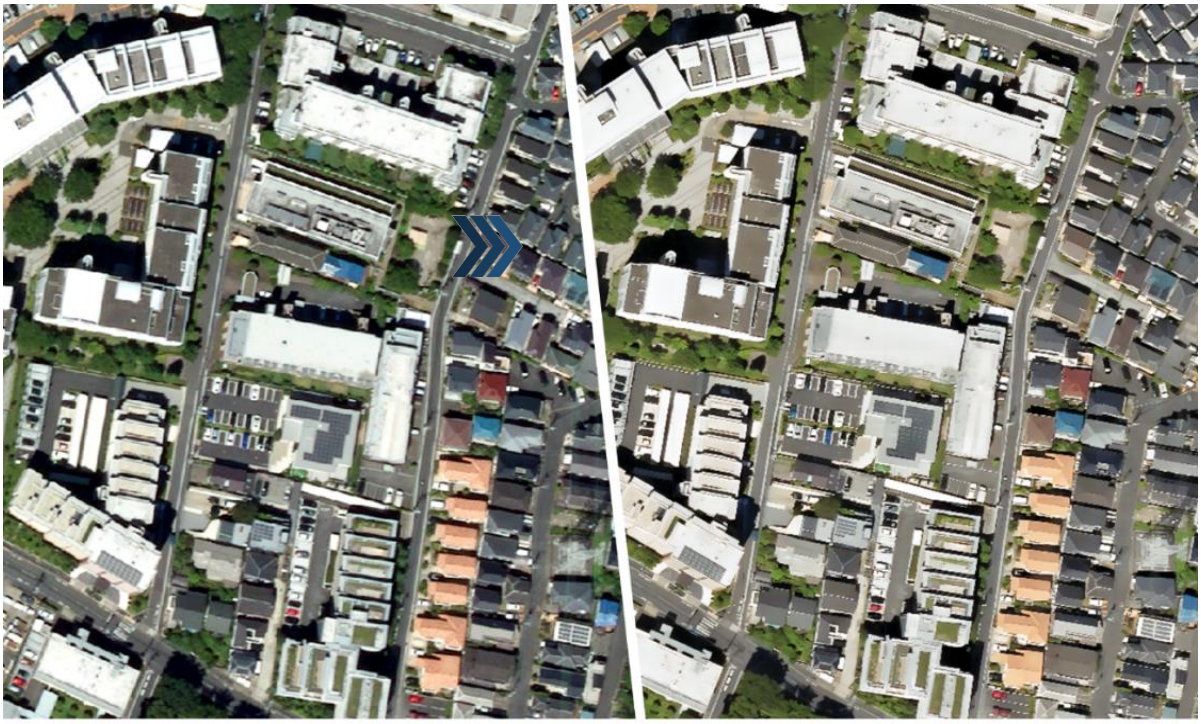
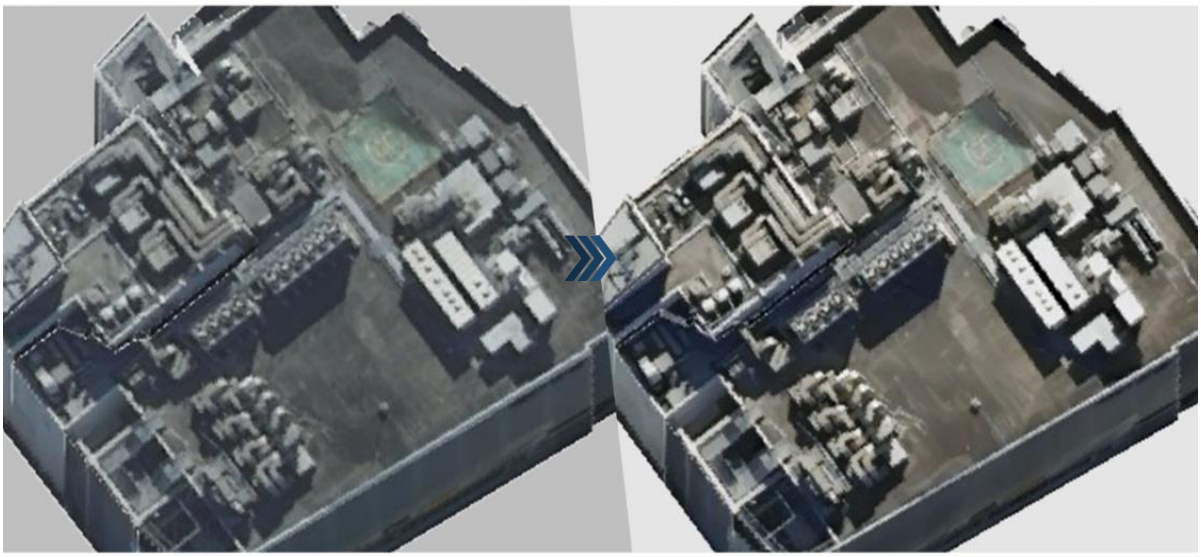
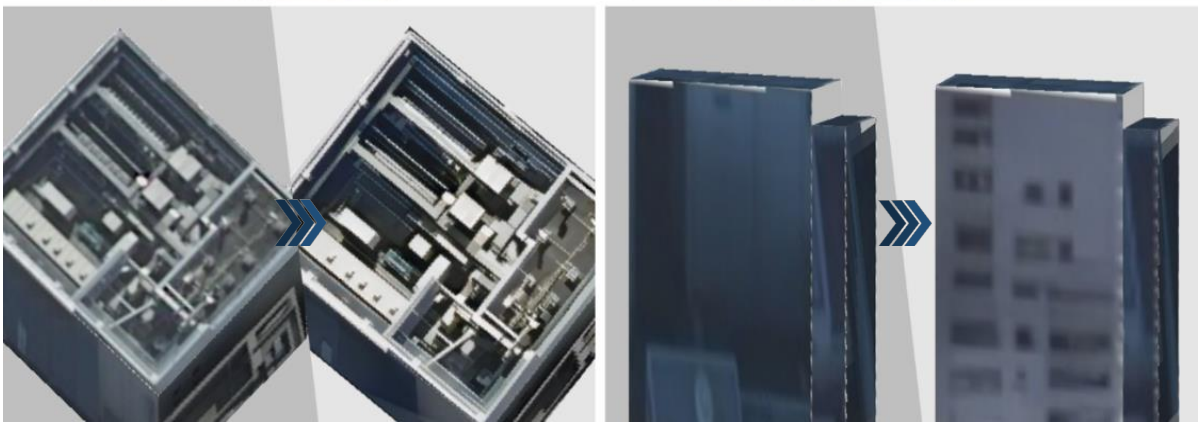
PLATEAU by MLIT

PLATEAU Technical Report

3D 都市モデル活用のための技術資料







3D 都市モデルのテクスチャ高解像度化手法及び描画パフォーマンス向上に関する技術調査レポート

series No. 62

Technical report on tool development to improve visibility and drawing performance of LOD2 building texture images

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 3D 都市モデルのテクスチャ高解像度化手法及び描画パフォーマンス向上に関する技術調査に関する技術調査レポートである。

本レポートは前半で 3D 都市モデルのテクスチャ高解像度化手法に関する技術調査、後半で描画パフォーマンス向上施策の調査検討について記載する。

1.2. 背景

PLATEAU の 3D 都市モデルの有用性、利便性、汎用性を更に高めるため、建物テクスチャの見た目（視認性）や描画パフォーマンスの向上をはじめとしたニーズの高い領域における最新技術の動向に関する技術調査を行う必要がある。

現在の 3D 都市モデルに使われているテクスチャは航空写真をもとに作られているため、解像度が低い問題や、撮影時の太陽高度の影響により影が含まれている問題があり、視認性に課題がある。3D 都市モデルの利用用途の拡大に伴い、テクスチャの見た目に関する要求も高くなっており、利用拡大のためにはテクスチャの高解像度化の手法の検討が必要である。一方で高解像度なテクスチャを使用した場合、WebGL 等での描画性能が低下することが予想されるため、高速にレンダリングするための調査も合わせて行う必要がある。

LOD2 建物テクスチャの視認性が向上し、よりリアルな街並みを表現できるようになれば、3D 都市モデルの利用が自治体や地図業界だけでなく、イベント用の映像やインタラクティブ広告などのエンターテインメント分野へも普及・拡大する可能性がある。

1.3. 目的

3D 都市モデルのテクスチャ高解像度化手法に関する技術調査では、建築物モデルのテクスチャを対象に、高解像度化手法として深層学習による超解像や Generative Adversarial Networks（敵対的生成ネットワーク、以下 GAN）を利用して航空写真画像を精細化し、視認性の良い建物テクスチャの作成方法の検討を目的とする。

描画パフォーマンス向上施策の調査検討では、PLATEAU が採用する CityGML 2.0 及び Web レンダリングフォーマットとしての 3D Tiles における描画パフォーマンス向上を実現するデータ構造の定義のための技術調査を行う。特にテクスチャ付き建物モデル LOD2 の特定ソフトウェアにおけるクライアント上での描画性能及び、3D Tiles の Web GIS アプリケーションでの描画パフォーマンス改善を目的とする。

また、開発したツールをオープンソースソフトウェア（以降 OSS）として PLATEAU GitHub に公開する。

2. 3D 都市モデルのテクスチャ高解像度化手法に関する技術調査

本章では 3D 都市モデルのテクスチャ高解像度化手法に関する技術調査について記載する。

2.1. 技術調査及び手法検討

建物テクスチャのうち屋根面と壁面でその視認性特性が異なることから、各々技術調査及び手法の検討を行った。検討フローを図 2-1 に、画像生成イメージを図 2-2 に示す。

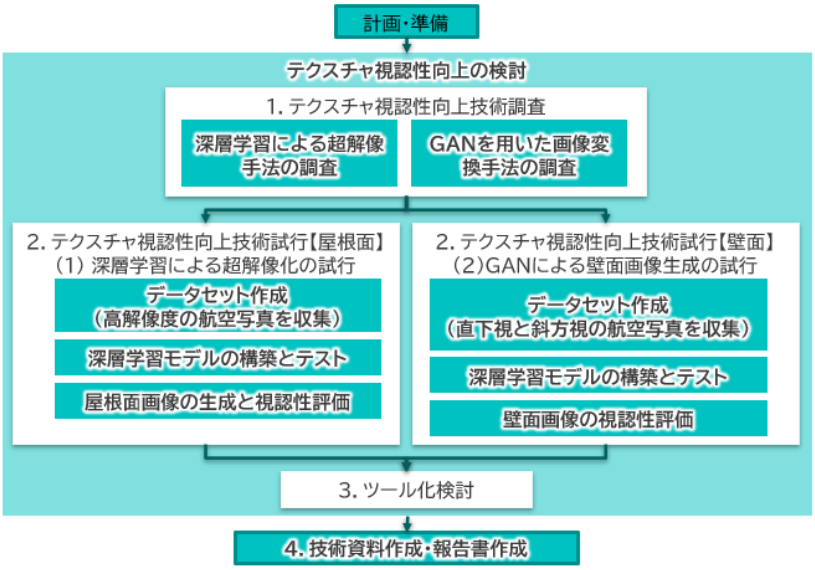


図 2-1　検討フロー

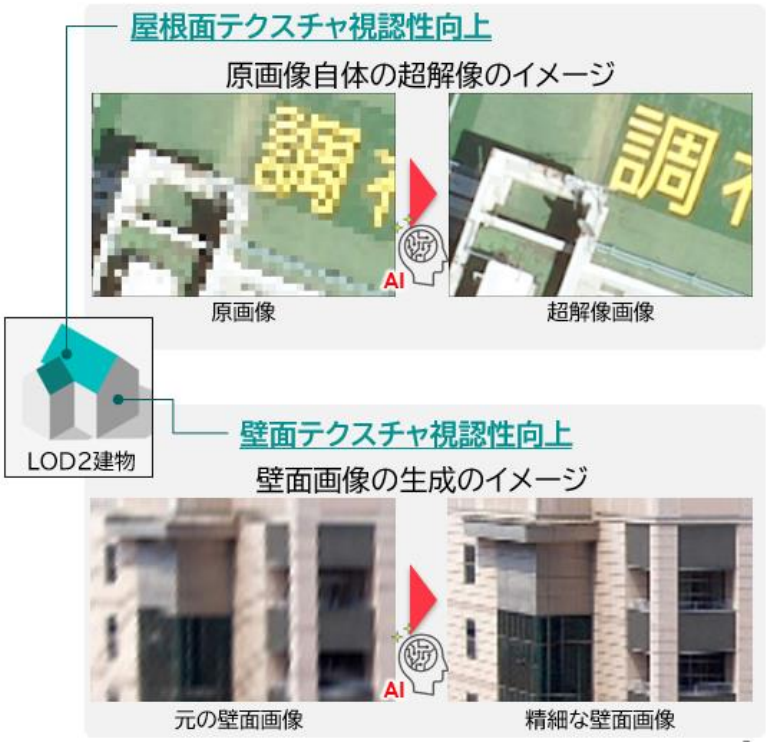


図 2-2　テクスチャ視認性向上による画像生成イメージ

2.1.1. 屋根面

LOD2 建物の屋根面の視認性を向上する手法として超解像技術が挙げられる。入力画像の解像度を上げる超解像技術は、深層学習を用いることで発展し、最近は GAN 用いることで現実の写真に近い自然な画像を生成できるようになった。屋根面の視認性向上では、深層学習による超解像技術のうち、入力画像の 4 倍の超解像化ができ、生成した画像に人工的なノイズ（アーティファクト）が現れにくい特徴を持つ ESRGAN[1]（Enhanced Super-Resolution Generative Adversarial Networks）（図 2-3）と、最新の Transformer と呼ばれる技術を利用した SwinIR[2]（図 2-4）を検討した。

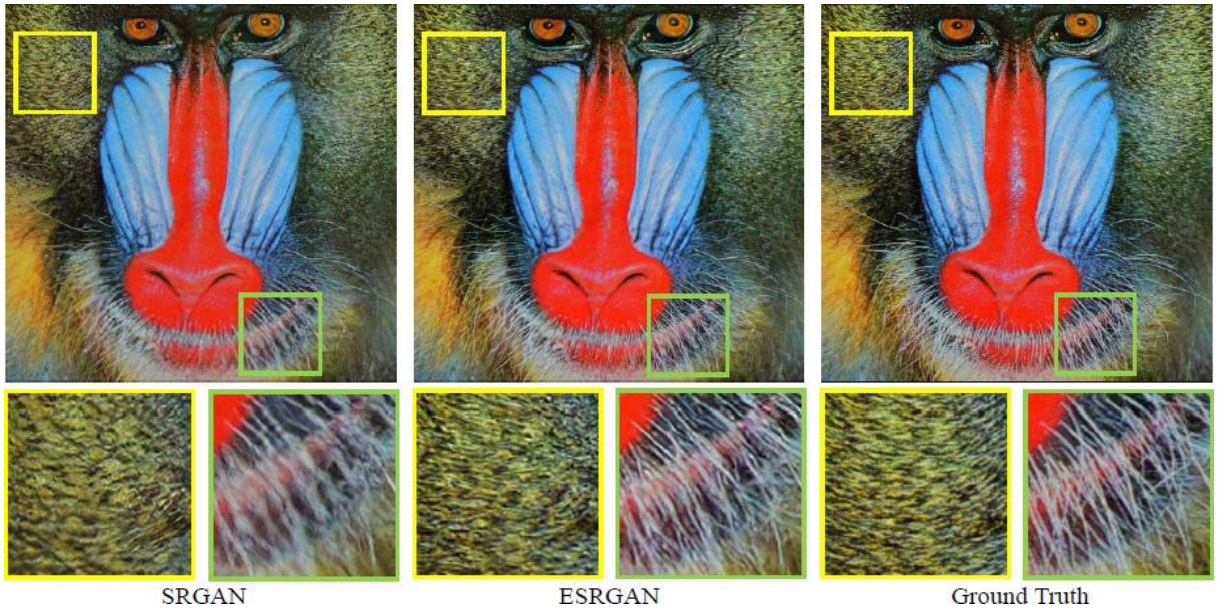


図 2-3　ESRGAN とその他の手法による超解像化の比較

出典：https://arxiv.org/abs/1809.00219

[1] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Chen Change Loy, Yu Qiao, Xiaoou Tang:ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks. arXiv preprint arXiv:1809.00219, 2018.

[2] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, Radu Timofte: SwinIR: Image Restoration Using Swin Transformer. arXiv preprint arXiv:2108.10257, 2021.

図 2-4　SwinIR とその他の手法による超解像化の比較

出典：https://arxiv.org/abs/2108.10257

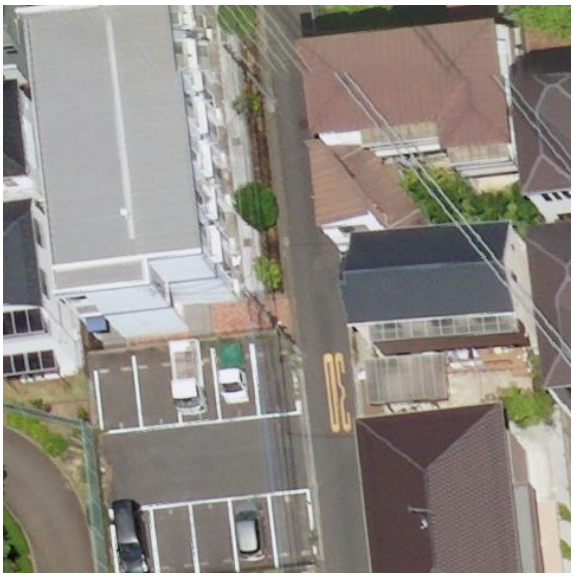
2.1.1.1. ESRGAN による超解像化の検討

深層学習を用いた最初の超解像手法は SRCNN[3]であり、Convolutional Neural Network（畳み込みニューラルネットワーク、以下 CNN）を用いて画像の特徴抽出を行うとともに、生成画像と本物の画像の平均二乗誤差が最小となるように学習を行う手法である。この手法は細かい情報を復元できず、ぼやけた画像が生成される課題があったが、その後に発表された SRGAN[4]（Super-Resolution Generative Adversarial Networks）では、入力画像の 4 倍の超解像化を実現し、GAN を用いることで写真に近い自然な画像を生成できるようになった。一方で、訓練データとテストデータが大きく異なる場合に、本来は生成されるべきではないアーティファクトが表れる課題があった。ESRGAN は、上記の SRCNN と SRGAN の課題を改善する手法として登場した。

本業務では、7.5cm 解像度の航空写真の簡易オルソ画像と、同じ画像を 30cm 解像度へダウンサンプリングした画像のペアを用意して ESRGAN の学習を行った（図 2-5）。学習データを用いた超解像化の試行結果では、地物の細部が精密に再現された画像が生成され、良好な結果が得られたため（図 2-6、図 2-7）、撮影時期が異なる 25cm 解像度の航空写真を用いて超解像化を試行したところ、地物の細部が再現されず入力画像と同様のぼやけた画像が生成された（図 2-8）。

[3] Chao Dong, et al.: Learning a deep convolutional network for image super-resolution, ECCV, pp.184-199, 2014.

[4] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, Wenzhe Shi: Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. arXiv:1609.04802, 2017.



