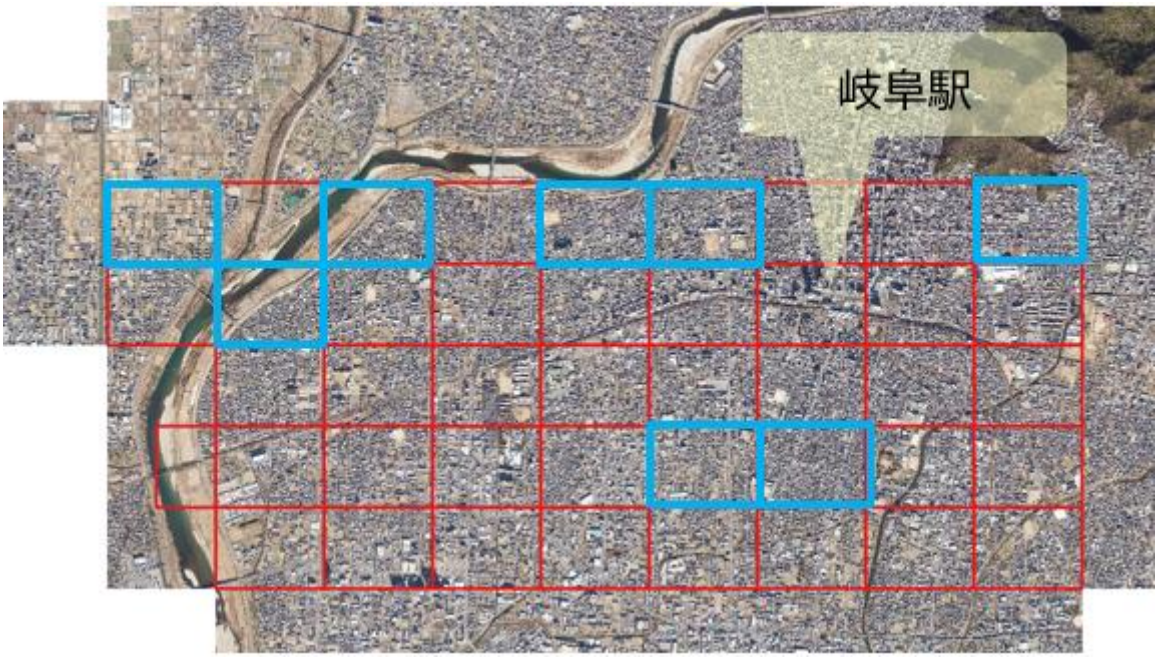


図 3-102　岐阜市_検証対象地域（青枠）

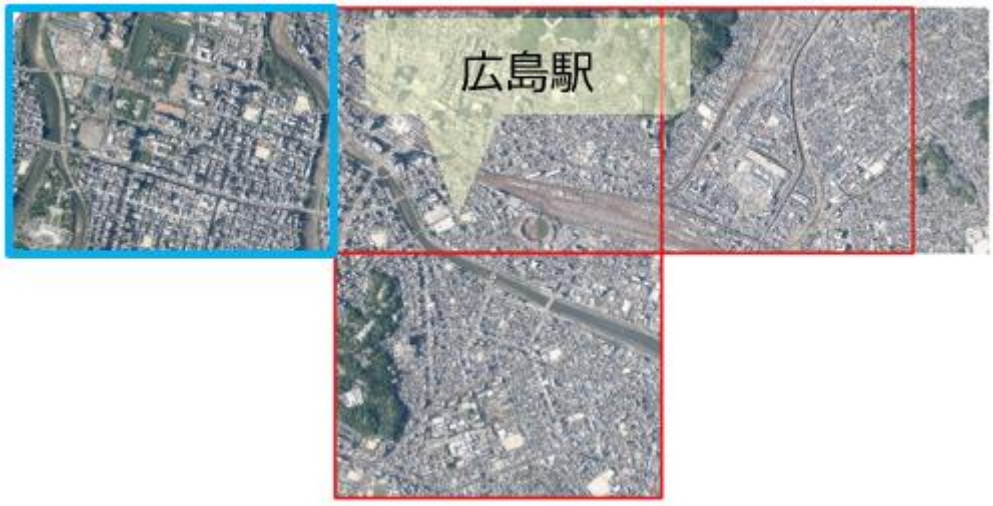


図 3-103　広島市_検証対象地域（青枠）

3.6.2.2. 評価方法

モデル正確度評価では定量評価と定性評価の二つの手法を用いた。

(1) 定量評価

自動作成ツールから生成した LOD2 道路ポリゴン（以下、「予測ポリゴン」）と手動作成による LOD2 道路ポリゴン（以下、「正解ポリゴン」）との比較で定量評価を行った（図 3-104）。