(a)高解像度画像(7.5cm 解像度)

(b)高解像度画像をダウンサンプリングした低解像度画像(30cm 解像度)

図 2-5　ESRGAN の学習に用いた高解像度画像と低解像度画像のペア

(a)入力画像(7.5cm 解像度から 30cm 解像度にダウンサンプリングした画像)

(b)超解像画像(7.5cm 解像度)

図 2-6　ESRGAN による学習データを用いた超解像化の試行結果（１／２）

(c) 真の高解像度画像(7.5cm 解像度)

図 2-7　ESRGAN による学習データを用いた超解像化の試行結果（２／２）

(a)入力画像(25cm 解像度)

(b)超解像画像(6.25cm 解像度)

図 2-8　ESRGAN によるテストデータを用いた超解像化の試行結果（超解像化が失敗した例）

図 2-8 の入力画像は、大気中の水蒸気量が多い時期に高高度から撮影された画像であり、画像がぼやけたり、にじんだりしている。超解像化が失敗した原因として、テスト時の入力画像の画質が学習時のものよりも低いことや、解像度が実態よりも低いことが影響したと考えられたため、学習時に画像を加工することにより、ぼやけた画像やノイズを含んだ画像を追加してデータ拡張（データの水増し）を行うこととともに、テスト時に入力画像の解像度を実態に合わせてダウンサンプリングして試行した。その結果、屋根の棟線が明瞭になり、鮮明な画像を得ることができたものの、画像全体に不自然なノイズ（明るいドットのようなパターン）が生じるようになった（図 2-9）。

(a)入力画像(25cm 解像度から 40cm 解像度にダウンサンプリングした画像)

(b)超解像画像(10cm 解像度)

(c) 真の高解像度画像(7.5cm 解像度)

図 2-9　ESRGAN によるテストデータを用いた超解像化の試行結果
(超解像化がやや改善した例)

以上の検討の結果、ESRGAN ではこれ以上の改善が難しい可能性があるため、より新しい超解像手法を検討することとした。

2.1.1.2. SwinIR による超解像化の検討

ESRGAN は、特徴抽出に CNN が用いられているため、オブジェクトのテクスチャを重視して学習する特徴を持つのに対して、SwinIR を含む Transformer モデルは、オブジェクトの形状を重視して学習する特徴を持つ（図 2-10）。そのため、大量の学習用データを用意した場合は CNN モデル

よりも高い性能を獲得でき、画像に含まれるノイズ等の影響も受けにくいと期待できる。そこで、SwinIR による超解像化を検討した。

(a) 通常の猫の画像
(CNN モデルの識別結果：猫、
Transformer モデルの識別結果：猫)

(b) 象の肌質の猫の画像
(CNN モデルの識結果：象、
Transformer モデルの識別結果：猫)

図 2-10　CNN モデルと Transformer モデルの比較
出典：https://arxiv.org/abs/1811.12231

2.1.1.1 の検討結果を踏まえて、SwinIR についても ESRGAN と同様に、学習時にぼやけた画像やノイズを含んだ画像を追加してデータ拡張を行い、テスト時に入力画像の解像度を実態に合わせてダウンサンプリングして試行した。その結果、ESRGAN とは異なり、不自然なノイズを伴わずに屋根の棟線が明瞭で鮮明な画像を得ることができた（図 2-11）。一方で、一部の入力画像に対しては建物の輪郭が歪んだ超解像画像が生成された（図 2-12）。

(a)入力画像(25cm 解像度から 40cm 解像度にダウンサンプリングした画像)

(b) 真の高解像度画像(7.5cm 解像度)

(c)ESRGAN による超解像画像(10cm 解像度)

(d)SwinIR による超解像画像(10cm 解像度)

図 2-11　ESRGAN と SwinIR によるテストデータを用いた超解像化の試行結果の比較

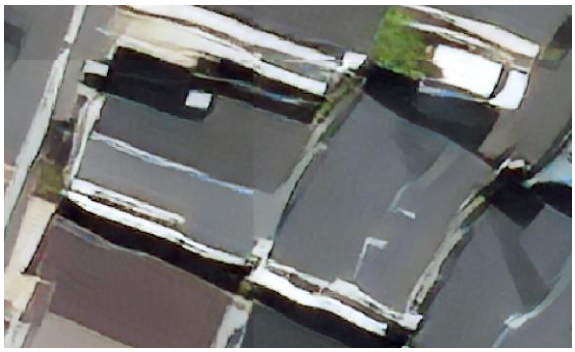
(a)入力画像　　(b)SwinIR による超解像画像

図 2-12　SwinIR による超解像画像に認められる建物の輪郭が歪む現象

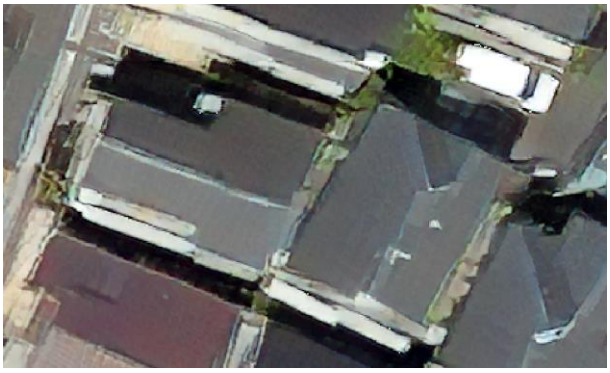
建物の輪郭が歪む原因として、学習時の入力画像にぼやけやにじみのある画像が不足していることや、データ量が不足していることが考えられた。従来のデータ拡張では、ダウンサンプリングやアップサンプリングを繰り返したり、画像にランダムなノイズを付加したりすることで、ぼやけた画像やノイズを含んだ画像を追加している。そこで、従来のデータ拡張に加えて、①GAN で生成したぼやけ・にじみ画像を追加した場合、②画像を回転してバリエーションを増やした場合、③新たな地域の画像を追加した上で画像回転も行った場合を検討した（図 2-13）。その結果、①のモデルでは建物の輪郭の歪みは解消されたものの不自然なノイズがある画像が生成され（図 2-13 (c)）、②と③では不自然なノイズを伴わずに建物の輪郭の歪みが解消された画像を得ることができた（図 2-13 (d)(e)）。なお、③のモデルは②のモデルに比べ、多様な地域を学習しているため汎用性が期待できる。

以上の ESRGAN 及び SwinIR による超解像化の検討結果を踏まえ、屋根面のテクスチャ高解像度化手法については、学習していない地域の画像へ適用した場合に人工的なノイズが現れにくく、汎用性が高い SwinIR を採用し、学習時のデータ拡張手法については、新たな地域の画像を追加した上でぼやけた画像やノイズを含んだ画像を追加し、さらに画像回転を行った手法を採用した。

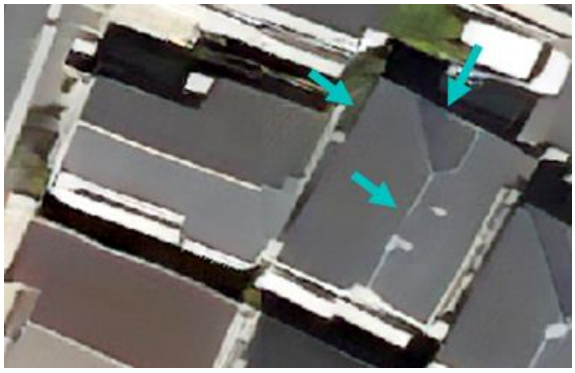
(a)入力画像(25cm 解像度)



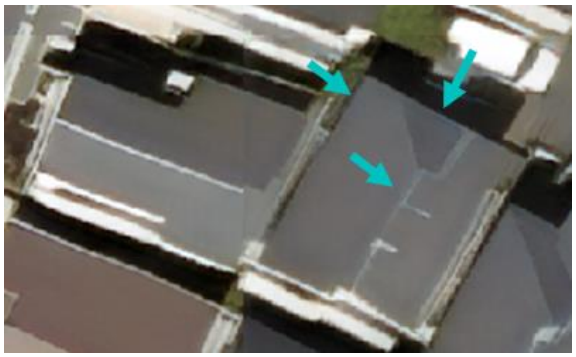
(b)従来のデータ拡張を行った SwinIR モデル



(c)GAN で生成したぼやけ・にじみ画像を追加した SwinIR モデル



(d)画像回転を追加した SwinIR モデル



(e)新たな地域の画像と画像回転を追加した SwinIR モデル

図 2-13　超解像化により建物の輪郭が歪む現象を改善するための検討結果

2.1.2. 壁面

LOD2 建物作成時に使用する一般的なデータセットは、航空写真の直下視画像（地上解像度 20～25 cm 程度）である。このため壁面テクスチャは下記に示す課題が挙げられる。

■ 課題 1

通常は直下視のデータで作成する場合が多いので、壁面は引き伸ばされた画像になりやすい（図 2-14）。地方自治体が保有する航空写真を使用する場合、直下視画像のみのケースが多く、直下視画像を建物壁面に貼付するため、画像がぼやけやすい。

(a)直下視画像を貼付した壁面画像例

(b)斜方視画像を貼付した壁面画像例

図 2-14　直下視/斜方視画像を貼付した壁面テクスチャ

■ 課題 2

屋根面で採用している超解像手法では、低解像度と高解像度のオルソ画像を使用しているため、ペア画像を作れるが、壁面の場合はモデルの位置正確度により、ペア画像となりにくい。このため屋根面で使用している手法（超解像）が適さない。

上記の課題から、「直下視画像を壁面に貼付したテクスチャ画像」から「オブリーク画像を壁面に貼付したテクスチャ画像」に変換する AI モデルを構築することとし、アンペア（非ペア）データセットで学習可能なCycleGAN[5]による画像変換手法を検討した。

[5] Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros: Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. arXiv preprint arXiv:1703.10593, 2017.

2.1.2.1. CycleGAN による画像変換手法の検討

画像生成の手法の一つに GAN によるドメイン変換がある。例として pix2pix[6]のようにあるドメインから別のドメインに画像変換する手法があり、空中写真を地図に変換したり、線画をリアルな写真に変換したりすることができる（図 2-15）。pix2pix ではペア画像を学習する必要があるが、アンペアデータセットで学習可能な手法として CycleGAN がある（図 2-16）。これはアンペアで、ソースドメイン X からターゲットドメイン Y への画像の変換を行うものである(図 2-17)。このため壁面テクスチャ視認性向上手法として CycleGAN を選定した。

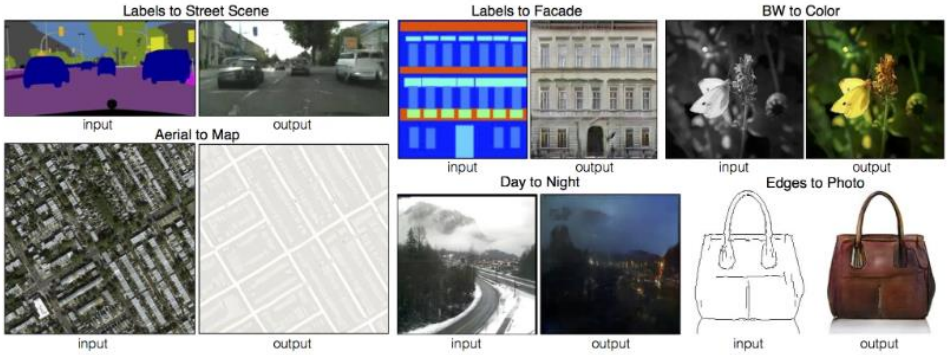


図 2-15　pix2pix による画像変換例

出典: https://arxiv.org/abs/1611.07004

図 2-16　CycleGAN による画像変換例

出典: https://arxiv.org/abs/1703.10593

[6] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros: Image-to-Image Translation with Conditional Adversarial Networks.
arXiv preprint arXiv:1611.07004, 2016.

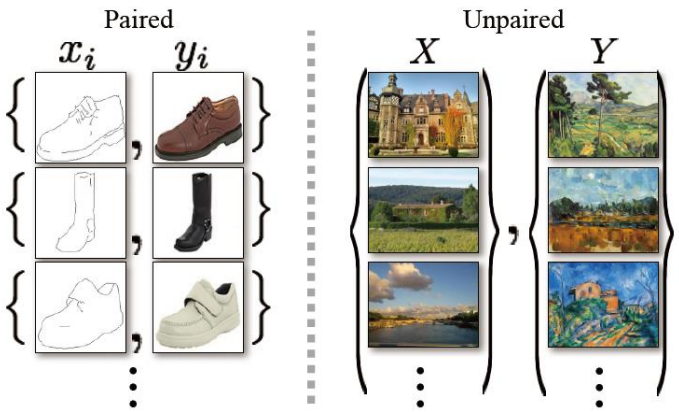


図 2-17　アンペアデータセットのイメージ

出典: https://arxiv.org/abs/1703.10593

図 2-18 に学習モデルの構築フローを示す。CycleGAN による学習は、A 群データセットから B 群データセットへの画像変換を行う学習とした。A 群データセットは直下視-LOD2 の壁面画像とし、B 群データセットはオブリーク-LOD2 の壁面画像とオブリーク画像の切り出し画像とした。図 2-19 に壁面画像の抽出および画像正対化処理フローを示す。LOD2 自動生成ツールで出力されたテクスチャ画像をジオメトリ情報に従い屋根面部と壁面部に分け、壁面部のみを正対化画像に変換した。またオブリーク-LOD2 の壁面画像は、オクルージョンによって壁ではない画像が貼付されているなど良好な画像が少ないことから(図 2-20)、斜方視画像から壁面部を切り出し、射影変換した画像を追加した(図 2-21)。斜方視画像からの画像の切り出しは、歪みが小さい、オクルージョンがない、暗くない、白飛びしていない等の良好な画像とした。

構築した学習済みモデルを用いて、テストデータで画像変換を試行した結果、ボケやブレが改善した壁面らしい画像を生成することができた（図 2-22）。

以上の検討結果より、壁面のテクスチャ高解像度化手法については、CycleGAN による画像変換手法を用いることとし、本項で示した学習モデル構築手法や壁面画像の抽出及び画像正対化処理手法を採用した。

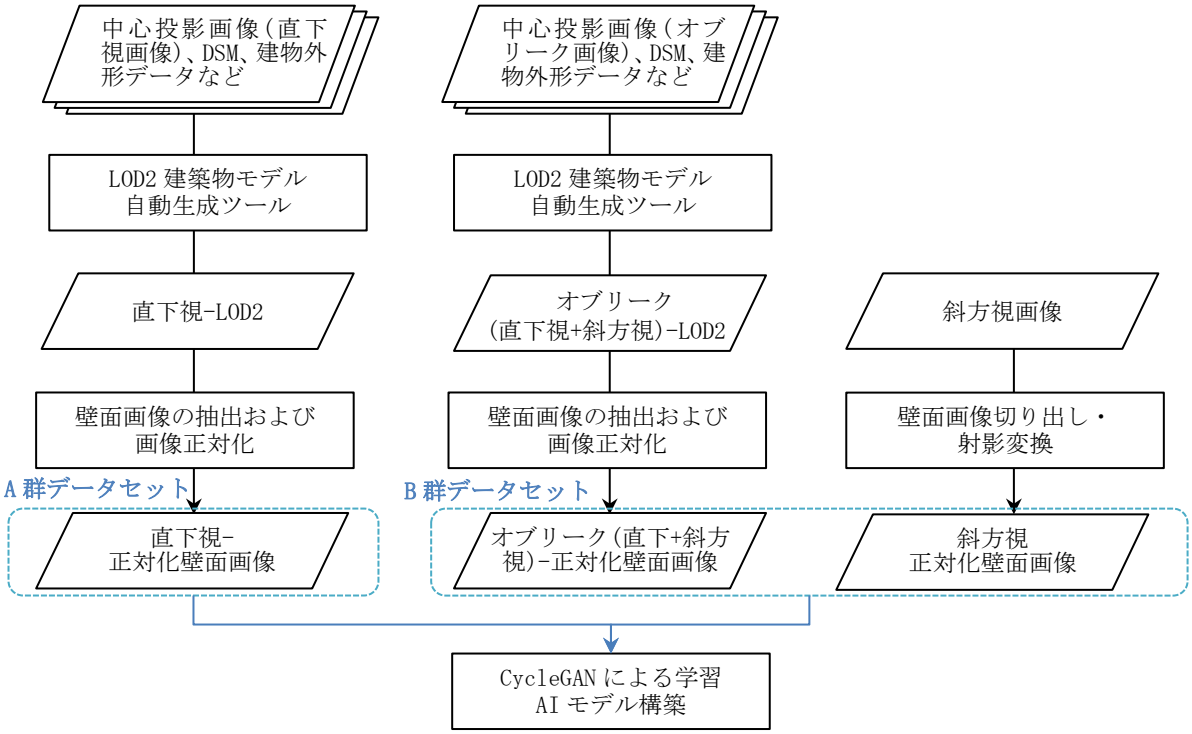


図 2-18　学習モデル構築フロー

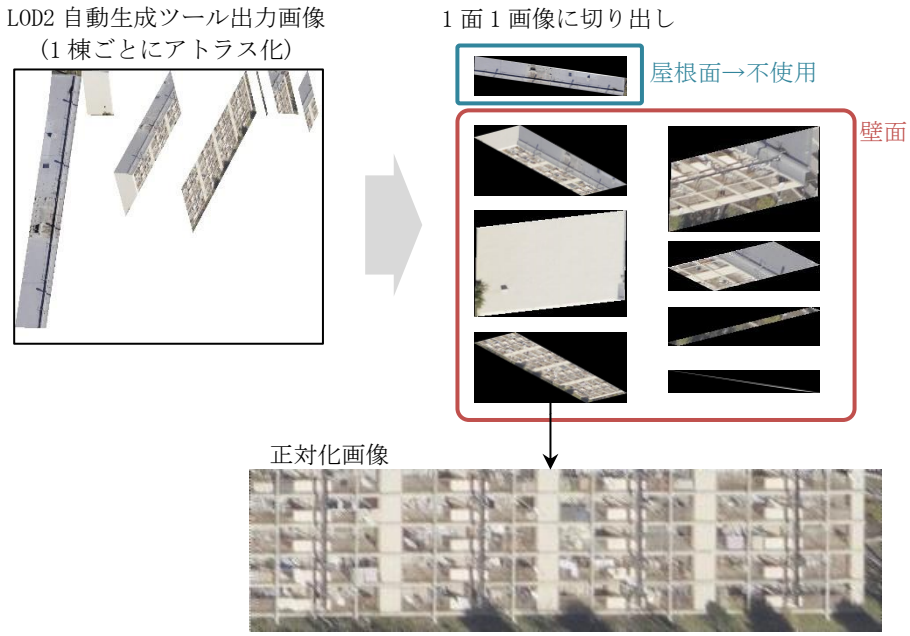


図 2-19　壁面画像の抽出及び画像正対化処理フロー

不良画像
・ オクルージョンにより壁ではない画像が貼付
・ 暗い、鮮明でない
※小さい壁面ほど不良画像が多い傾向にある

図 2-20　オブリーク-LOD2 の壁面画像の例

a)　斜方視画像からの壁面切り出し

b) 射影変換

図 2-21　斜方視画像の壁面切り出し・射影変換

図 2-22　CycleGAN による画像変換の試行結果
（各図の左は変換前画像、右は CycleGAN による変換後画像）

2.2. 建物テクスチャ視認性向上ツール開発

2.2.1. 設計方針

建物テクスチャ視認性向上ツールの設計方針を以下に示す。

- 開発及び運用コストを減らすため、OSS を積極的に利用する。
- 開発成果を OSS として PLATEAU GitHub に公開する。
- 建物テクスチャ視認性向上ツールは屋根面視認性向上ツールと壁面視認性向上ツールで構成される。屋根面視認性向上ツールは、LOD2 建築物モデル自動生成ツールへの入力データである航空写真画像の画像変換を行う。壁面視認性向上ツールは、LOD2 建築物モデル自動生成ツールで出力されたテクスチャ画像の画像変換を行う。

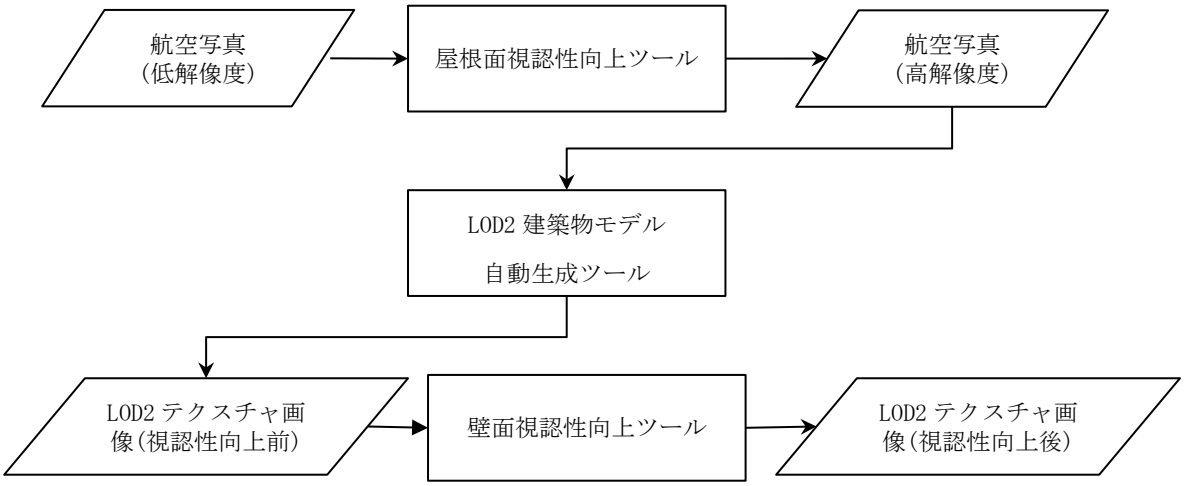


図 2-23　建物テクスチャ視認性向上ツール利用イメージ

2.2.2. 屋根面視認性向上ツール

2.2.2.1. 制約条件

LOD2 建築物モデル自動生成ツール（https://github.com/Project-PLATEAU/Auto-Create-bldg-lod2-tool）を使用し LOD2 建物モデルを作成することを想定している。このため、LOD2 建築物モデル自動生成ツールの入力条件に定められた空中写真諸元を踏まえ、本ツールに入力する画像条件を下記とする。

- 直下視の航空写真（中心投影、地上画素寸法: 25 cm）
- 画像は 24 ビットフルカラー形式

なお入力画像の地上画素寸法は 25cm を想定しているが、PLATEAU において公開されているモデル整備で使用している航空写真解像度は 10～20cm が主流であることを鑑み、屋根面視認性向上ツールでは、地上画素寸法 10～25cm の画像を入力できるものとした。

2.2.2.2. 機能概要

本ツールの主な機能を以下に示す。

・　航空写真（中心投影）を入力とし、地上画素寸法 6.25cm 相当に超解像化された画像を出力する。

2.2.2.3. システム構成

本システムは、①データ分割機能、②高解像度画像生成機能、③データ統合機能の 3 つの機能から構成される（図 2-24）。各機能は独立しており、入力したデータは順に処理される。本システムの入力データは、航空写真（中心投影）である。

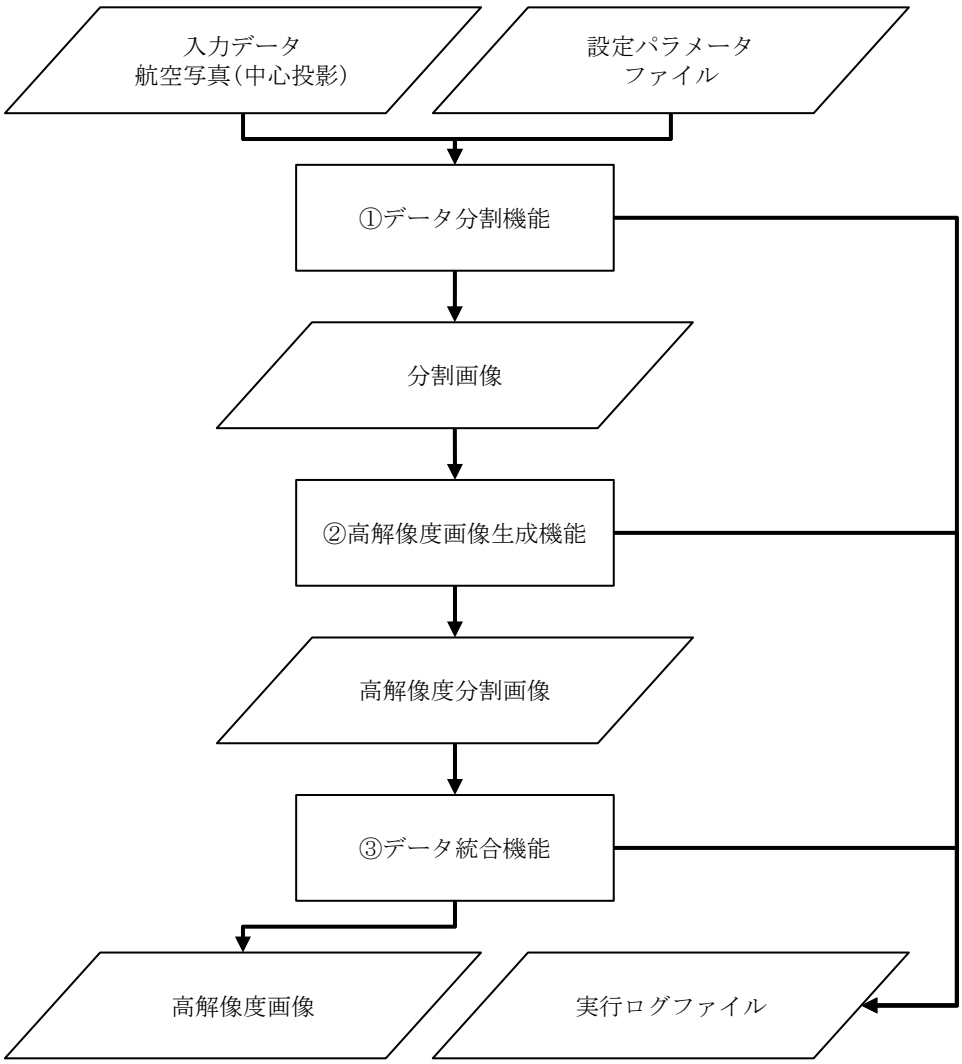


図 2-24　システム構成図

2.2.2.4. システムインタフェース

(1) 外部インタフェース

本ツールに入力するデータは、航空写真（直下視の中心投影画像）と設定パラメータファイルである。設定パラメータファイルは、入出力データの保存場所等を手動編集で記載する。

(2) 内部インタフェース

表 2-1 に、実行時に出力される中間ファイル及び出力ファイルを示す。実行時引数として渡した出力ディレクトリに、以下の 3 ディレクトリに分けてファイルが出力される。

表 2-1　中間及び出力ファイル

ディレクトリ名	種別	内容
split_images	中間出力	データ分割機能により分割された画像ファイルを格納
resolution_split_images	中間出力	高解像度画像生成機能により高解像度化した分割画像ファイルを格納
resolution_images	最終出力	データ統合機能により統合した画像ファイルを格納

(3) ユーザインタフェース

本ツールの実行は、コマンド プロンプトで下記のコマンドを入力して行う。

```
> python CreateSuperResolution.py param.json
```

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

param.json : 設定パラメータファイル

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
```

2.2.2.5. 動作環境

(1) ハードウェア環境

本システムの推奨環境を以下に示す。

推奨環境:

CPU: Intel® Core™ i7-11700F 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 3.7）である。表 2-2 に使用ライブラリを示す。

表 2-2　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	4.63.0	MIT License, Mozilla Public License 2.0	プログレスバーの表示
numpy	1.21.5	BSD License （BSD-3-Clause）	数値計算
Pillow	9.0.1	Historical Permission Notice and Disclaimer	画像の読み書き
PyYAML	6.0	MIT License	yaml ファイルの読み書き
torch	1.12.0	BSD License （BSD-3）	機械学習
torchvision	0.13.0	BSD	機械学習
lightning	1.9.5	Apache Software License （Apache-2.0）	機械学習
opencv-python	4.5.4.60	Apache Software License （Apache 2.0）	画像の読み書き、画像処理
networkx	2.6.3	BSD License	グラフ構造の管理
openmim	0.3.6	Apache 2.0 license	OpenMMLab ライブラリの管理
mmcv	2.0.0rc4	Apache Software License	機械学習
mmedit	1.0.0rc5	Apache Software License （Apache License 2.0）	機械学習
mmengine	0.10.1	Apache Software License （Apache License 2.0）	機械学習

2.2.2.6. ユースケース

本システムを用いた屋根面テクスチャ視認性向上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（直下視の中心投影画像）を準備する。

(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。記載内容は、「2.2.2.8 ファイル仕様(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。

2.2.2.7. モジュール設計

(1) データ分割モジュール

① 概要

航空写真（中心投影）を読み込み、本システムが動作可能な画像サイズに変更する。

② 入力データ

表 2-3 にデータ分割モジュールの入力データの一覧を示す。

表 2-3 データ分割モジュールの入力データ一覧

No.	データ名	入力元	説明
1	航空写真（中心投影）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"InputDir"キーが指定するフォルダ内に存在する航空写真（中心投影）ファイルのパス

③ 出力データ

表 2-4 にデータ分割モジュールの出力データ一覧を示す。

表 2-4 データ分割モジュールの出力データ一覧

No.	データ名	出力先	説明
1	分割画像ファイル	[split_images]フォルダ	・分割された画像ファイル
2	ログメッセージ	[OutputLogDir]フォルダ	・データ分割する際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"フォルダにある実行ログファイルに追記する

④ 処理内容

データ分割モジュールを読み込み、本システムが動作可能な画像サイズを取得する。分割対象の範囲が元の画像サイズを超える場合のみオーバーラップを適応する。取得する分割画像データは以下のとおりである。

・ 画像サイズ：120×120 px

⑤ 例外処理

表 2-5 にデータ分割モジュールの例外処理一覧を示す。

表 2-5 データ分割モジュールの例外処理一覧

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	入力画像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

(2) 高解像度画像生成モジュール

① 概要

機械学習モデルを使用して、分割した画像ファイルの高解像度化を行う。

② 入力データ

表 2-6 に高解像度画像生成モジュールの入力データ一覧を示す。

表 2-6 高解像度画像生成モジュールの入力データ一覧

No.	データ名	入力元	説明
1	分割画像ファイル	[split_images] フォルダ	・分割された画像ファイル
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 出力データ

表 2-7 に高解像度画像生成モジュールの出力データ一覧を示す。

表 2-7　高解像度画像生成モジュールの出力データ一覧

No.	データ名	出力先	説明
1	高解像度　分割画像ファイル	[resolution_split_images]フォルダ	・高解像度化した分割画像ファイル
2	ログメッセージ	[OutputLogDir]フォルダ	・高解像度化する際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"フォルダにある実行ログファイルに追記する

④　処理内容

本処理では、機械学習モデルを使用して、分割した画像ファイルの高解像度化を行う。推論には、超解像手法である SwinIR を使用する。学習には、高解像度画像(地上画素寸法 6.25cm 相当)とこれをダウンサンプリングした低解像度画像(地上画素寸法 25cm 相当)の画像ペアを使用した。表 2-8 に学習データ数を、図 2-25 に学習データの一例を示す。

表 2-8　学習データ数

学習地域※1	画像枚数
東京都※2	24,012
埼玉県	4,476
奈良県	16,928
計	45,416

※1 それぞれ都心部の画像を使用

※2 2 地域を対象

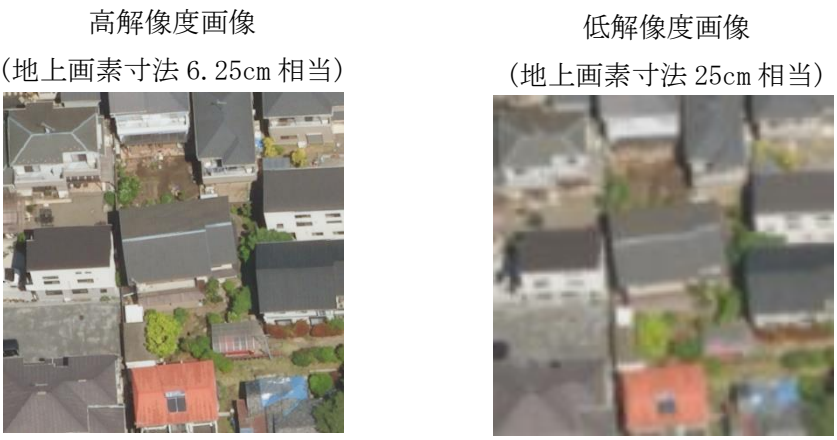


図 2-25　学習データ一例

⑤ 例外処理

表 2-9 に高解像度画像生成モジュールの例外処理一覧を示す。

表 2-9　高解像度画像生成モジュールの例外処理一覧

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) データ結合モジュール

① 概要

高解像度化した分割画像ファイルを統合して、１枚の高解像度の画像ファイルを作成する。

② 入力データ

表 2-10 にデータ統合モジュールの入力データの一覧を示す。

表 2-10　データ統合モジュールの入力データ一覧

No.	データ名	入力元	説明
1	高解像度 分割画像ファイル	[resolution_split_images]フォルダ	・高解像度化した分割画像ファイル

③ 出力データ

表 2-11 にデータ統合モジュールの出力データの一覧を示す。

表 2-11　データ統合モジュールの出力データ一覧

No.	データ名	出力先	説明
1	高解像度画像ファイル	[resolution_images]フォルダ	・統合した高解像度画像ファイル
2	ログメッセージ	[OutputLogDir]フォルダ	・データ統合時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"フォルダにある実行ログファイルに追記する

④ 処理内容

本処理では、高解像度化した分割画像ファイルを統合して、1 枚の高解像度の画像ファイルを作成する。統合対象の範囲が、分割前の画像サイズの 4 倍を超える場合のみオーバーラップを適応する。

⑤ 例外処理

表 2-12 にデータ統合モジュールの例外処理一覧を示す。

表 2-12　データ統合モジュールの例外処理一覧

No.	エラー名称	発生条件	処理内容
1	ファイル書き出しエラー	出力ファイルの書き出しに失敗	エラーログを出力し、モジュールの処理を終了する。

2.2.2.8. ファイル仕様

ファイル仕様を表 2-13 に示す。

表 2-13　ファイル仕様

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（中心投影）	航空写真の中心投影画像
3	実行ログ	実行履歴を記録する

(1) 設定パラメータ

実行時に使用する設定パラメータのファイル仕様を表 2-14 に示す。

表 2-14　設定パラメータファイル仕様

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。

表 2-15 に実行時に使用する設定パラメータを示す。

表 2-15　設定パラメータ

No.	キー名	値形式	説明
1	InputDir	文字列	推論に使用する航空写真(中心投影)が保存されているディレクトリのパス
2	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
3	GSD	数値	航空写真（中心投影）の地上画素寸法
4	【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。デフォルトは[cuda]
5	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 コードと同階層にログファイルを作成する。
6	【任意】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、false とする。

(2) 航空写真

表 2-16 に航空写真（原画像）ファイルの仕様を示す。

表 2-16　航空写真（原画像）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」で指定する。
入力先	データ分割モジュール
出力元	-
特記事項	-

(3) 実行ログ

表 2-17 に実行ログファイルの仕様を示す。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、処理時間等である。

表 2-17　実行ログファイルの仕様

ファイル形式	TXT
ファイル名	main_log.txt
格納フォルダ	設定パラメータで指定する。