評価指標には機械学習や画像処理で良く使用されている Intersection over Union(IoU)、Precision(適合率)、Recall(再現率)を使用した（図 3-105）。

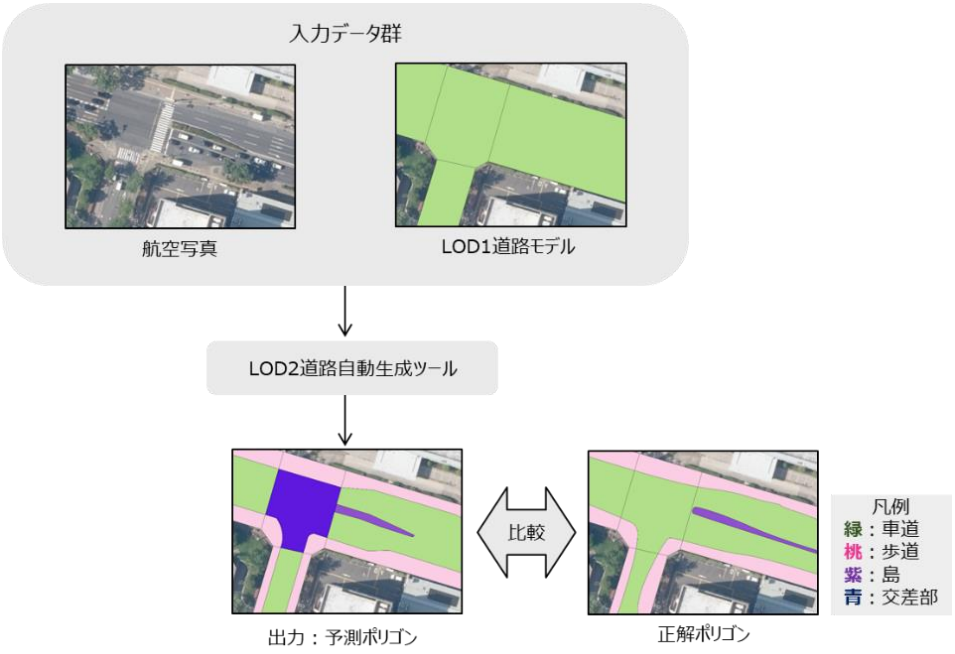


図 3-104　定量評価フロー図

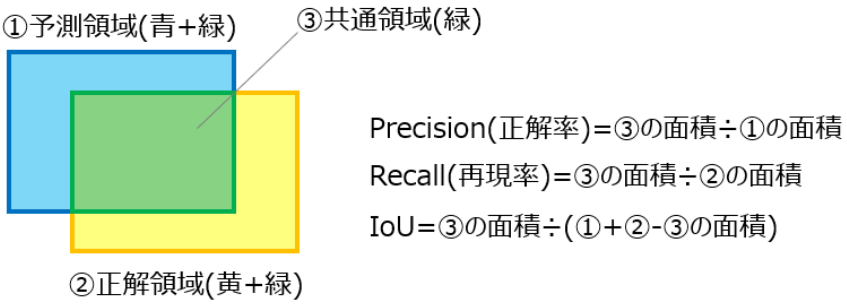


図 3-105　Precision、Recall、IoU の計算方法

(2) 定性評価

入力された LOD1 道路モデルごとに、生成した LOD2 予測ポリゴンと正解ポリゴンとの領域誤差量を算出し、その誤差が 1.75m 以上離れている場合を誤りとし、その誤りポリゴンの割合を評価値として算出した。

評価値をもとに A/B/C 評価を決定した。※A/B/C 評価は自動計算

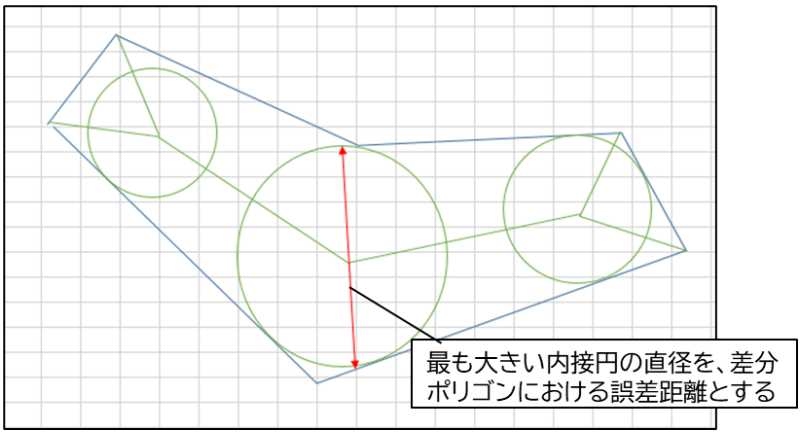


図 3-106　予測と正解ポリゴンの差分ポリゴン

図 3-107　領域の差分結果例

算出した評価値をもとに A/B/C 判定を行った。A/B/C 評価の基準を

表 3-83 に示す。

表 3-83　定性評価基準

評価指標	評価基準
A	修正不要（誤り箇所なし）
B	誤り箇所が 5 割以下
C	誤り箇所が 5 割以上

3.6.2.3. 評価結果

(1) 定量評価結果

岐阜市と広島市の定量評価結果を表 3-84、表 3-85 に示す。岐阜市広島市ともに、良好な結果が得られた。広島市では、島の評価値が低い要因は、対象データが少ないことで少量の誤認識でも数値への影響が大きいことが考えられる。広島市の予測ポリゴンと正解ポリゴンの混同行列を表 3-86 に示す。島の合計面積は僅か 138.043 ㎡であることが分かる。

表 3-84　岐阜市_定量評価結果

評価指標	車道	車道交差部	歩道	島	Mean
Precision	0.967	0.697	0.928	0.814	0.852
Recall	0.940	0.875	0.881	0.822	0.880
IoU	0.911	0.634	0.825	0.692	0.766

表 3-85　広島市_定量評価結果

評価指標	車道	車道交差部	歩道	島	Mean
Precision	0.925	0.622	0.914	0.062	0.631
Recall	0.902	0.931	0.688	0.637	0.790
IoU	0.840	0.595	0.646	0.06	0.535

表 3-86　広島市_予測・正解ポリゴンの混同行列

		正解ポリゴン				
		車道	車道交差部	歩道	島	合計(面積)
予測ポリゴン	車道	317647.268	5702.273	20061.804	5.603	343416.948
	車道交差部	24511.870	79514.922	23768.011	44.658	127839.461
	歩道	8888.847	205.049	96692.322	0.000	105786.219
	島	1194.359	0.000	90.942	88.043	1373.345
	合計(面積)	352242.345	85422.245	140613.079	138.304	

(2) 定性評価結果

岐阜市と広島市の定性評価結果を表 3-87、表 3-88 に示す。岐阜市広島市ともに、修正不要の A 評価が多い結果となった。岐阜市では、A 評価が 8 割超えの結果となり、予測ポリゴンと正解ポリゴンとほぼ同等であることが示されている。

表 3-87　岐阜市_定性評価結果

評価指標	基準	道路数(ID 毎)	割合
A	修正不要（誤り箇所なし）	1864	85%
B	誤り箇所が 5 割以下	136	6%
C	誤り箇所が 5 割以上	183	8%