入力先	-
出力元	システム全般
特記事項	-

2.2.3. 壁面視認性向上ツール

2.2.3.1. 制約条件

LOD2 建築物モデル自動生成ツール（https://github.com/Project-PLATEAU/Auto-Create-bldg-lod2-tool）を使用し作成された LOD2 建物モデルの壁面部のテクスチャ視認性を向上することを想定している。このため、本ツールに入力する画像条件を下記とする。

- LOD2 建築物モデル自動生成ツールで出力されたテクスチャ画像（1 棟単位アトラス化画像）

2.2.3.2. 機能概要

本ツールの主な機能を以下に示す。

- LOD2 建築物モデル自動生成ツールで出力されたアトラス化テクスチャ画像を入力とし、壁面部のみ視認性を向上させた画像変換されたアトラス化テクスチャ画像を出力する。

2.2.3.3. システム構成

本システムは、①変換対象壁面の抽出及び正対化機能、②壁面画像生成機能、③アトラス化画像再構成機能の 3 つの機能から構成される。各機能は独立しており、入力したデータは順に処理される（図 2-26）。本システムの入力データは、LOD2 建築物モデル自動生成ツールで出力されたテクスチャ画像（1 棟単位アトラス化画像）である。

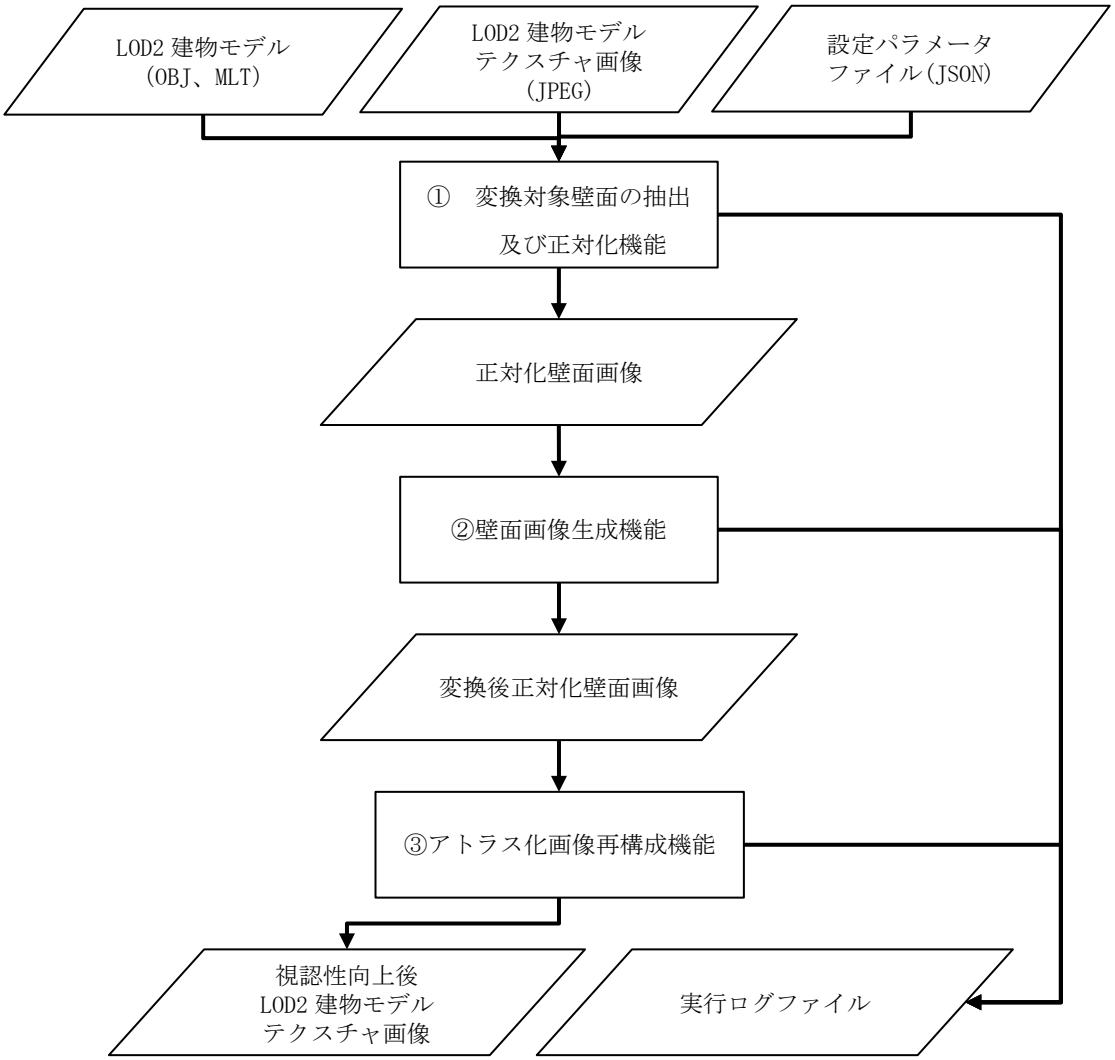


図 2-26　システム構成図

2.2.3.4. システムインタフェース

(1) 外部インタフェース

入力するデータは、LOD2 建築物モデル自動生成ツールで出力されたモデル及びアトラス化テクスチャ画像、パラメータファイルである。パラメータファイルは、入出力データの保存場所や変換対象壁面条件等を記載し、手動編集で標準フォーマットに変換する。

(2) 内部インタフェース

表 2-18 に実行時に出力される中間ファイル及び出力ファイルを示す。実行時引数として渡した出力ディレクトリに以下の 3 ディレクトリに分けてファイルが出力される。

表 2-18　中間及び出力ファイル

ディレクトリ名	種別	内容
processA	中間出力	変換対象壁面の抽出及び正対化機能により生成された正対化壁面画像ファイルを格納
processB	中間出力	壁面画像生成機能により変換された正対化壁面画像ファイルを格納
processC	中間出力	アトラス化画像再構成機能で正対化壁面画像ファイルを元の形に変換した壁面画像ファイルを格納
XXX_appearance （「XXX」はファイル識別子）	最終出力	アトラス化画像再構成機能により生成された視認性向上後 LOD2 建物モデルテクスチャ画像ファイルの格納

(3) ユーザインタフェース

本ツールの実行は、コマンド プロンプトで下記のコマンドを入力して行う。

> python main.py param.json

main.py: 本システムの Python コード

param.json ： 設定パラメータファイル

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

2.2.3.5. 動作環境

(1) ハードウェア環境

本システムの推奨環境を以下に示す。

推奨環境:

CPU: Intel® Core™ i7-11700F 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 3.7）である。使用ライブラリは屋根面テクスチャ視認性向上ツールと同様である（表 2-2）。

2.2.3.6. ユースケース

本システムを用いた壁面テクスチャ視認性向上の手順は以下のとおりである。

(1) 入力データの準備

・ LOD2 建築物モデル自動生成ツールで LOD2 建物モデルを作成し、出力された OBJ ファイル、MTL ファイル及びアトラス化テクスチャ画像を準備する。

(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。記載内容は、「2.2.3.8 ファイル仕様(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。

2.2.3.7. モジュール設計

(1) 変換対象壁面の抽出及び正対化モジュール

① 概要

LOD2 建物モデル（OBJ、MTL）及びテクスチャ画像を読み込み、変換対象壁面テクスチャ画像のみ抽出し正対化する。

② 入力データ

表 2-19 に変換対象壁面の抽出及び正対化モジュールの入力データ一覧を示す。

表 2-19　変換対象壁面の抽出及び正対化モジュールの入力データ一覧

No.	データ名	入力元	説明
1	LOD2 建物モデ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"InputDir"キーが指定するフォルダ内に存在する LOD2 建物モデルファイルのパス

③ 出力データ

表 2-20 に変換対象壁面の抽出及び正対化モジュールの出力データ一覧を示す。

表 2-20　変換対象壁面の抽出及び正対化モジュールの出力データ一覧

No.	データ名	出力先	説明
1	正対化壁面画像ファイル	[processA]フォルダ	・正対化された壁面画像ファイル
2	ログメッセージ	[OutputLogDir]フォルダ	・壁面抽出及び正対化する際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"フォルダにある実行ログファイルに追記する

④ 処理内容

LOD2 建築物モデル自動生成ツールで出力された OBJ・MTL ファイル及びテクスチャ画像を読み込み、以下に示す流れで正対化壁面画像を生成する。なお「3) 統合壁面の正対化」で変換対象画像抽出の際に設定された上限値・下限値はユーザーにより変更可能な数値である。

1) 建物面の法線による壁面判定

- OBJ ファイルから建物の三次元頂点座標を読み込む
- 取得した頂点画像から各面の法線ベクトルを算出する
- 水平面に対し直角である法線ベクトルの面を壁面とする

2) 分割された壁面の統合

- ①で算出した法線ベクトルを用いて、下記の条件を全て満たす三角形面を同一平面上にあるとして統合する
  - a) 面の法線ベクトル間の内積の絶対値が 0.999 以上
  - b) 面間で共有する頂点が存在する

3) 統合壁面の正対化

- テクスチャ画像の UV 座標及び面の頂点座標から射影変換式を取得する
- 上記式をテクスチャ画像に適用する
- UV 座標範囲外をマスク処理し対象壁面のみの正対化画像を生成する
- 縦横ともに下限値以上、上限値以下の画像のみを抽出する（下限値：16px、上限値：256px）

(2) 壁面画像生成モジュール

① 概要

機械学習モデルを使用して正対化壁面画像ファイルを壁面らしい画像に変換する。

② 入力データ

表 2-21 に壁面画像生成モジュールの入力データ一覧を示す。

表 2-21　壁面画像生成モジュールの入力データ一覧

No.	データ名	入力元	説明
1	正対化壁面画像ファイル	[processA]フォルダ	・正対化された壁面画像ファイル
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 出力データ

表 2-22 に壁面画像生成モジュールの出力データ一覧を示す。

表 2-22　壁面画像生成モジュールの出力データ一覧

No.	データ名	出力先	説明
1	画像変換後　正対化壁面画像ファイル	[processB]フォルダ	・壁面らしい画像に変換された正対化壁面画像ファイル
2	ログメッセージ	[OutputLogDir]フォルダ	・壁面画像を生成する際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"フォルダにある実行ログファイルに追記する

④ 処理内容

機械学習モデルを使用して正対化壁面画像ファイルを壁面らしい画像に変換する。画像変換は CycleGAN を使用する。表 2-23 に学習に使用したデータ概要・データ数を、図 2-27 に学習データの一例を示す。

表 2-23　学習データ概要・データ数

	データ概要	画像枚数
変換前画像	・ LOD2 建物を作成する際の標準的なデータセットを想定 ・ 直下視-LOD2 の壁面画像	4791 枚
変換後画像	・ オブリーク-LOD2 の壁面画像 ・ オブリーク画像の切り出し画像	4791 枚

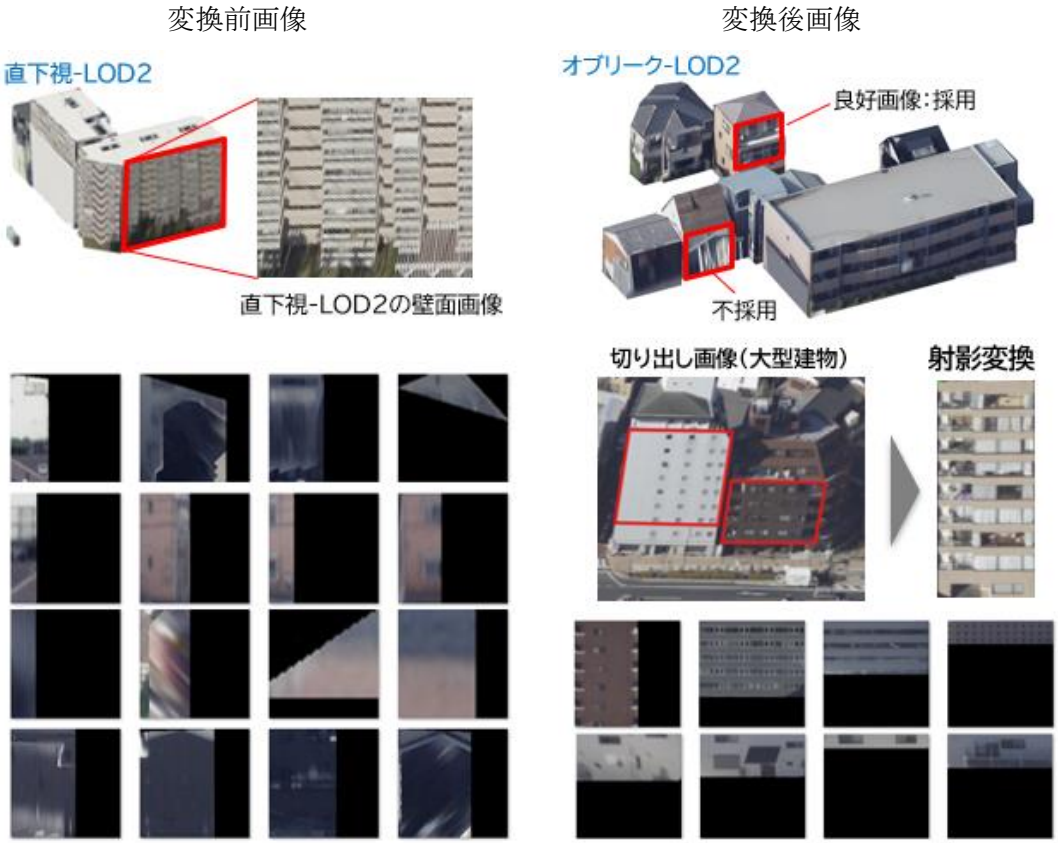


図 2-27　学習データ一例

(3) アトラス化画像再構成モジュール

① 概要

壁面画像生成モジュールで生成された正対化壁面画像ファイルを変換対象壁面の抽出及び正対化モジュールで算出した射影変換式を用いて入力時のアトラス化画像に再構成する。

② 入力データ

表 2-24 にアトラス化画像再構成モジュールの入力データ一覧を示す。

表 2-24　アトラス化画像再構成モジュールの入力データ一覧

No.	データ名	入力元	説明
1	画像変換後　正対化壁面画像ファイル	[processB]フォルダ	・壁面らしい画像に変換された正対化壁面画像ファイル

③ 出力データ

表 2-25 にアトラス化画像再構成モジュールの出力データ一覧を示す。

表 2-25　アトラス化画像再構成モジュールの出力データ一覧

No.	データ名	出力先	説明
1	画像変換後 アトラス化画像ファイル	[XXX_appearance] フォルダ (「XXX」はファイル識別子)	変換後壁面画像を再構成したアトラス化画像ファイル
2	ログメッセージ	[OutputLogDir] フォルダ	・アトラス化画像の再構成時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"フォルダにある実行ログファイルに追記する

④　処理内容

壁面画像生成モジュールで生成された正対化壁面画像ファイルを変換対象壁面の抽出及び正対化モジュールで算出した射影変換式を用いて入力時のアトラス化画像に再構成する。

2.2.3.8. ファイル仕様

ファイル仕様を表 2-26 に示す。

表 2-26　ファイル仕様

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	LOD2 建物モデル	LOD2 建築物モデル自動生成ツールで出力されたテクスチャ付き LOD2 建物モデル（OBJ ファイル、MLT ファイル、JPEG ファイル）
3	実行ログ	実行履歴を記録する

(1) 設定パラメータ

実行時に使用する設定パラメータファイル仕様を表 2-27 に示す。

表 2-27　設定パラメータファイル仕様

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。

表 2-28 に実行時に使用する設定パラメータを示す。

表 2-28　設定パラメータ

No.	キー名	値形式	説明
1	InputDir	文字列	LOD2 建物モデルファイルのパス
2	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
3	【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。デフォルトは[cuda]
4	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 コードと同階層にログファイルを作成する。
5	【任意】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、false とする。

(2) LOD2 建物モデル

表 2-29～表 2-31 に LOD2 建築物モデル自動生成ツールで出力されたテクスチャ付き LOD2 建物モデル（OBJ ファイル、MLT ファイル、JPEG ファイル）ファイル仕様を示す。

表 2-29　LOD2 建物モデル（OBJ ファイル）ファイルの仕様

ファイル形式	OBJ
ファイル名	XXX.obj、「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」で指定する。
入力先	-
出力元	-
特記事項	-

表 2-30　LOD2 建物モデル（MLT ファイル）ファイルの仕様

ファイル形式	MLT
ファイル名	XXX.mtl、「XXX」はファイルの識別子となる。
格納フォルダ	OBJ ファイルと同じディレクトリ
入力先	-
出力元	-

特記事項	「設定パラメータ」で指定された OBJ ファイルに記述された識別子をキーに参照

表 2-31　LOD2 建物モデル（JPEG ファイル）ファイルの仕様

ファイル形式	JPEG
ファイル名	XXX.jpg、「XXX」はファイルの識別子となる。
格納フォルダ	MLT ファイルに記述されたパスフォルダ
入力先	-
出力元	-
特記事項	-

原則として LOD2 建築物モデル自動生成ツールで出力されたテクスチャ付き LOD2 建物モデルのファイル構成を想定している。

〈ファイル構成例〉