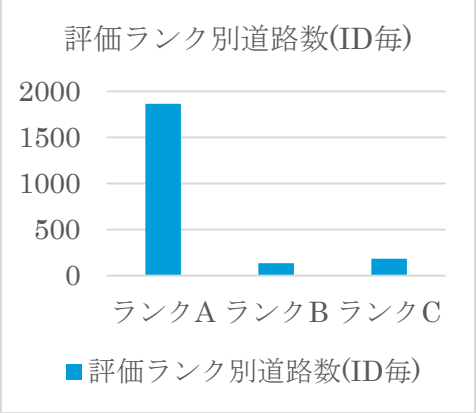


図 3-108　岐阜市_定性評価結果

表 3-88　広島市_定性評価結果

評価指標	基準	道路数(ID 毎)	割合
A	修正不要（誤り箇所なし）	648	66%
B	誤り箇所が 5 割以下	212	21%
C	誤り箇所が 5 割以上	126	13%

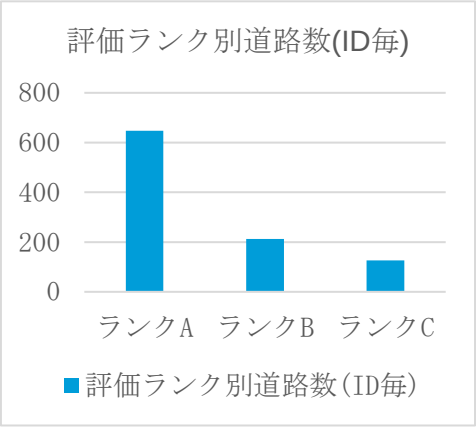


図 3-109　広島市_定性評価結果

3.6.2.4. モデル作成例

モデル作成例を図 3-110、図 3-111 に示す。

図 3-110　道路 LOD2 モデル作成例(1)

図 3-111　道路 LOD2 モデル作成例(2)

3.6.3.費用削減効果評価

費用削減効果評価では、自動作成ツールによる道路 LOD2 モデル作成作業の工数削減率について評価を行った。評価対象地域は 3.6.2.1 と同じ地域を選定した。

3.6.3.1. 評価方法

モデル作成するにあたり頂点数が多いポリゴンほど修正にかかる作業コストが大きいことから、修正が必要なポリゴンの頂点数を基準にして評価指標を設けた。作業削減率の評価基準を表 3-89 に示す。

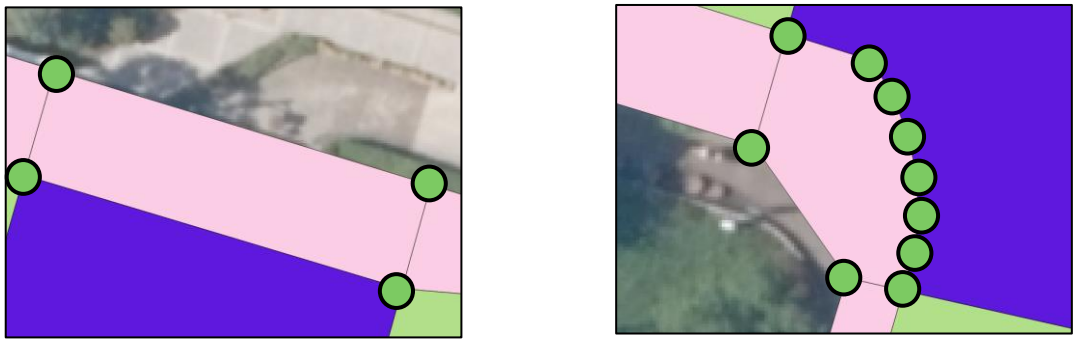


図 3-112　頂点数と修正コスト

表 3-89 経済効果の算出基準

評価指標	工数削減率	評価基準
-	100%	修正不要（誤り箇所なし）
A	75%	要修正かつ頂点数が 10 点以下
B	50%	要修正かつ頂点数が 20 点以下
C	25%	要修正かつ頂点数が 20 より多い
D	0%	対象外（オクルージョン、経年変化が含まれる面積が大きい等）

3.6.3.2. 評価結果

岐阜市と広島市の評価結果を表 3-90、表 3-91 に示す。合計工数削減率は、岐阜市では 93.6%、広島市では 89.0%という結果となった。(2)定性評価結果の通り、修正不要の A 評価が多いことから工数削減率も高いことが分かる。

表 3-90　岐阜市_経済効果の算出結果

評価指標	基準	工数削減率	ポリゴン数	割合
-	修正不要	100%	2649	87%
A	要修正 かつ 頂点数が 10 点以下	75%	174	6%
B	要修正 かつ 頂点数が 20 点以下	50%	68	2%
C	要修正 かつ 頂点数が 20 より多い	25%	155	5%
D	対象外（オクルージョン、経年変化が含まれる面積が大きい等）	0%	4	0%

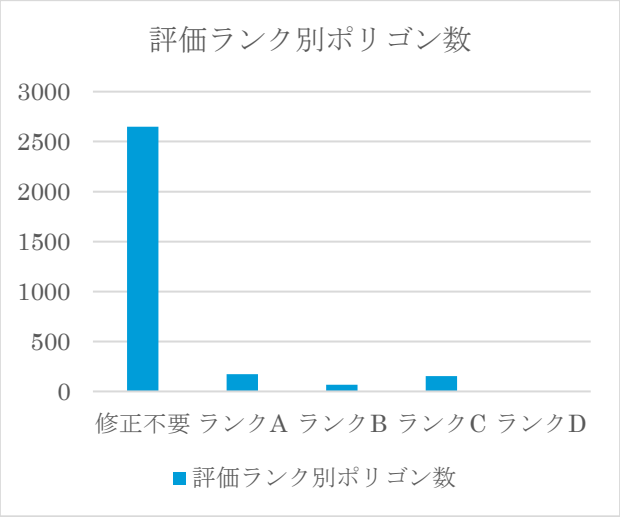


図 3-113　岐阜市_経済効果の算出結果

表 3-91　広島市_経済効果の算出結果

評価指標	基準	工数削減率	ポリゴン数	割合
-	修正不要	100%	1909	78%
A	要修正 かつ 頂点数が 10 点以下	75%	82	3%
B	要修正 かつ 頂点数が 20 点以下	50%	36	2%
C	要修正 かつ 頂点数が 20 より多い	25%	290	12%
D	対象外（オクルージョン、経年変化が含まれる面積が大きい等）	0%	123	5%

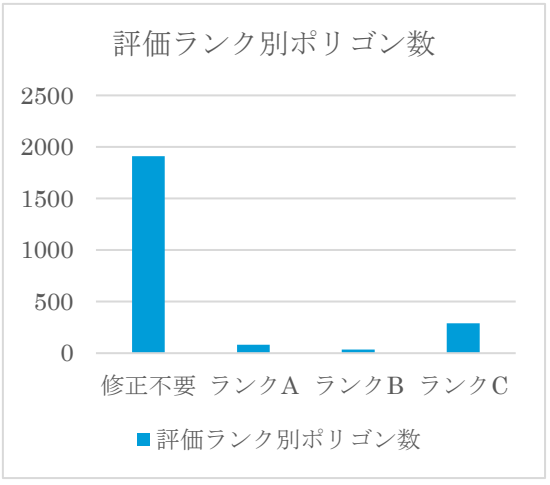


図 3-114　広島市_経済効果の算出結果

3.6.4.データ整備事業者による検証

LOD2 道路モデル自動作成ツールの使いやすさを検証することを目的として、データ整備事業者（以下、「検証者」という。）の協力の下でユーザビリティ検証を行った。

実際に OSS として公開予定であるプログラムを検証者に共有し、検証者のデータでモデルの作成を行った。

3.6.4.1. 試行環境

試行に利用した PC スペックを表 3-92 に示す。

表 3-92　試行に利用した PC スペック

OS	Windows 10 Pro 64 ビット
CPU	Intel® Core™ i7-11800H プロセッサー 8 コア / 16 スレッド / 2.30GHz [最大 4.60GHz] / 24MB キャッシュ
GPU	NVIDIA GeForce RTX 3060 Laptop GPU / GDDR6 6GB
メモリ	32GB メモリ [16GB×2 (DDR4-3200) / デュアルチャネル]
ストレージ	SSD 512GB NVM Express SSD (M.2 PCI Express 接続)

3.6.4.2. 試行の観点

試行において表 3-93 の観点でヒアリングした。

表 3-93　試行の観点

項目	定義
品質	生成した道路モデルの形状、交差部等の分離状況を評価
効率性／処理速度	既往の手作業を含む手順と比較し、データ生産が効率化するかを評価
利用しやすさ	開発したツールの使いやすさを評価