```
obj （"InputDir"で指定されたフォルダ）
  └ XXX
      └ XXX.mtl
      └ AAA.obj
      └ BBB.obj
XXX_appearance （テクスチャ画像格納フォルダ）
  └ AAA.jpg
  └ BBB.jpg
```

(3) 実行ログ

表 2-32 に実行ログファイルの仕様を示す。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、処理時間等である。

表 2-32　実行ログファイルの仕様

ファイル形式	TXT
ファイル名	main_log.txt
格納フォルダ	設定パラメータで指定する。
入力先	-
出力元	システム全般
特記事項	-

2.3. 建物テクスチャ視認性向上ツール検証

2.3.1. 検証概要

建物テクスチャ視認性向上ツールの検証は下記の 2 点について実施した。

i) LOD2 建築物モデル自動生成ツールで生成した LOD2 建物テクスチャとの比較

ii) 公開 LOD2 建物テクスチャとの比較

2.3.2. 検証データ

2.3.2.1. LOD2 建築物モデル自動生成ツールで生成した LOD2 建物テクスチャとの比較

検証対象地は、LOD2 建築物モデル自動生成ツールで生成した LOD2 建物が公開されている地域である神奈川県 川崎市 川崎区・幸区の一部とした（図 2-28）。検証に使用した LOD2 建物を図 2-29 に示す。航空写真地上解像度は 20cm、評価対象棟数は 544 棟である。

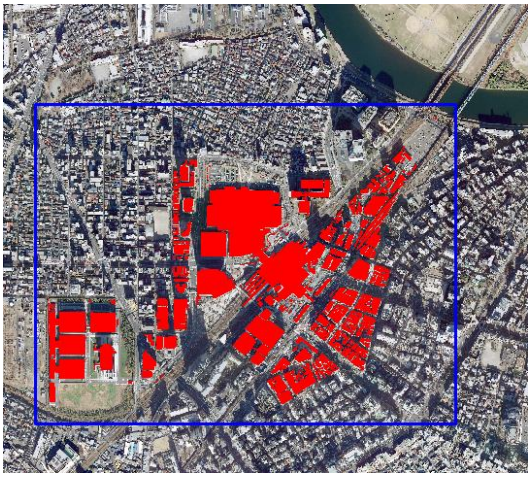


図 2-28 検証対象地（川崎市）

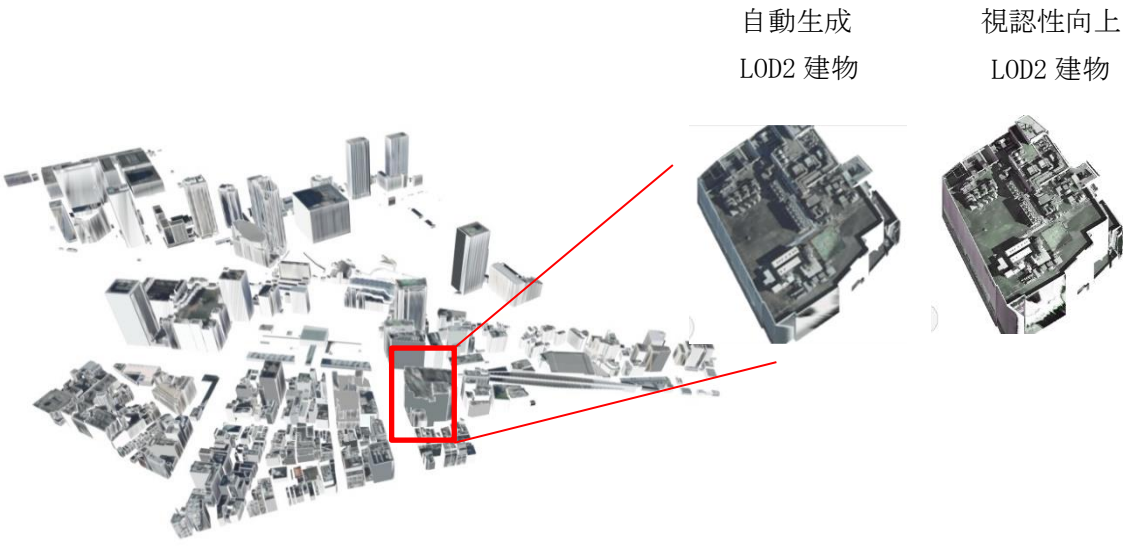


図 2-29 検証データ（川崎市）

2.3.2.2. 公開 LOD2 建物テクスチャとの比較

対象地域は、公開されている広範囲にテクスチャ付き LOD2 が整備されている岐阜市の一部である。検証データは、屋根面視認性向上ツールを用いて航空写真の画像変換を行い、これを用いて LOD2 建築物モデル自動生成ツールでテクスチャ付き LOD2 建物を生成後に、壁面視認性向上ツールで壁面を画像変換した LOD2 建物とした検証に使用した LOD2 建物を図 2-31 に示す。航空写真地上解像度は 12cm、評価対象棟数は 512 棟である。

図 2-30　検証対象地（岐阜市）

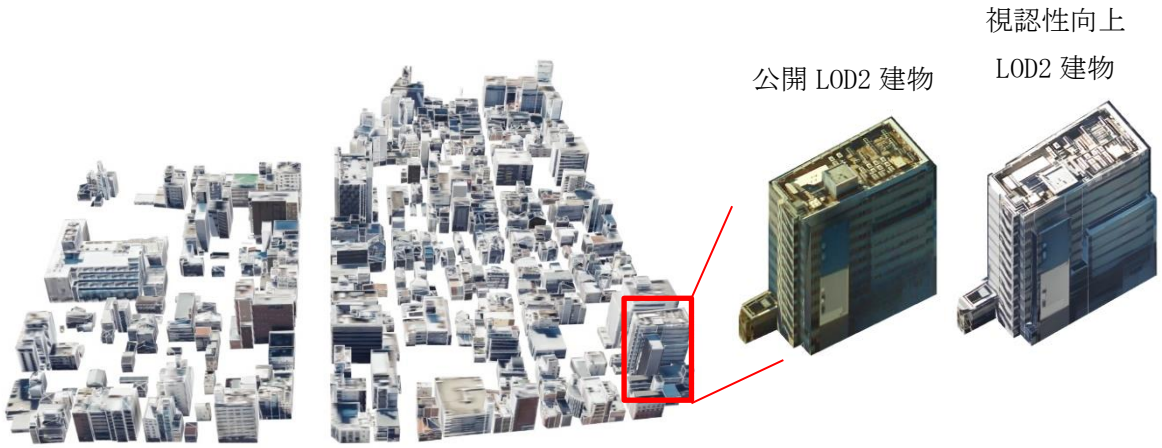


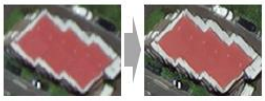
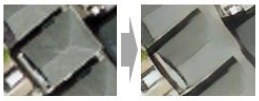
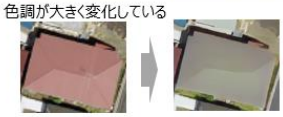
図 2-31　検証データ（岐阜市）

2.3.3. 屋根面の検証

2.3.3.1. 評価項目・基準

表 2-33 に記載した 3 項目について、ランク A～C の評価基準を設定した。検証は各 LOD2 建物を鳥瞰的に 1 方向から見て比較した。

表 2-33　屋根面における評価項目・基準

No.	評価着眼点	ランクA 元画像と比較し、鮮明な画像になっている	ランクB 元画像と比較し、鮮明ではあるが不自然なテクスチャを含む	ランクC 元画像と比較し、鮮明になっていない・劣化している
1	ボケ・ブレ改善、エッジ鮮明化	ボケ・ブレが改善、輪郭が鮮明化 	ボケ・ブレがやや改善・エッジにやや歪みあり 	エッジが鮮明でない・大きい歪みがある 
2	色調の改善	色調が変化していない 	–	色調が大きく変化している 
3	屋根面らしい画像の生成	良好な画像生成 	やや不自然なテクスチャ 	元画像と明らかに異なる画像を生成 

※ LOD2 建築物モデル自動生成ツールにおいて LOD2 モデルが生成されない建物やテクスチャが貼付されない建物は評価対象外とした

2.3.3.2. 評価結果

(1) LOD2 建築物モデル自動生成ツールで生成した LOD2 建物テクスチャとの比較

図 2-32 に視認性評価結果を示す。ボケ・ブレ改善、エッジの鮮明化では A ランクが 87%、色調では A ランクが 98%となった。約 84%で屋根面らしい画像が生成された。

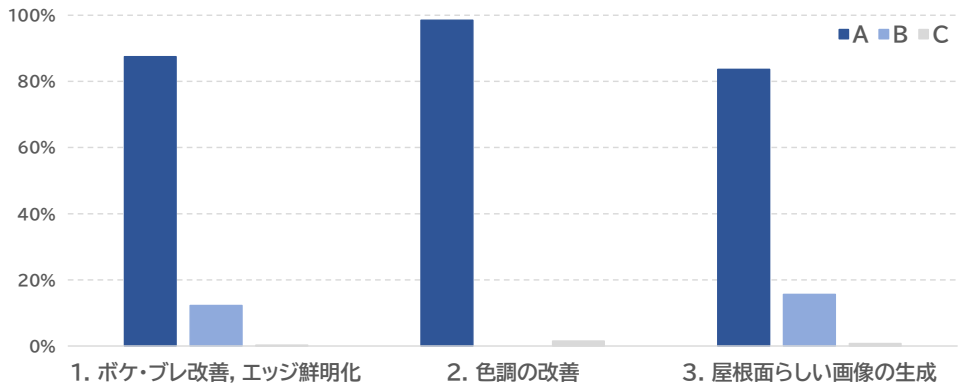


図 2-32　屋根面の LOD2 建築物モデル自動生成ツールで生成した LOD2 建物テクスチャとの比較（川崎市）

(2) 公開 LOD2 建物テクスチャとの比較

図 2-33 に視認性評価結果を示す。ボケ・ブレ改善、エッジの鮮明化では A ランクが 70%、色調では A ランクが 93%となった。約 79%で屋根面らしい画像が生成された。

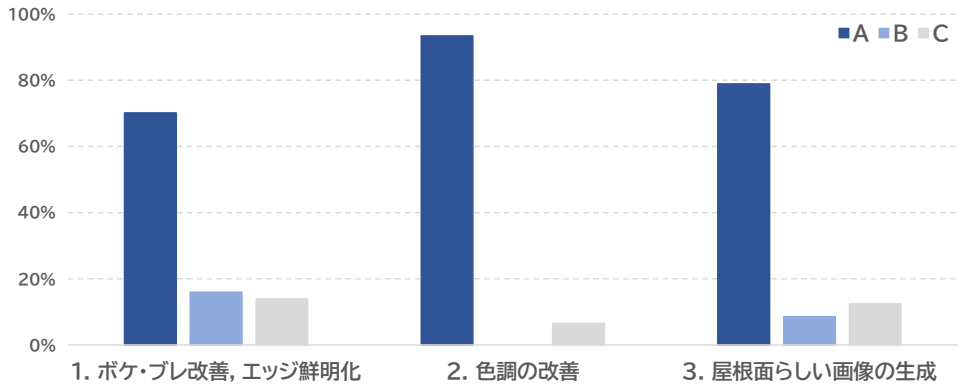
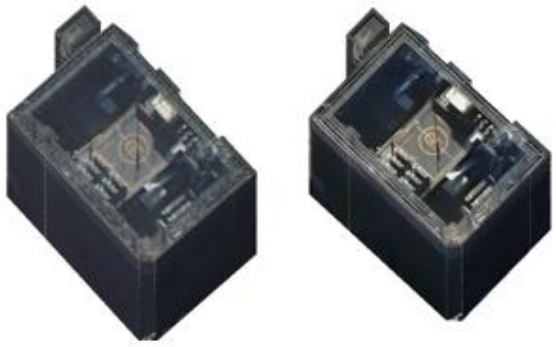
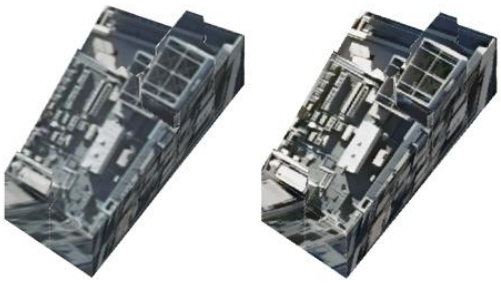


図 2-33　屋根面の公開 LOD2 建物テクスチャとの比較（岐阜市）

2.3.3.3. 視認性向上 LOD2 建物例

図 2-34 に屋根面視認性向上 LOD2 建物例を示す。

<table>
<tr><td>  </td><td>  </td></tr>
<tr><td>左：視認性向上前 LOD2※1、右：視認性向上後 LOD2</td><td>左：視認性向上前 LOD2※1、右：視認性向上後 LOD2</td></tr>
<tr><td>  </td><td>  </td></tr>
<tr><td>左：視認性向上前 LOD2※1、右：視認性向上後 LOD2</td><td>左：視認性向上前 LOD2※1、右：視認性向上後 LOD2</td></tr>
</table>

※ LOD2 建築物モデル自動生成ツールで生成した LOD2

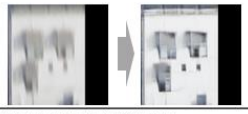
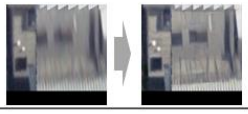
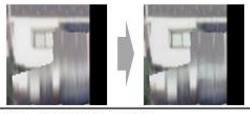
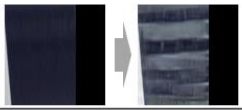
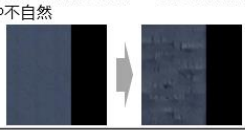
図 2-34　屋根面視認性向上 LOD2 建物例

2.3.4. 壁面の検証

2.3.4.1. 評価項目・基準

表 2-34 に記載した 3 項目について、ランク A～C の評価基準を設定した。検証は各 LOD2 建物側面を 4 方向から見て比較した。

表 2-34 壁面の評価項目・基準

No.	評価項目	ランクA 元画像と比較し、鮮明な画像になっている	ランクB 元画像と比較し、やや鮮明な画像になっている/ほぼ変化がない	ランクC 元画像と比較し、劣化している
1	ボケ・ブレ改善、エッジ鮮明化	ボケ・ブレが改善/エッジが鮮明化 	ボケ・ブレがやや改善/変化なし 	ボケ・ブレが改善されず劣化 
2	色調の改善	色調が改善している/変化なし 	–	色調が大きくが変化している 
3	壁面らしい画像の生成	良好に壁面らしい画像を生成 	壁面らしい画像を生成しているが、テクスチャがやや不自然 	壁面ではない画像を生成 

※ LOD2 建築物モデル自動生成ツールにおいて LOD2 モデルが生成されない建物、テクスチャが貼付されない建物、ジオメトリが大きく変わっている建物及び壁面でない画像が貼付されている建物は対象外とした

2.3.4.2. 評価結果

(1) LOD2 建築物モデル自動生成ツールで生成した LOD2 建物テクスチャとの比較

図 2-35 に視認性評価結果を示す。ボケ・ブレ改善、エッジの鮮明化では、A ランクが 51%となり約半数の壁面が改善された。色調では A ランクが 95%となった。約 62%で壁面らしい画像が生成された。

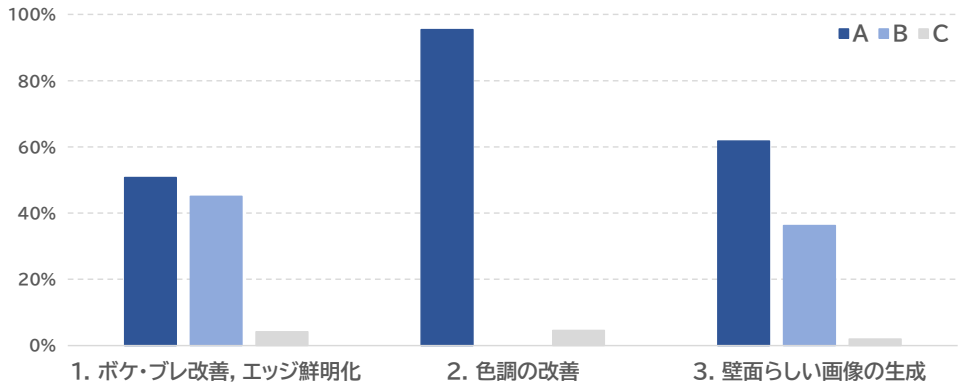


図 2-35 壁面の LOD2 建築物モデル自動生成ツールで生成した LOD2 建物テクスチャとの比較（川崎市）

(2) 公開 LOD2 建物テクスチャとの比較

図 2-36 に視認性評価結果を示す。ボケ・ブレ改善、エッジの鮮明化では、A ランクが 46%、B ランクが 51%となり若干 B ランクが上回る結果となった。色調では A ランクが 97%となった。約 37%で壁面らしい画像が生成された。

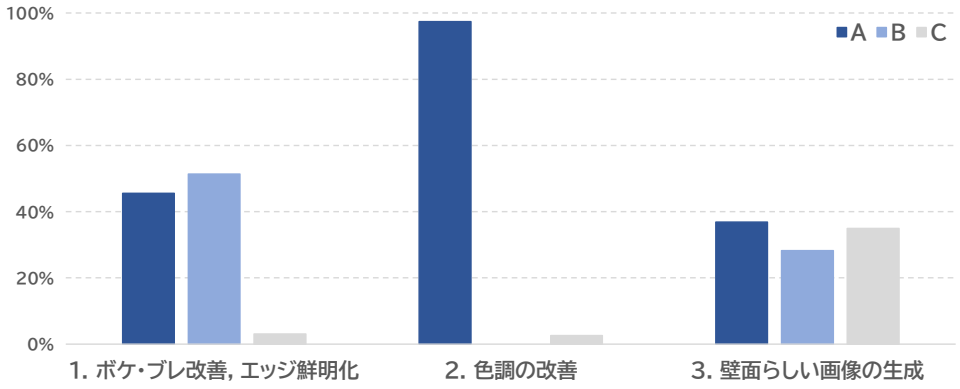
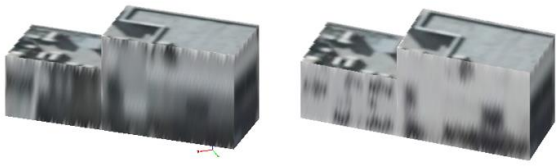


図 2-36　壁面の公開 LOD2 建物テクスチャとの比較（岐阜市）

2.3.4.3. 視認性向上 LOD2 建物例

図 2-37 に壁面視認性向上 LOD2 建物例を示す。

	
左：視認性向上前 LOD2※1、右：視認性向上後 LOD2	左：視認性向上前 LOD2※1、右：視認性向上後 LOD2
	
左：視認性向上前 LOD2※1、右：視認性向上後 LOD2	左：視認性向上前 LOD2※1、右：視認性向上後 LOD2

※ LOD2 建築物モデル自動生成ツールで生成した LOD2

図 2-37　壁面視認性向上 LOD2 建物例

2.4. 調査検討のまとめ

LOD2 建物テクスチャ視認性向上ツールの開発を行った。屋根面の視認性向上の検証を行った結果、LOD2 建築物モデル自動生成ツールで生成した LOD2 建物テクスチャとの比較では、各評価項目で建物の 8～9 割以上が改善し、また公開 LOD2 建物テクスチャとの比較では建物の 7～9 割以上に改善が見られ、ツールの有効性が確認できた。壁面の視認性向上の検証を行った結果、LOD2 建築物モデル自動生成ツールで生成した LOD2 建物テクスチャとの比較では、ボケ・ブレ改善、エッジの鮮明化で評価した壁面の約半数が改善し、6 割以上で壁面らしい画像を生成することができた。一方で公開 LOD2 建物テクスチャとの比較では、ボケ・ブレ改善、エッジの鮮明化及び壁面らしい画像生成でやや改善の割合が低かった。開発した壁面視認性向上ツールは、変換対象サイズの上限値及び下限値を設定している。検証地域である岐阜市（岐阜駅北部）は比較的小さな建物が密集しており、変換対象外となった壁面が多数あり、結果としてランク B が相対的に増加したと考えられる。

なお、壁面画像生成は、GAN の中でもスタイル変換手法を採用しているため、入力画像の壁面形状から大きくテクスチャが変わる場合がある点に留意する必要がある。

3. 描画パフォーマンス向上施策の調査検討

本章では 3D 都市モデルの描画パフォーマンス向上施策の調査検討に関して記載する。

3.1. 建物テクスチャに関する技術調査

描画性能向上を検討する事前調査として、既往の建物テクスチャについて、調査を実施した。

3.1.1. 画像ファイル形式・サイズ

画像ファイル形式は JPEG、PNG、TIFF 形式で作成されている。様々なソフトでの利用を想定した汎用性や Web での利用を考慮すると JPEG、PNG が推奨される。

1 ファイルは 1 つの建物相当となっていて。画像サイズは、1024 ピクセル以下が多いが、大規模な建物ではそれ以上となり、最大で 1 辺が 8192 ピクセルのケースがあった。Web 環境での利用及び描画性能を考慮すると 2048 ピクセル以下とすることが推奨される。一方で実際に使用されているテクスチャサイズは 4098 ピクセルも一定数あり、1 棟を 1 枚の画像でアトラス化することも考慮することや各種ソフトウェアでの利用を考慮した上で 4098 ピクセルが上限と考えられる。

各辺長を 2 の累乗（8、64、128、256、512、1024、2048、4096 ピクセル）としているケースが多く、これは描画時の浮動小数点処理を行う上で、2 の累乗とすることが効率的であることに起因する。表 3-1 に自治体ごとのテクスチャの例を示す。また国際航業(株)の過年度成果を対象としたテクスチャ構造に関する調査結果を表 3-2 に示す。ここからも 1 棟ごとにアトラス化することが主流となっていることが確認できる。

表 3-1　自治体ごとのテクスチャの例

項目	京都市	加賀市	岡崎市	南相馬市
LOD2 メッシュ数	49	88	27	37
1 メッシュあたりのテクスチャ総容量（最大値）［KB］	22,297	7,225	6,035	92,993
1 棟あたりの平均ファイルサイズ［KB］	4.3	5.8	2.1	20.1
1 棟あたりの平均的な画像解像度	256×128	256×128	96×96～128×128	256×256

表 3-2 過年度成果を対象としたテクスチャ構造に関する調査結果[7]

過年度成果　調査結果の抜粋									
作成業者		国際航業	国際航業	パスコ	パスコ	アジア航測	アジア航測	朝日航洋	中日本航空
整備自治体		茂原市	横浜市	横浜市	南相馬市	岐阜市	岡崎市	武雄市	白石町
テクスチャサイズ	最小	32×64	64×64 32×128	15×119	16×32	8×8	8×8	16×16	512×512
	最大	4,096×4,096	4,096×4,096	8,192×8,192	4,096×4,096	2,048×3,072	1,536×1,024	1,024×1,024	512×512
画像 1 辺		2 の累乗	2 の累乗	整数	2 の累乗	2 の倍数	2 の倍数	2 の累乗	2 の累乗

[7] PLATEAU Technical Report　3D 都市モデルの標準仕様の有用性に関する調査（2023 年度）表 2－13 引用

過年度成果　調査結果の抜粋								
作成業者	国際航業	国際航業	パスコ	パスコ	アジア航測	アジア航測	朝日航洋	中日本航空
整備自治体	茂原市	横浜市	横浜市	南相馬市	岐阜市	岡崎市	武雄市	白石町
アトラス画像に含まれる建物単位	1 棟	1 棟	1 棟	1 棟	1 棟の同一境界面	1 棟	複数棟	1 棟
アトラス化の基点	左上	左上	左下	左下	左下	左下	左上	左下
撮影画像変換の有無	変換なし	変換あり	変換なし	変換なし	変換なし	変換なし	変換なし	変換あり
最大余白含有率	86.10%	66.00%	81.30%	84.80%	99.00%	93.20%	70.30%	64.10%

高層ビルなどは形状がシンプルなため、ポリゴン数は少ないが、1 棟のもつ画像サイズは大きくなるため、高層を含み・建物密集度が高い都市部ではエリア当たりの画像容量が大きくなる傾向がある。図 3-1 に建築物 1 棟分の画像ファイルの例を示す。

図 3-1　大型の建築物の 1 棟分の画像ファイル例 2048×1024 （左）、小規模な建築物の 1 棟分の画像ファイル例 64×32（右）

3.1.2. 建物テクスチャ

3.1.2.1. CityGML のテクスチャ定義

建築物モデルのテクスチャの貼付は、建築物を構成する面の頂点に対して、対応する画像ファイルの UV 座標を定義し、ひも付けを行っている。CityGML 形式において、テクスチャは以下の記載で定義される。

```
<app:surfaceDataMember>
  <app:ParameterizedTexture>
    <app:imageURI>53394525_bldg_6697_appearance/17790.jpg</app:imageURI>
    <app:mimeType>image/jpg</app:mimeType>
    <app:target uri="#poly_SJLM0346_p16926_2">
      <app:TexCoordList>
        <app:textureCoordinates ring="#line_SJLM0346_p16926_2">0.8524984 （略）0.7636788</app:textureCoordinates>
      </app:TexCoordList>
    </app:target>
  </app:ParameterizedTexture>
</app:surfaceDataMember>
```

上記の定義に加えてマテリアル定義が可能である。

```
<app:surfaceDataMember>
  <app:X3DMaterial>
    <app:diffuseColor>1 1 1</app:diffuseColor>
    <app:shininess>0.0625</app:shininess>
    <app:target>#fme-gen-ce7180a0-b597-4c43-b596-f5036405a010</app:target>

    <app:target>#fme-gen-b57636b0-a4ac-4f92-bff2-589de46a36ac</app:target>

  </app:X3DMaterial>
</app:surfaceDataMember>
```

3.1.3. データ変換

CityGML から 3D Tiles への変換は、FME（Safe Software 社製のデータ変換ソフトウェア）を使用している。FME は、プログラミングを必要としないノーコードで変換できるソフトウェアである。様々な入力、変換、出力機能を持つモジュールが提供されており、それらの処理をつなぎ合わせるだけでデータ変換処理ができる。この処理の流れを記述したファイルを、ワークスペースファイル（.fmw）という。

CityGML から 3D Tiles へ変換するワークスペースとして、「PLATEAU2 可視化用データ変換 01 建築物.fmw」が FME Hub Project PLATEAU（https://hub.safe.com/publishers/project-plateau）で公開されている。このワークスペースを実行すると、図 3-2 に示すウィンドウが開く。このウィンドウでは、変換パラメータを指定できる。しかし、調整可能なパラメータは、Tile Settings や Texture Setting など一部に限られている。調整可能なパラメータ一覧を表 3-3 に示す。

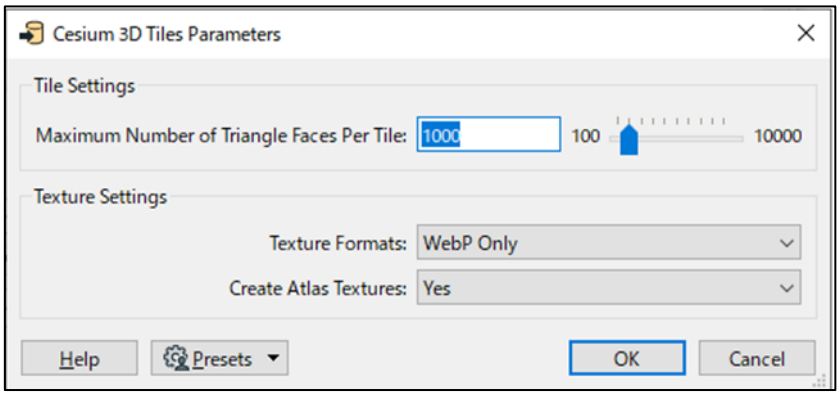


図 3-2　FME の 3D Tiles 変換パラメータ

表 3-3　FME の 3D Tiles 変換で調整可能なパラメータ

カテゴリ	項目	内容
Tile Settings	タイル当たりの最大面数	デフォルト：1000 設定可能範囲：1000～10000
Texture Settings	テクスチャフォーマット	WebP／JPEG・PNG／JPEG・PNG・WebP
	アトラス化	ON／OFF

3.1.4. 3D Tiles の構成

3D Tiles は図 3-3 に示すツリー構造で構成されており、モデル情報持つ物理的なファイル(b3dm)がツリーに沿った階層構造となっている。データにより階層数は異なるが、4、5 層程度となっている。

ルートに tileset.json ファイルが格納されていて、階層構造とその切替えに関わるパラメータを持つ。

描画時はカメラ距離が小さくなると小範囲のモデルに切り替える、カメラ距離が大きくなると広範囲なモデルに切り替える制御がされる。

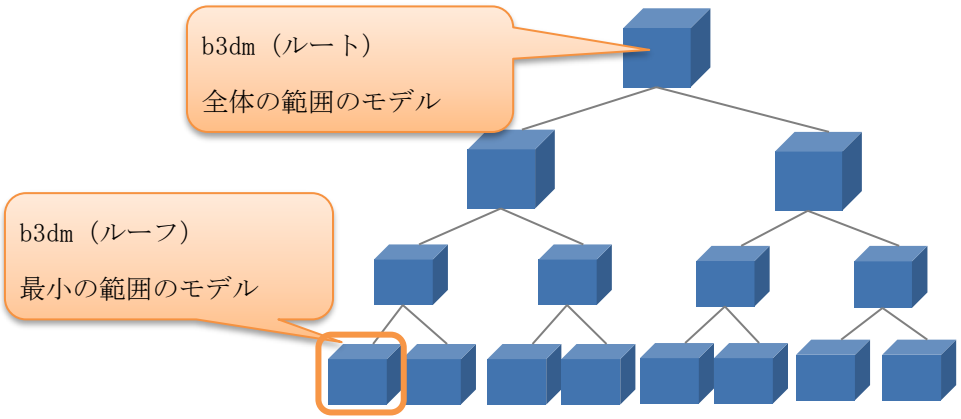


図 3-3　3D Tiles の構成

b3dm ファイルは図 3-4 に示す構造となっており、body 部分にテクスチャ情報を持つ。

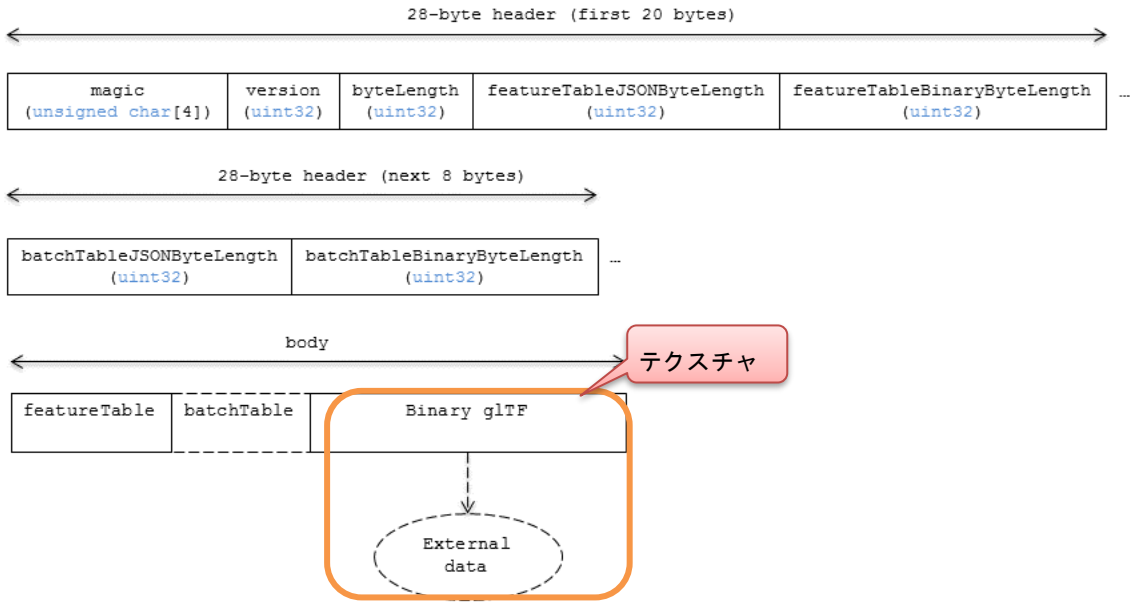


図 3-4　b3dm の構造

出典：GitHub - CesiumGS/3d-tiles-tools（ https://github.com/CesiumGS/3d-tiles-tools）

タイルのレンダリング制御（図 3-5）では、子タイルをレンダリングするかはピクセル単位で測定される空間誤差によって判断される。tileset.json に定義された[geometricgError]はジオ

メトリの簡略化された表現により生じる幾何学的誤差を定義したものであり、ルートほど大きい値を持ち、リーフタイルでは 0 に近くなる。幾何学的誤差を画面空間におけるカメラ距離、画面サイズ、解像度と共に使用して、SSE を算出する。SSE が最大許容値を超えると、タイルが絞り込まれ、レンダリングの対象となる。

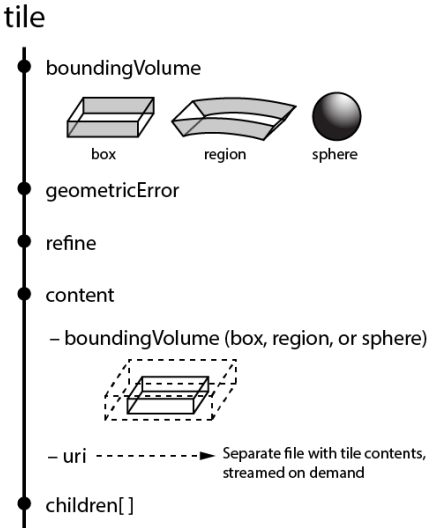


図 3-5　レンダリング制御

出典：3D Tiles Specification 1.0 (ogc.org)
https://docs.ogc.org/cs/18-053r2/18-053r2.html#figure_13

3.1.5. 建物テクスチャのまとめ方

前述の現行の建物テクスチャ、3D Tiles の構造等を踏まえ、建物テクスチャは近傍の建物で 1 つの画像にまとめることが望ましいと推測される。ただし、3 次メッシュでは画像ファイルサイズが大きくなり過ぎるため、適切なサイズでまとめる必要がある。以降でまとめ方（アトラス化）の検討を記載する。

3.2. アトラス化手法検討

令和 4 年度の取組において CityGML 2.0 へ標準仕様が拡大した際、各社アトラス化の取組を実施している。現状の建物テクスチャは、建物 1 棟に対して 1 つの画像ファイルを貼付する手法が主流となっている。

しかし CityGML 2.0 において、1 棟 1 画像ファイルとすると、画像ファイル数が多くなり、描画時の画像参照（読み込み）に時間を要することが課題である。

そこで、1 棟 1 画像ファイルの CityGML 2.0 から、複数棟 1 画像ファイルへ統合するアトラス化手法の検討及び再アトラス化ツールの開発を行った。

3.2.1. 再アトラス化のアルゴリズム

アトラス化を行うためには、CityGML 2.0 の読み書き及び Appearance に記述されているポリゴンの並び替えを行うアルゴリズムが必要である。

3.2.1.1. CityGML 2.0 の読み書き

CityGML 2.0 の読み書きの一部に、オープンソース（plateaupy）を使用している。plateaupy は、PLATEAU(CityGML)の Python 版パーサ及びビューア用モジュールである。

plateaupy：GitHub - AcculusSasao/plateaupy: PLATEAU parser and viewer in Python

(1) 読み出し

CityGML 2.0 からオープンソース（plateaupy）を使用して、建物ごとの情報を読み出したのち、表 3-4 の情報を保持する。

なおポリゴンの座標は UV 座標に変換した際に 0～1 となるように指定する。(負数は非対応とする。)

表 3-4 CityGML 2.0 から読み出す属性

タグ	内容
app:imageURI	画像 URI
app:mimeType	拡張子タイプ
app:target uri	ポリゴンの ID
app:target app:TexCoordList/app:textureCoordinates ring	ポリゴンの座標

(2) 書き出し

書き出し時は、読み込んだテクスチャ記述部分を削除した後、アトラス化後の情報に書き換えた画像 URI とポリゴンの座標を反映したテクスチャ情報と差し替えを行う。

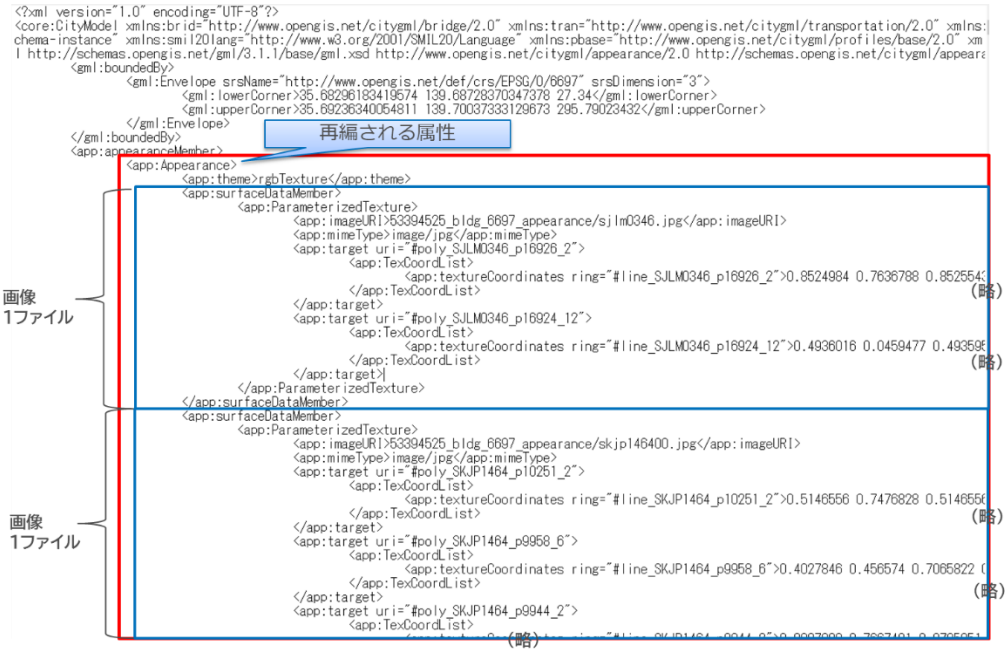


図 3-6　再アトラス化で再編される属性サンプル

入力画像と再アトラス化画像の例を図 3-7 に示す。

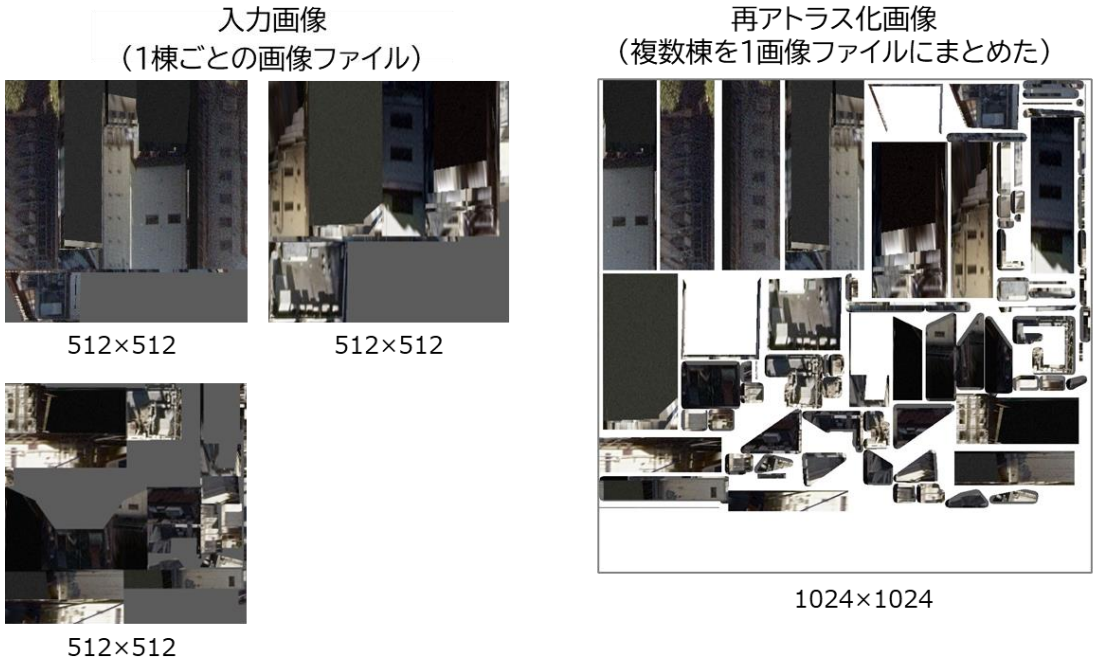