3.6.4.3. 検証結果

利用するデータの画像解像度により、精度が大きく変わるため、利用しやすさの向上には、利用するデータに関する詳しい説明が必要であることがわかった。

航空写真を入力として、AI を用いて画像のセグメンテーションを行い、道路ポリゴンを作成する手法について、公共測量成果とするためのルールが未整備の状態では実業務に利用できないとの意見も挙がった。

3.6.5.まとめと課題

岐阜市と広島市で選定した対象地域において、自動作成ツールから作成したモデルの正確度と費用削減効果の評価を行い、開発したツールの有効性が確認できた。モデル正確度評価では、修正不要のモデルが岐阜市では全体の 8 割以上、広島市では全体の 6 割以上を占めており、良好な結果となった。また、費用削減効果評価では、岐阜市と広島市のそれぞれの合計工数削減率は 93.6%、89.0%となっており、経済的な有効性も示されている。

一方で、影やオクルージョンの影響により正しくないモデルが生成された課題も見られた。最新技術の適用などによる精度向上は今後の取り組みが必要である。また、今回の検証は岐阜市と広島市の狭い範囲で行われたが、ツールの汎化性能を確認するためには、より広範囲な検証が必要と考えられる。

4. OSS 化

開発した LOD2 建築物モデル自動作成ツール、LOD1-2 道路モデル自動作成ツールは、OSS として PLATEAU GitHub に公開した。GPL v3.0 ライセンス[14]で公開しているため、誰でも使用・改良可能である。

表 4-1　LOD2 建築物モデル自動作成ツールの公開

URL	https://github.com/Project-PLATEAU/Auto-Create-bldg-lod2-tool
公開内容	建築物 LOD2 自動作成ツールのプログラム、チュートリアル、ユーザマニュアル
ライセンス	GPL v3.0

表 4-2　LOD1-2 道路モデル自動作成ツールの公開

URL	https://github.com/Project-PLATEAU/Auto-Create-bldg-lod2-tool
公開内容	道路 LOD1-2 自動作成ツールのプログラム、チュートリアル、ユーザマニュアル
ライセンス	GPL v3.0

5. 成果と課題

LOD2 建築モデルにおいては、AI モデルへの追加学習及び処理の改修により建物形状に対する汎用性が高まり、これまでの学習に利用した地域以外での精度向上を図ることができた。今後はツールの利用環境も含めた検討を行い、使いやすさを向上し、実業務への適用を進める必要がある。

LOD1 道路モデルにおいては、DM データを入力とし、モデルを自動作成できることが実証できたが、道路縁の形状等の条件によって対応できないケースがあるという課題があり、幾何処理のアルゴリズム改修など精度向上を検討し、実業務に適用できるよう自動化の対象範囲を増やす必要がある。ツールのインタフェースはシェープファイルを中間出力として、作業フローに即したツールであることが確認できた。

LOD2 道路モデルにおいては、島など微細な形状でのモデル作成に課題はあるが、航空写真を利用した道路面の分離が実証でき、精度も良好な結果となった。実業務への適用のためには、航空写真を入力とする AI によるセグメンテーションを活用したデータ作成のルール整備が必要と考える。

道路モデル作成における費用削減効果の検証では、手作業が多い道路モデル作成に自動化ツールを用いることで作業時間の短縮に寄与できることが確認できた。今後、LOD2 自動作成ツールは

[14] GPLv3 クイック・ガイド - GNU プロジェクト - フリーソフトウェアファウンデーション

ツールの利用環境を含めた検討を行い、利用しやすいツールとして実用化へ進めることが必要である。

本業務で開発した自動作成ツールを OSS として PLATEAU GitHub に公開した。様々な企業・団体が無料でこのツールを利用して建物、道路データを作成することができる。これにより、地方公共団体等のデータ整備拡大が加速され、まちづくり DX の更なる推進が期待できる。