図 3-7　入力画像と再アトラス化画像

3.2.1.2. ポリゴンの並び替えアルゴリズム

ポリゴンを並び替えるアルゴリズムとして、長方形詰め込みアルゴリズムのうち Bottom-left-fit 法(以下「BLF 法」という)を使用した。本来左下から詰め込むアルゴリズムだが、座標軸が左上基準であるため左上から同方法を用いて並び替えを行う。

(1) 処理条件

BLF 法は、1 枚のキャンバス（画像）に画像を詰め込む問題である。そのため、読み込んだ画像の処理条件を 6 項目、独自に設定した。

✓ アトラス化は、CityGML ファイルごと処理を実施する。

✓ 入力画像が出力画像サイズより大きい場合、アトラス化の対象外とする。（図 3-8）。
（アトラス化は行わず、入力画像のまま保存する。）

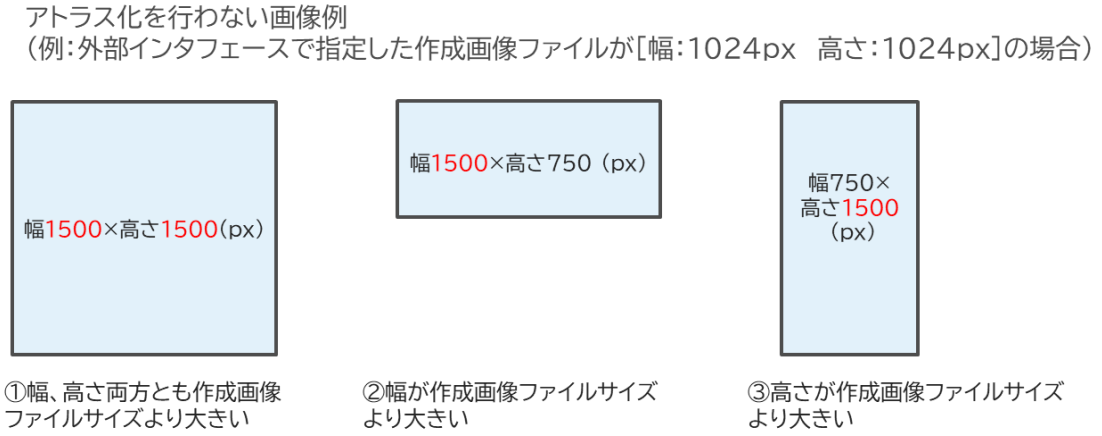


図 3-8 アトラス化を行わない画像例

✓ 1 枚の画像へ格納する建物画像は、原則として 4 次メッシュ相当とする（図 3-9）。
作成画像が基準となるパラメータ設定サイズを超える場合、余剰画像を新規画像ファイルに格納する。

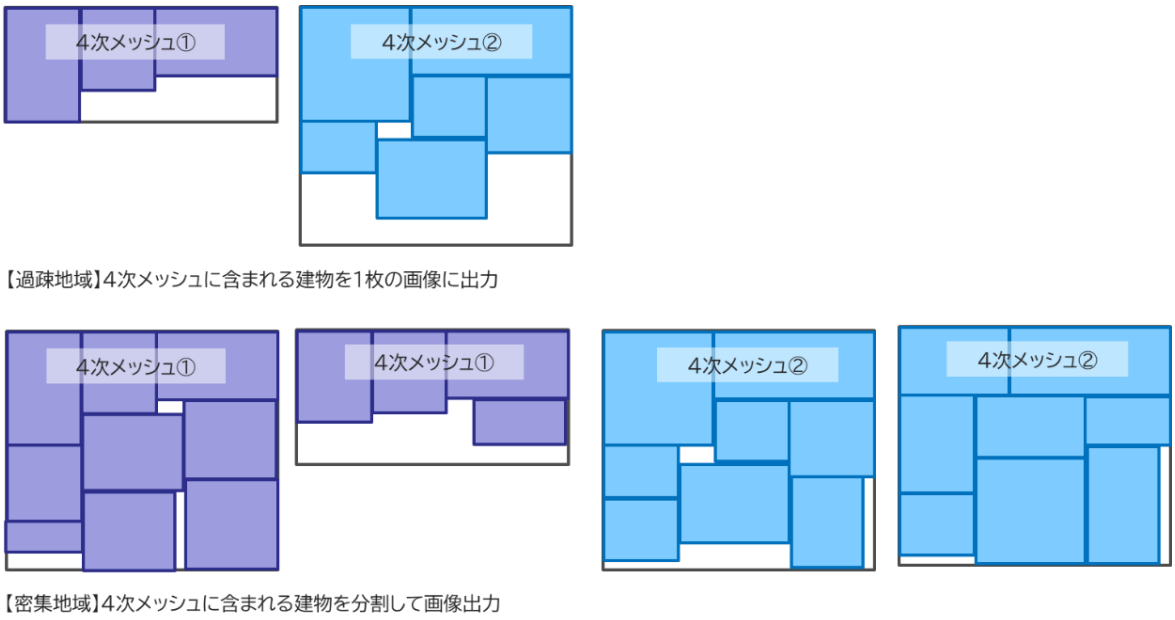


図 3-9 複数 4 次メッシュの画像格納イメージ

✓ 建物 1 棟分のテクスチャは同画像に格納し、複数画像にまたがらないようにする。

✧ 作成画像は高さと幅がそれぞれ 2 の累乗の長さになるように作成する。最大サイズをパラメータで[幅：2048px]［高さ:2048px]に設定している場合、[幅：2048px]［高さ:1024px]や[幅 ： 2048px]［高さ:512px]等の画像出力が想定される。

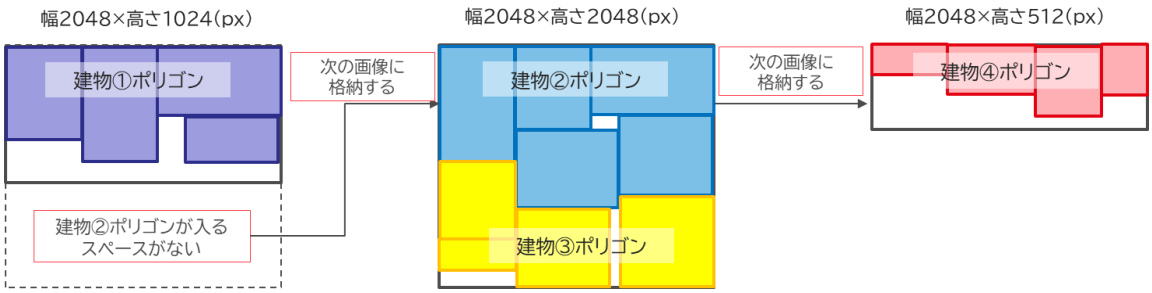
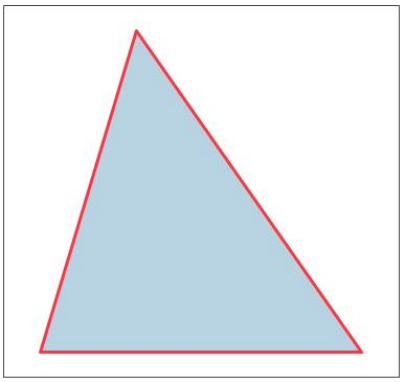
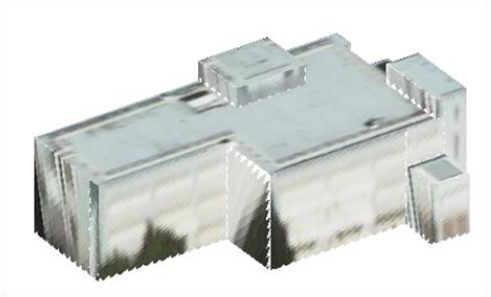


図 3-10　同 4 次メッシュ内での建物の画像格納イメージ

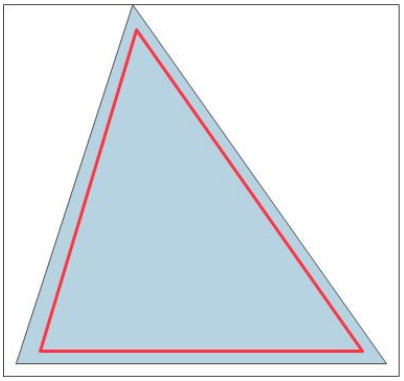
✓ 建物の屋根の切り出しは、入力 CityGML ファイルに記載のポリゴン座標を基準に行う。

✓ 入力 CityGML ファイルに記載のポリゴン座標を基準に切り出しを行うと、ポリゴンの境目が不自然になることがある。この場合、余白設定(外部インタフェースのポリゴン余白設定)を行うことにより大きめに元画像から切り出しを行うことができる（図 3-11）。なお、余 白設定を行っている場合、元画像によっては再アトラス後のポリゴン間に隙間ができることがある。





元画像のポリゴン座標どおりに切り出した場合、ポリゴンの境目が不自然になる場合がある



上記の場合、元画像のポリゴン座標より大きめに切り出すことで、ポリゴンの境目がスムーズになる場合がある

図 3-11　ポリゴン切り出し時の余白設定

(2) BLF 法

BLF 法のアルゴリズムフローを図 3-12 に示す。

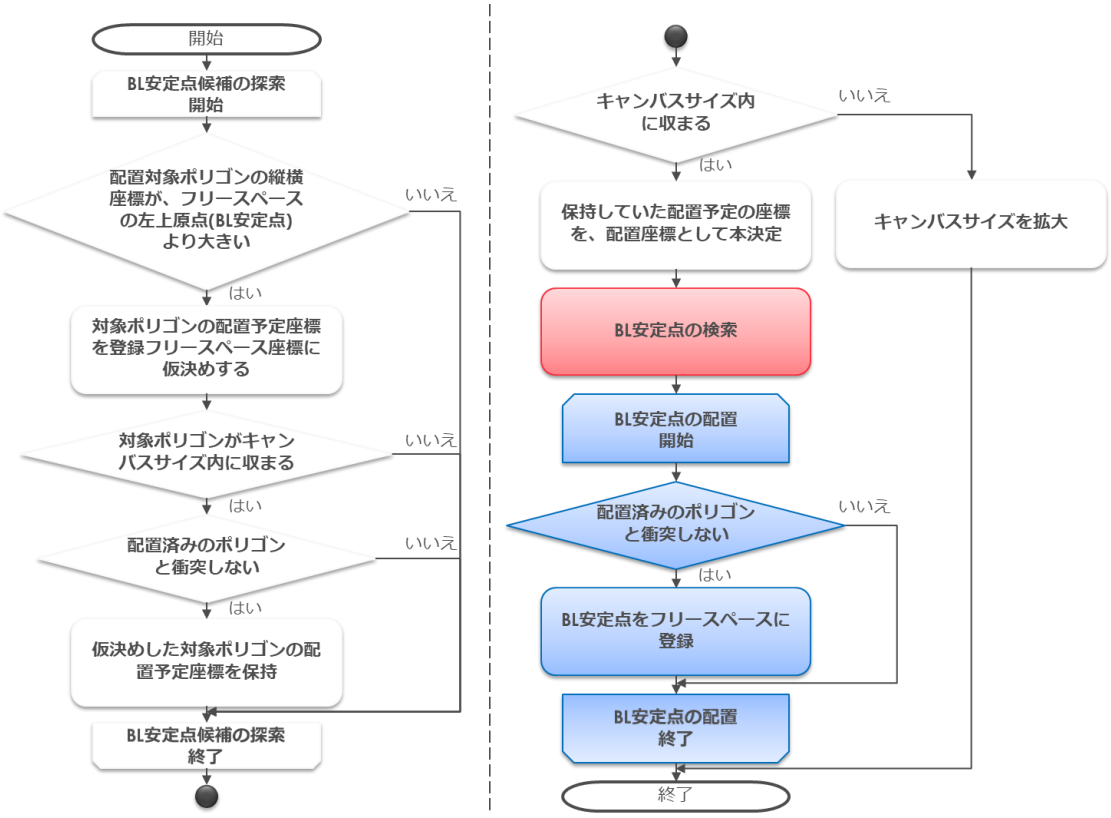


図 3-12　BLF 法のアルゴリズムフロー

①　BL 安定点候補の探索

図 3-12 において破線より左側で示す、「BL 安定点候補の探索」について説明する。 BL 安定点とは、複数のポリゴンを重ならないように配置できる位置の中で、左や上方向に動かすことができない点のことをいう。BL 安定点候補の探索では、キャンバス内に配置する予定の座標を取得する。

まず初めに、配置対象のポリゴンとキャンバスの枠（出力画像サイズ相当）間における BL 安定点の候補は、図 3-13 に示す赤色と黄色点となる。

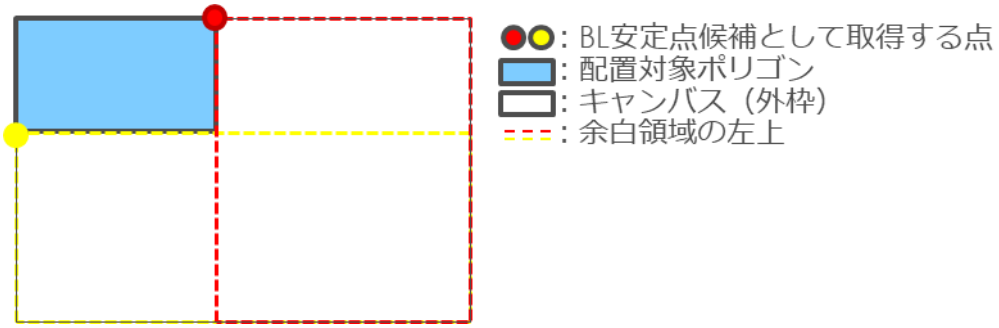


図 3-13 BL 安定点候補の取得

②　BL 安定点の検索

図 3-12 において赤色で示す、「BL 安定点の検索」について説明する。

BL 安定点の探索には 4 つのパターンがあり、ポリゴンをキャンバスに配置する際に、全ての座標を保持しておく。配置対象ポリゴンを基準として、配置済みポリゴンの位置がどこにあるのかにより保持する座標が異なる。4 つの探索パターンを以下に示す（図 3-14、図 3-15、図 3-16、図 3-17）。

1)　配置済みポリゴンが左側にある場合

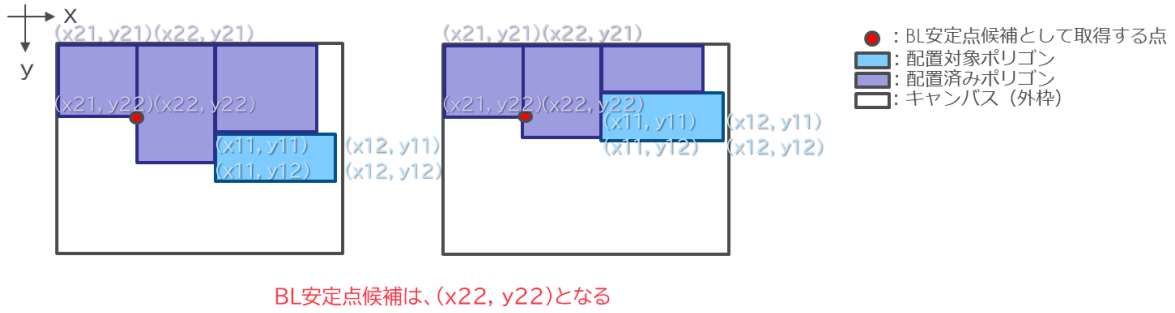


図 3-14　配置済みポリゴンが左側にある場合の BL 安定点候補

2)　配置済みポリゴンが右側にある場合

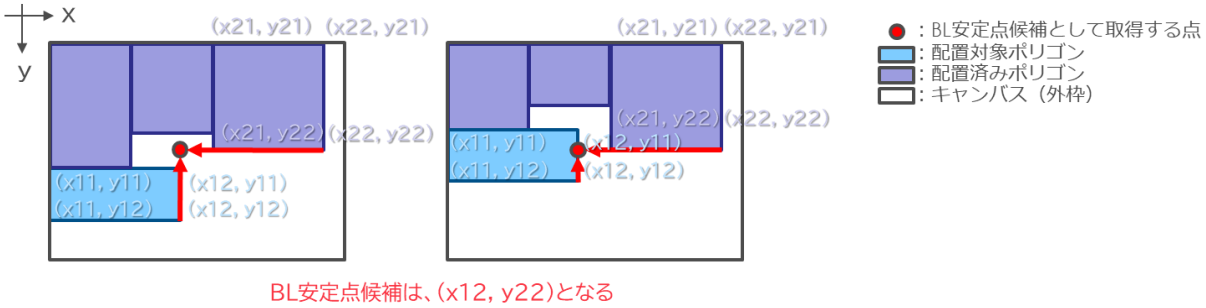


図 3-15　配置済みポリゴンが右側にある場合の BL 安定点候補

3)　配置済みポリゴンが上側にある場合

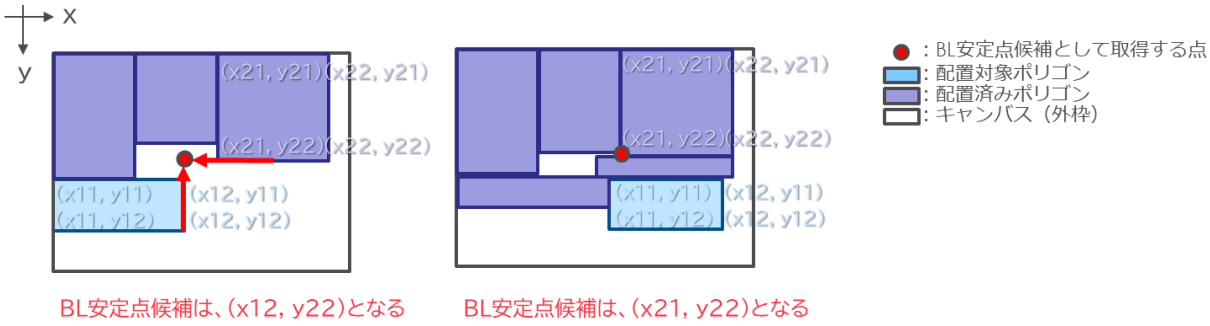


図 3-16　配置済みポリゴンが上側にある場合の BL 安定点候補

4)　配置済みポリゴンが下側にある場合

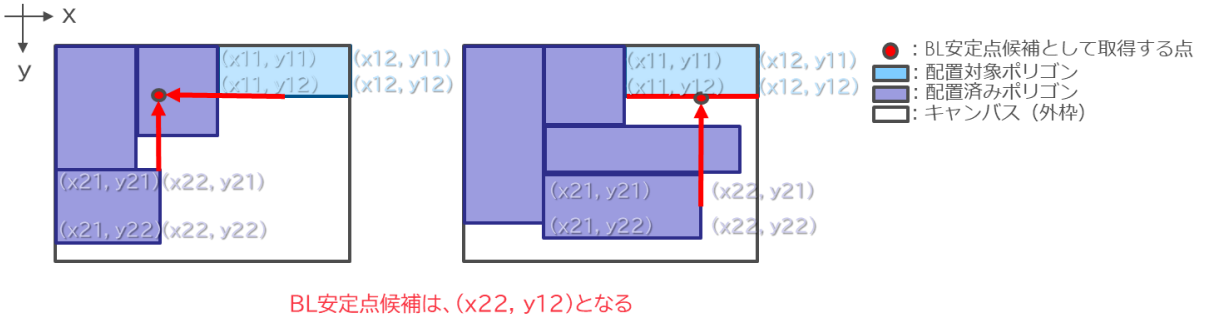


図 3-17　配置済みポリゴンが下側にある場合の BL 安定点候補

③　BL 安定点の配置

「BL 安定点の配置」は、図 3-12 の青色で示す処理である。BL 安定点を配置するために、検索した BL 安定点候補の中から、既に配置済みのポリゴンと衝突しないポリゴンのみを BL 安定点として登録する（図 3-18）。

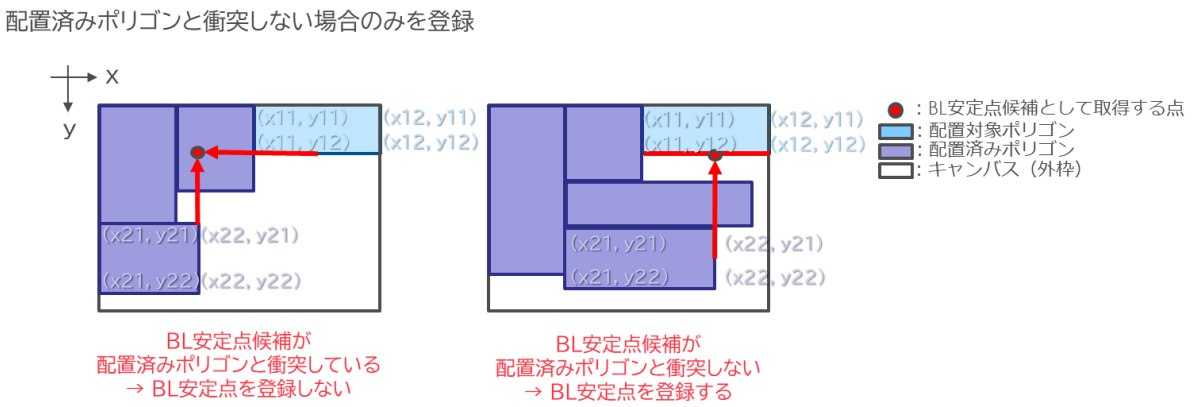


図 3-18　配置済みポリゴンと衝突しない場合のみ BL 安定点を登録

3.2.2. 再アトラス化ツールの開発

3.2.2.1. 機能概要

本処理は、令和 4 年度に整備された CityGML 2.0 とひも付く、テクスチャ画像ファイル（1 ファイルに対し建物 1 棟）のデータを対象とする。

再アトラス化ツールの処理フローを図 3-19 に示す。

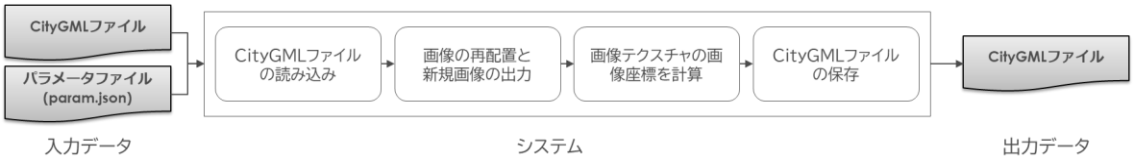


図 3-19　再アトラス化ツールの処理フロー

① CityGML 読込

建物 LOD2 の CityGML を読み込む。テクスチャ画像の形式は JPEG/PNG とする。建物内でポリゴンの高さ順にソートし、建物座標からまとめる範囲を計算する。

② 画像再配置

入力した画像を一定の範囲ごとに集約し、再配置した画像を作成する。再配置では画像の回転等は行わず、そのまま配置する。範囲は 5 次メッシュ相当を一定で分割した範囲の建物を 2048px 以下の画像にアトラス化する。大型の建物では 1 棟で規定の画像サイズを超える場合もあり、その場合はアトラス化対象から自動除外する。

③ 画像テクスチャ座標計算

アトラス化した画像における UV 座標を算出する。

④ CityGML 出力

画像座標（UV）と参照画像（URI）を更新する。

3.2.2.2. システムインタフェース

(1) 外部インタフェース

本ツールは、CityGML 2.0 に記載されている画像ファイル名とポリゴン名を、再編した後のデータに差し替える処理を行う。入力するデータ一覧を表 3-5 に示す。

表 3-5　再アトラス化ツールの入力データ一覧

入力データ	内容
CityGML 2.0	GML 形式及び Appearance の画像一式
パラメータファイル	再アトラス化の処理をするためのパラメータ情報ファイル（JSON 形式）

① CityGML 2.0

入力に使用する CityGML 2.0 のフォルダ構成を図 3-20 に示す。なお、再アトラス化ツールの出力フォルダ構成は、入力する CityGML 2.0 のファイル構成と同じであり、CityGML ファイル及び Appearance の画像一式のフォルダを出力する。

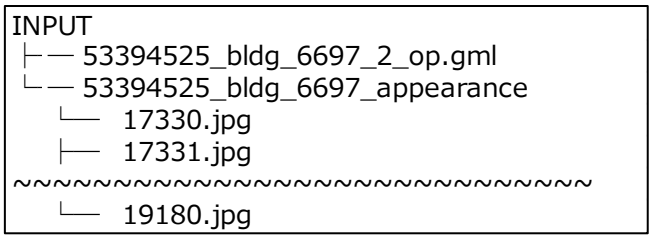


図 3-20　再アトラス化ツールの入力サンプル（CityGML 2.0 フォルダ構成）

再アトラス化後の CityGML ファイルの例を図 3-21 に示す。

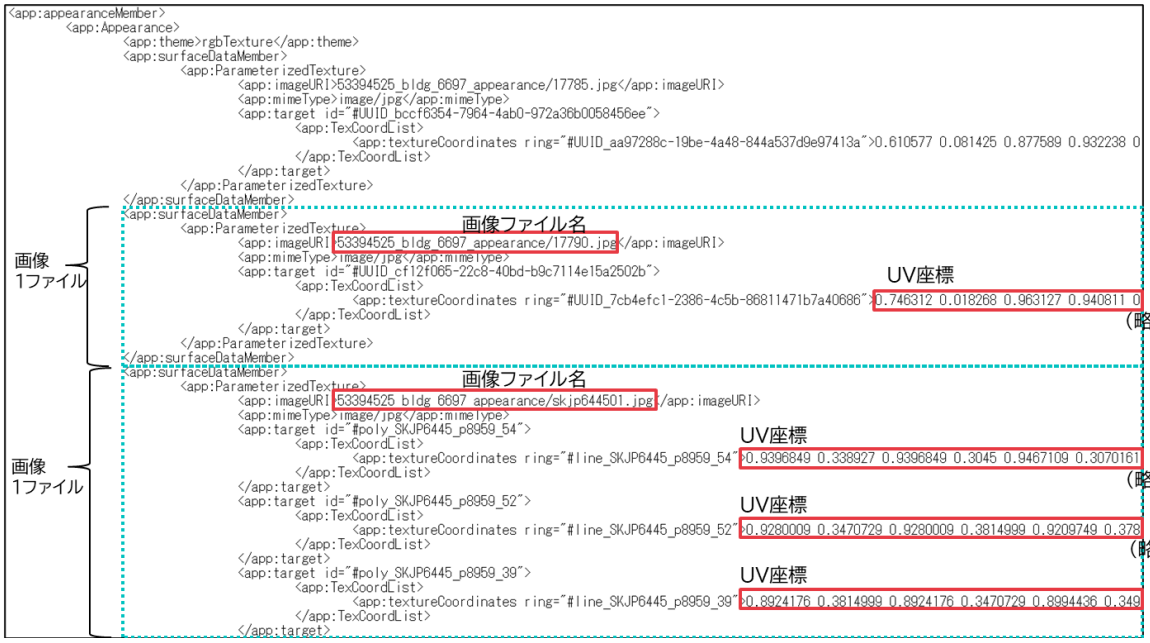


図 3-21　再アトラス化後の CityGML ファイル例

②　パラメータファイル（param.json）

パラメータを記載した param. json ファイルは、使用環境に合わせて編集することができる。パラメータファイルには、入出力フォルダの相対パス及び内部で使用する各種パラメータを記載する必要がある（図 3-22）。