6. 用語集

表 6-1　用語の一覧

用語	説明
内部標定要素	航空写真撮影に使用したカメラのレンズ焦点距離、センサーサイズ、レンズの放射方向歪曲収差を表す近似多項式の係数などの情報。
外部標定要素	航空写真撮影時のカメラの 3 次元座標と 3 軸の傾き情報。
オルソ画像	航空カメラが撮影した空中写真に対して、正射変換を行った画像。
OBJ ファイル	3D モデルを表現するためのファイルフォーマット。
LAS ファイル	LiDAR(Light Detection and Ranging、Laser Imaging Detection and Ranging)で取得した点群データを保存するためのファイルフォーマット。
シェープファイル	GIS データフォーマットの 1 つ。道路や建物などの位置や形状、属性情報を持つベクターデータ(ポイント、ライン、ポリゴン)を格納することが可能
地域メッシュ	緯度・経度に基づいて地域をほぼ同じ大きさの網の目に分割したもの
第 1 次地域区画	全国の偶数緯度及びその間隔(120 分)を 3 等分にした緯度における緯線並びに 1 度ごとの形成によって分割してできる区域であり、1 辺は約 80km(緯度間隔 40 分，経度間隔 1 度)。
第 2 次地域区画(統合地域メッシュ)	全国の偶数緯度及びその間隔(120 分)を 3 等分にした緯度における緯線並びに 1 度ごとの形成によって分割してできる区域であり、1 辺は約 80km(緯度間隔 40 分，経度間隔 1 度)。
基準地域メッシュ(第 3 次地域区画)	第 2 次地域区画を緯線方向及び経線方向に 10 等分して出来る区域であり、一辺は約 1km(緯度間隔 30 秒，経度間隔 45 秒)
2 分の 1 地域メッシュ	基準地域メッシュを緯線方向及び経線方向に 2 等分して出来る区域であり、一辺は約 500m(緯度間隔 15 秒，経度間隔 22.5 秒)。
4 分の 1 地域メッシュ	2 分の 1 地域メッシュを線方向及び経線方向に 2 等分して出来る区域であり、一辺は約 250m(緯度間隔 7.5 秒，経度間隔 11.25 秒)。
ArcGIS	ESRI 社により提供される GIS ソフトウェア。地理情報を収集、整理、管理、解析、伝達、および配布するための包括的なシステム
QGIS	地理空間情報データの閲覧、編集機能を有するオープンソースソフトウェアの GIS ソフトウェア
FME	Safe Software 製のデータ変換等を行う機能を有するソフトウェア
公共測量成果検査支援ツール(PSEA)	公共測量において作成された数値地形図データ(DM)等の表示機能と簡易検査機能、汎用フォーマット等への変換機能を有する、国土地理院が提供しているフリーウェア。

表 6-2　略語の一覧

略語		説明
CityGML	City Geography Markup Language	地理空間データに関する標準化団体である Open Geospatial Consortium (OGC) が策定した 3D 都市モデルのためのオープンデータモデル及びデータ形式の国際標準。
GML	Geography Markup Language	Open Geospatial Consortium (OGC)によって開発された地理的な特徴を表現するための XML(Extensible Markup Language)文法。
LOD	Level Of Detail	モデルの詳細度を表し、LOD0～LOD4 の 5 段階が定義されている。
SfM/MVS	Structure from Motion/Multi View Stero	計測対象をさまざまな位置、角度から撮影した画像を基に写真同士の対応関係を解析することで、計測対象の三次元形状を復元する技術。
DSM	Digital Surface Model	数値表層モデル(地物表面)
DM	Digital Map	地形、地物などの地図情報をディジタル形式の数値地形図として作成すること。作成されたデーは DM データという
OSS	Open Source Software	利用者の目的を問わずソースコードを使用、調査、再利用、修正、拡張、再配布が可能なソフトウェアの総称。

7. 参考資料

参考資料	バージョン	参照目的
3D 都市モデル標準製品仕様書	ver.3.5	道路モデルの仕様確認
3D 都市モデル標準作業手順書	ver.3.5	道路モデルの作業手順の確認
道路形状ポリゴンを用いた、道路幅員ネットワークデータの自動生成(奥秋恵子著)	-	道路中心線の作成方法の参考論文

AI 等を活用した LOD2 自動作成ツールの開発及び OSS 化
技術検証レポート

2024 年 3 月　発行

委託者：国土交通省　都市局
受託者：アジア航測株式会社