```
{
  "FilePath": {
    "InputGMLFolderPath": "./input",
    "OutputGMLFolderPath": "./output"
  },
  "OutputWidth": 2048,
  "OutputHeight": 2048,
  "BackGroundColor": 255,
  "Extentpixel": 1
}
```

図 3-22　再アトラス化ツールの可変内部パラメータファイルサンプル

- 入力フォルダパス(InputGMLFolderPath)

図 3-20 の「INPUT」を示す。「INPUT」フォルダには、CityGML ファイルを格納してあるものとする。

- 出力フォルダパス（OutputGMLFolderPath）

再アトラス化した CityGML ファイルを保存するフォルダである。パスを記載すると自動的に作成される。実行する度にフォルダを作成するため、2 度実行する場合は初めに保存したファイルは削除される。

- 作成画像ファイルの幅（OutputWidth）

出力する画像ファイルの幅（単位：ピクセル）設定である。
2 の累乗の数値を指定する。

- 作成画像ファイルの高さ（OutputHeight）

出力する画像ファイルの高さ（単位：ピクセル）設定である。
2 の累乗の数値を指定する。

- 作成画像の背景色（BackGroundColor）

出力する画像の背景色設定である。
0(黒)～255(白) の数値を指定する。

- ポリゴン余白（Extentpixel）

元画像から画像を切り出す際、外側を余分に切り取る場合の幅（単位：ピクセル）設定である。
描画時にポリゴンの境目がスムーズになるように、元画像を大きめに切り出す際に使用する。不要な場合は 0 を指定する。

(2) 内部インタフェース

① コンフィグレーションファイル（config.py）

ツールの内部ファイルとして、コンフィグレーションファイル（通称：コンフィグファイル、ファイル名：config.py）がある。コンフィグファイルは、固定の内部パラメータを記載する。コンフィグファイルには、図 3-22 に示す可変内部パラメータ（param.json）とは異なり、ツール利用時に変更することを想定しない変数を記述している。コンフィグファイルに記載のパラメータを表 3-6 に示す。

表 3-6　再アトラス化ツールの固定内部パラメータ

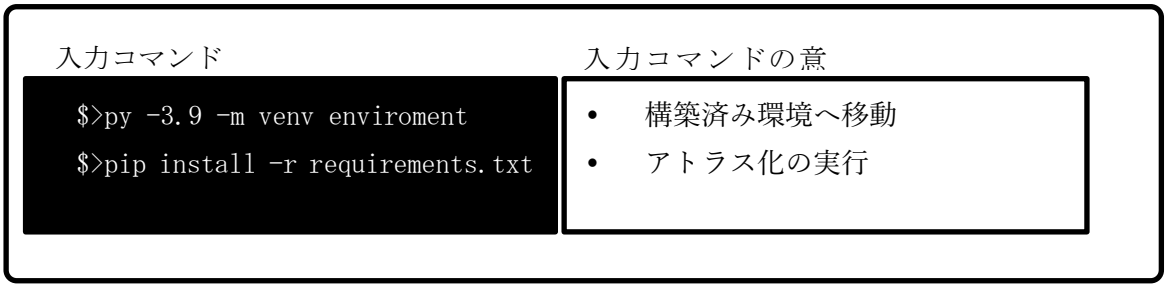
変数名	値	意味
SYSYTEM_VERSION	"1.0.0"	システムのバージョン管理番号

(3) ユーザインタフェース

ユーザーはコマンド ライン実行で、再アトラス化を行う。コマンド ライン実行に使用するアプリは、Windows PowerShell 又はコマンド プロンプト等のターミナルの使用を想定する（図 3-23）。

図 3-23　再アトラス化ツールのユーザインタフェース

ユーザーは、「3.2.2.3 動作環境」で述べた構築環境を使用する。構築した環境へアクセスして、コマンド ラインで実行する。処理が完了すると、可変内部パラメータファイル(.json)に記載した出力フォルダへ、再アトラス化された CityGML 2.0 が保存される。



(4) 使用するライブラリ

Python をはじめとした複数のライブラリを使用する。各ライブラリ名とバージョンを表 3-7 に示す。

表 3-7 再アトラス化ツールの使用ライブラリ

名称	バージョン
Python	3.9
lxml	4.9.2
mypy	1.4.0
mypy-extensions	1.0.0
numpy	1.25.0
opencv-contrib-python	4.7.0.72
tomli	2.0.1
typing_extensions	4.6.3
tqdm	4.65.0

3.2.2.3. 動作環境

本ツールは、Windows 環境での使用を想定する。そのため、Windows で動作する環境構築を記載する。

(1) ベース環境の構築

再アトラス化ツールは、Python 言語で記述している。よって、ツールの環境構築前にベースとなる環境を構築する。手順を以下に示す。

① 公式サイトへアクセス

以下の公式サイトをブラウザで開く。

https://www.python.org/downloads/windows/

② インストーラーのダウンロード

「Python 3.9.x」という表示の下にある「Windows installer (64-bit)」をクリックすると、Python インストーラーのダウンロードが始まる。本ツールでは、Python 3.9.13 を使用する（図 3-24）。

図 3-24　再アトラス化ツールのベース環境

③ インストーラーの実行

ダウンロードが完了後、インストーラーをダブルクリックで開く。「Add Python 3.9 to PATH」にチェックを入れ、「Install Now」をクリックする（図 3-25）。

図 3-25　再アトラス化ツールの環境インストール（環境パスの設定）

④ 管理者権限の付与

インストールには管理者権限が必要なため、ユーザアカウント制御確認ダイアログが表示される場合がある。表示された場合は、「はい」をクリックする。

⑤ インストールの完了

図 3-26 の画面が表示されると、環境構築は完了である。

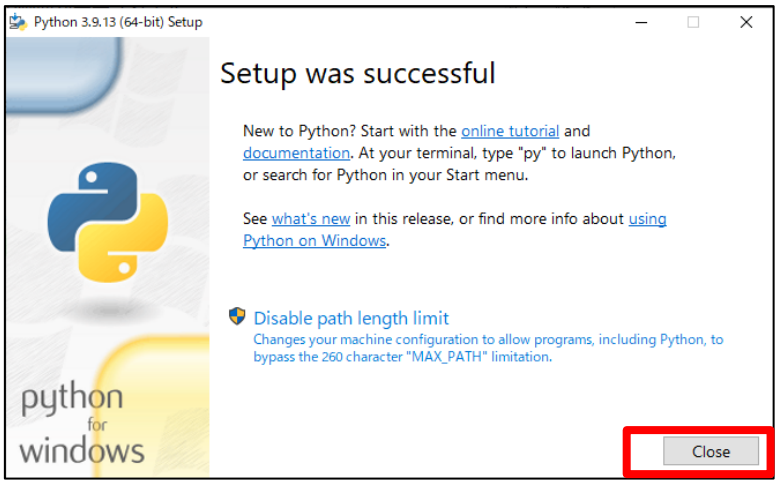


図 3-26　再アトラス化ツールの環境構築完了画面

次に、再アトラス化ツールに必要なライブラリをインストールする。

(2) Windows PowerShell の起動

Windows の検索ボックスで「Windows PowerShell」と入力を行い、ターミナルを開く。ここでは、Windows PowerShell を使った方法を記載するが、コマンド プロンプト等でも同様にして実行可能である。

① 実行環境場所の作成

ターミナル上で、実行環境のフォルダを作成する。

入力コマンド	入力コマンドの意味
$>cd C:¥Users¥User_name¥Desktop $>mkdir my_atlas_project $>cd my_atlas_project	• デスクトップへ移動 • 「my_atlas_project」フォルダを作成 • 「my_atlas_project」フォルダへ移動

② ライブラリー括インストールファイルのコピー

「my_atlas_project」フォルダへ「requirements.txt」をコピーする。

③ ライブラリのインストール

Python 3.9 の仮想環境を作成して、システムに必要なライブラリをインストールする。

入力コマンド	入力コマンドの意味
$>py -3.9 -m venv enviroment $>pip install -r requirements.txt	• Python 3.9 の環境を作成 • アトラスに必要なライブラリをまとめてインストール

④ 構築完了

「my_atlas_project」フォルダへ、アトラス化のソースコード一式と入力データをコピーして、実行環境の構築は完了である。

3.3. 建物モデル LOD2 の生成実証

技術調査結果に基づき、一定規模の CityGML 2.0 データセットを用いてデータ作成実証を行った。アトラス化検討で開発した再アトラス化ツールを使用して、建物モデル LOD2 の作成実証を行った。

3.3.1. 生成実証

実証環境は、表 3-8 に示すデータ及び PC を使用した。

表 3-8 実証データと環境

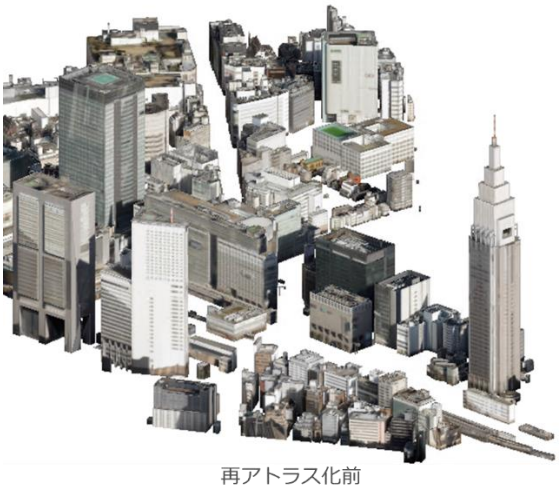
項目	名称	内容
検証データ	渋谷区データ	CityGML4 ファイル、画像容量：500MB
PC スペック	プロセッサ	Intel(R)Xeon(R)W-2123 CPU @ 3.60GHz 3.60GHz
	実装 RAM	16.0GB
	OS	Windows 10 Pro (21H2)

再アトラス化ツールの処理は、本実証環境において約 25 分掛かる。

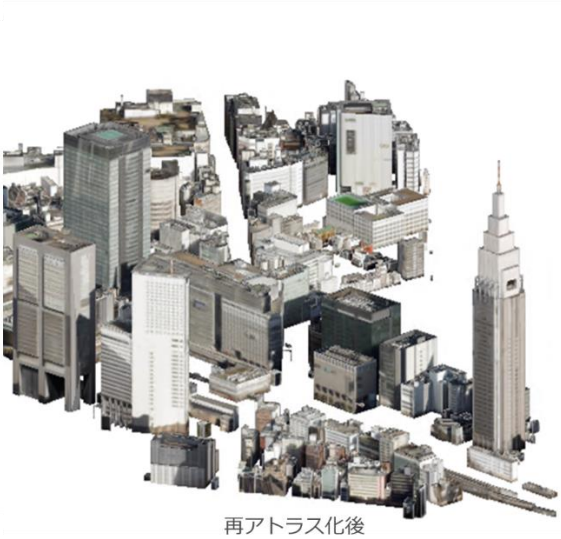
3.3.2. 描画検証

処理結果は、FZKViewer で表示し、目視で確認した。

再アトラス化前後で見え方の違いを図 3-27 に示す。表示した結果、見え方に違いがないことを確認した。



再アトラス化前



再アトラス化後

図 3-27　再アトラス化の前後の描画結果

3.3.3. ファイル構成の確認

処理前後における画像ファイル数と総容量の比較結果を表 3-9 に示す。画像ファイル数は約58%削減した。一方で、総容量は約 100%（2 倍）へと増加していることが分かる。これは再アトラス化した際、1 枚の画像に異なる大きさや形状の画像を詰め込むことで、余白が生じてしまうためである（図 3-28）。

表 3-9　再アトラス化前後のファイル数と容量

メッシュ ID	画像ファイル数[MB]		総容量[MB]	
	再アトラス化前	再アトラス化後	再アトラス化前	再アトラス化後
53394525	1715	651	259	274
53394526	463	240	56	73
53394535	1031	440	100	140
53394536	1055	480	86	480
合計	**4264**	**1811**	**501**	**967**

約 58%削減　　約 100%増加

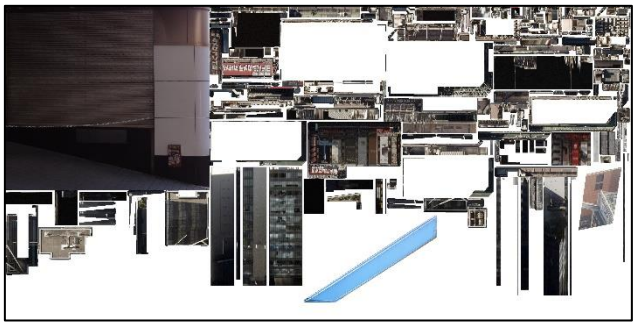


図 3-28　再アトラス化後の画像例（余白）

3.4. 建物モデル LOD2 の描画性能検証

クライアントアプリ及び Web GIS アプリでの描画性能を検証する。検証として入力データに、令和 4 年度業務で作成された複数都市の建物 LOD2（3D Tiles）を用いる。

テクスチャ付き建物 LOD2 の 3D Tiles を PLATEAU VIEW で表示する際、テクスチャのサイズ、解像度、ファイル数等により、表示速度や見え方に差異が生じる。CityGML 2.0 からテクスチャ付き建物 LOD2 の 3D Tiles へ変換し、建物テクスチャの違いによる描画性能を検証した。

3.4.1. クライアントアプリ性能検証

クライアントアプリは CityGML 形式が直接インポート可能な FZKViewer における描画性能を確認した。再アトラス化を実施した場合では、表示されるまでの時間が短縮することが確認できた。

3.4.2. Web GIS アプリでの性能検証

性能検証は汎用的な Web ブラウザで 3D Tiles を表示した際の描画性能を評価した。検証に用いた 3D Tiles の作成条件及び性能検証の確認観点、検証結果を記載する。

3.4.2.1. 検証環境

実証環境は、表 3-10 に示す PC を使用した。

表 3-10　Cesium 性能検証における実証環境

項目	名称	内容
PC スペック	プロセッサ	Intel(R)Xeon(R)W-2123 CPU@3.60GHz
	実装 RAM	32.0GB
	OS	Windows 10 Pro (21H2)

3.4.2.2. 3D Tiles 変換

既往の PLATEAU 用に公開されている 3D Tiles 変換のワークスペース「PLATEAU2 可視化用データ変換 01 建築物.fmw」（図 3-29）を使用して CityGML 2.0 から 3D Tiles へ変換する。ワークスペースオブジェクトの 3DTilesWriter 処理のみ、本検証用にパラメータを変更できるよう変更して使用した。

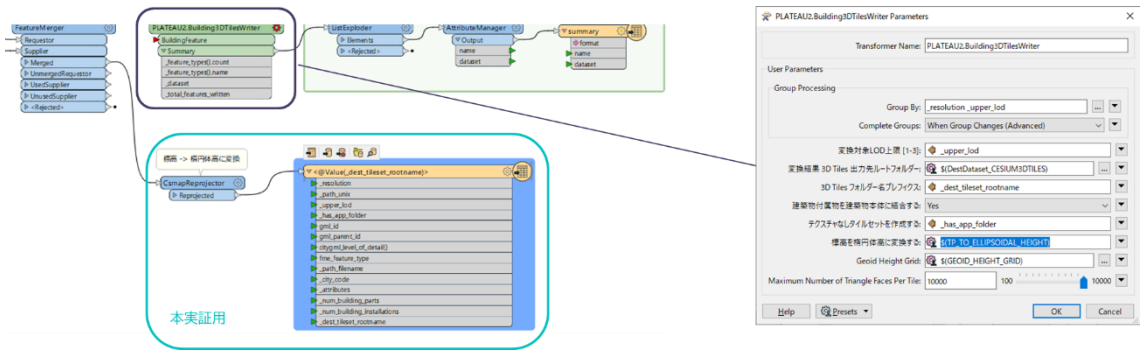


図 3-29　3D Tiles 変換のワークスペース

出典：PLATEAU2 可視化用データ変換 00 総合 | FME Hub (safe.com)

CityGML から 3D Tiles へ変換する際に、性能検証のために以下の 3 点のパラメータを変更してデータを作成した。

1. 最大タイル数(10000/30000/50000)
2. 再アトラス化ツールの使用有無(有/無)
3. FME によるアトラス化の実施有無(有/無)

また、3D Tiles データに含まれる建物の疎密による影響を検証するために、渋谷、加賀の 2 都市でデータ作成を実施した。

3.4.2.3. 性能検証における確認点

確認点は下記の 4 点とした。

1. 描画中のフレームレート
2. メモリ使用量
3. リクエスト数
4. 初期読込時間

3.4.2.4. 検証結果

Cesium での性能評価結果を表 3-11 に記載する。表中の「ー」はメモリ上限を超過したため、Web ブラウザ上で 3D Tiles の描画に失敗した箇所を指す。加賀市のような範囲当たりのテクスチャ量が多くない場合には、再アトラス化による効果がみられるが、画像をまとめることで、メモリ使用量は増加するため、渋谷のような範囲当たりのテクスチャの量が多い場合には、画像のまとめ方での描画性能の向上では限度がみられた。表内にパラメータとして設定した条件、確認点を記載する。

表 3-11　Cesium での性能検証結果

作成条件パターン	最大タイル数	再アトラス化ツール使用有無	FMEによるアトラス化処理	作成都市	フレームレート(FPS)		メモリ使用量(MB)	リクエスト数	初期読込時間(秒)
					最小	描画後			
1	10,000	有	無	渋谷	－	－	－	－	－
2	30,000	有	無	渋谷	－	－	－	－	－
3	50,000	有	無	渋谷	3	60	1,320.3	9,480	11.61
4	10,000	有	有	渋谷	1	60	1,227.4	1,131	15.27
5	30,000	有	有	渋谷	1	60	1,058.4	997	13.24
6	50,000	有	有	渋谷	1	60	1,121.1	1,020	13.85
7	10,000	無	無	渋谷	－	－	－	－	－
8	30,000	無	無	渋谷	－	－	－	－	－
9	50,000	無	無	渋谷	－	－	－	－	－
10	10,000	無	有	渋谷	3	60	1,726.5	1,020	13.85
11	30,000	無	有	渋谷	3	60	1,079.2	1,148	12.08
12	50,000	無	有	渋谷	3	60	1,147.0	1,203	8.64
13	10,000	有	無	加賀	15	20	347.8	863	5.17
14	30,000	有	無	加賀	16	20	559.4	852	5.32
15	50,000	有	無	加賀	16	16	559.4	864	5.23
16	10,000	有	有	加賀	16	16	189.9	862	5.24
17	30,000	有	有	加賀	16	16	580.3	860	5.28
18	50,000	有	有	加賀	16	16	580.3	854	5.54
19	10,000	無	無	加賀	16	60	85.3	862	5.23
20	30,000	無	無	加賀	44	60	102.8	852	5.23
21	50,000	無	無	加賀	48	60	102.8	854	5.21
22	10,000	無	有	加賀	38	60	88.7	862	5.21
23	30,000	無	有	加賀	37	60	114.1	853	5.23
24	50,000	無	有	加賀	42	60	114.1	854	5.23

3.5. 調査検討のまとめ

再アトラス化により、初期表示速度に改善がみられることが確認できたことから、データ整備の手順において再アトラス化を行うことは描画性能を確保する上で、有効であることが確認できた。しかし、画像のまとめ方で描画性能を改善することには限度があり、タイル化の手法、解像度や閲覧環境と合わせて検討していく必要がある。また、今回開発したツールは、現行のデータ整備の手順に適用できる点、既往の整備済みのデータにも適用できる点が利用の促進に繋がり、描画性能向上の一助となると考える。

4. OSS 化

開発した建物テクスチャ視認性向上ツール、再アトラス化ツールは、OSS として PLATEAU GitHub に公開した。表 4-1 に公開概要を示す。GPL v3.0 ライセンスで公開しているため、誰でも使用・改良可能である。

表 4-1　PLATEAU GitHub における公開概要

屋根面視認性向上ツール	URL	https://github.com/Project-PLATEAU/Auto-Create-bldg-lod2-tool
	公開内容	建物（屋根面）テクスチャ視認性向上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
壁面視認性向上ツール	URL	https://github.com/Project-PLATEAU/Auto-Create-bldg-lod2-tool
	公開内容	建物（壁面）テクスチャ視認性向上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
再アトラス化ツール	URL	https://github.com/Project-PLATEAU/Auto-Create-bldg-lod2-tool
	公開内容	再アトラス化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. 用語集

■ 用語の一覧

用語	説明
GAN(Generative Adversarial Networks)	敵対的生成ネットワーク。生成器と識別器とが敵対的な関係で学習し、新しい画像を生成する手法。
CNN(Convolutional Neural Network)	畳み込みニューラルネットワーク。主に畳み込み層、プーリング層、全結合層から構成されるニューラルネットワークの一種。画像認識の分野で広く使われる。
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。
XML(Extensibule Markup Language)	基本的な構文規則を共通とすることで、任意の用途向けの言語に拡張することを容易としたマークアップ言語の総称。
アトラス化	PLATEAU が採用する都市スケールのデータセット単位 (CityGML2.0) における Appearance クラスの統合処理のことである。Appearance クラスには、テクスチャ画像ファイル名や座標情報等、テクスチャに関する属性が記述されている。

6. 参考資料

■　参考資料の一覧

参考資料	バージョン	参照目的
実証環境構築マニュアル	ver.3.0	PLATEAU の実証環境の調査
3D 都市モデル標準製品仕様書	ver.3.5	建物テクスチャの仕様の確認
3D 都市モデル標準作業手順書	Ver.3.5	建物テクスチャの仕様の確認

3D 都市モデルのテクスチャ高解像度化手法及び描画パフォーマンス向上に関する
技術調査レポート

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