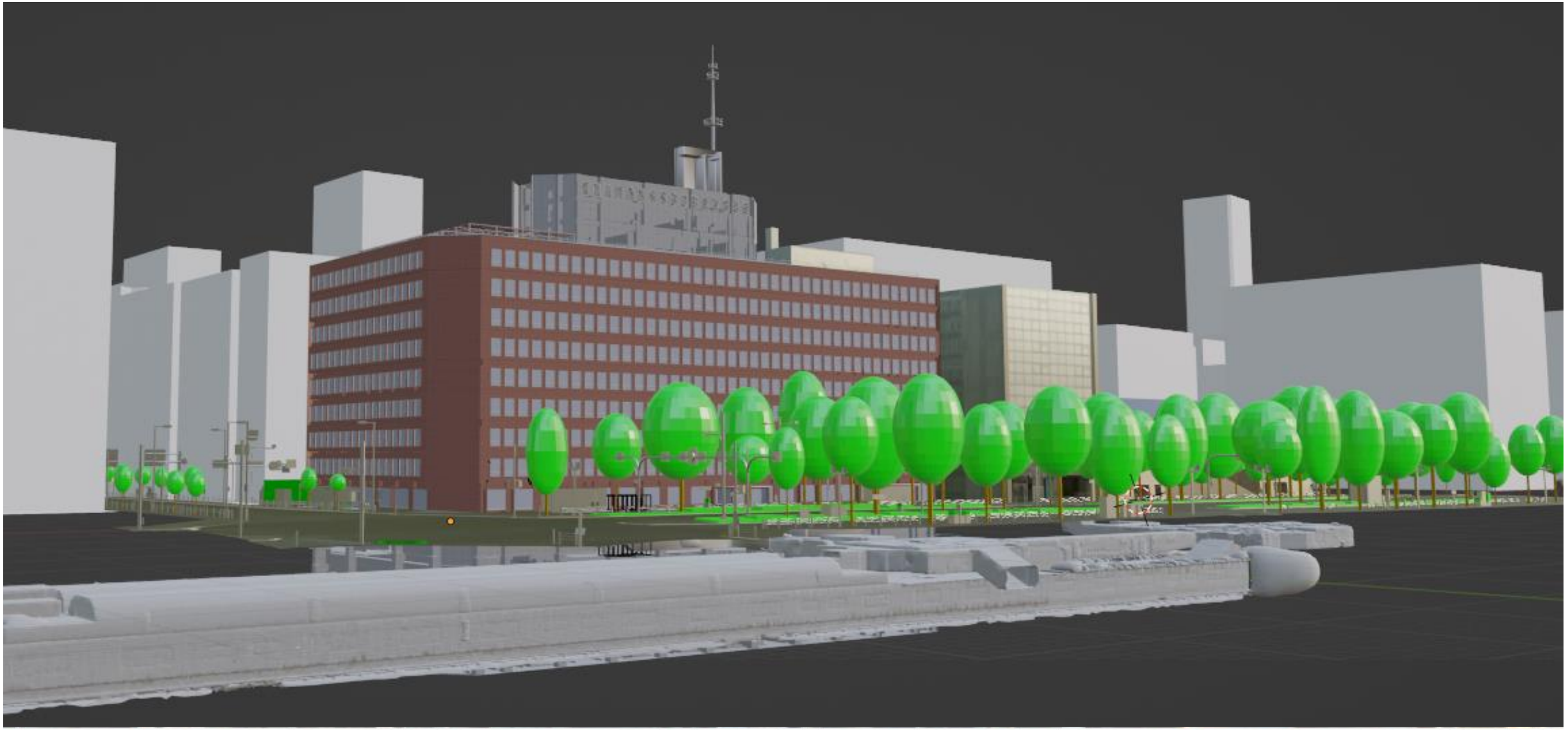
都市空間の統合デジタルツインの構築　技術検証レポート

Technical Report for Digital twin integrated multiple urban spatial data

PLATEAU by MLIT



目次

Ⅰ. 実証概要

Ⅱ. 実証技術の概要

Ⅲ. 実証システム

Ⅳ. 実証技術の検証

Ⅴ. 成果と課題

全体概要（1/2）

ユースケース名	都市空間の統合デジタルツインの構築
実施場所	大阪府大阪市天満周辺 / 大阪府大阪市本町駅周辺
目標・課題 ・創出価値	• デジタルツインを社会実装していくためには、都市、建物、設備等の様々なオブジェクトを再現する点群、CG、BIMモデル等の複数の3Dデータを統合し、相互運用性を確保することが必要になる。 - 3D都市モデル、点群データ、BIMモデルと複数の3Dモデルを統合する上で標準的な対応方法が定まっておらず、各事業者・各業界で統一されたデジタルツインの構築ができていない。 - デジタルツインを活用した事業を起こす際に、仕様の策定～開発まで個別対応が必要となることで開発コストが大きくなり、事業が発展しづらい環境となっている。 • 3D都市モデルをベースに、BIMモデルと点群データを統合したデジタルツインを構築し、パーソナルモビリティ運用とARナビによって相互運用性が確保できているかを確認することで、デジタルツインの構築手法を確立する。
ユースケース の概要	• 3D都市モデルをベースとして異なる3Dデータを統合する手法を開発する。 - あらゆるヒト・ロボティクス・建物・都市をつなぐデジタルツインの構築を実現する。 - データ統合の方法について整理・比較検討し、標準的なモデル統合手法を開発することでこれらをデジタルツイン構築のための3Dデータ統合ガイドラインとして取りまとめる。 • 統合したデータをパーソナルモビリティ運用システムとARナビにより精度検証を行い、各手法の有用性を検証する。

全体概要（2/2）

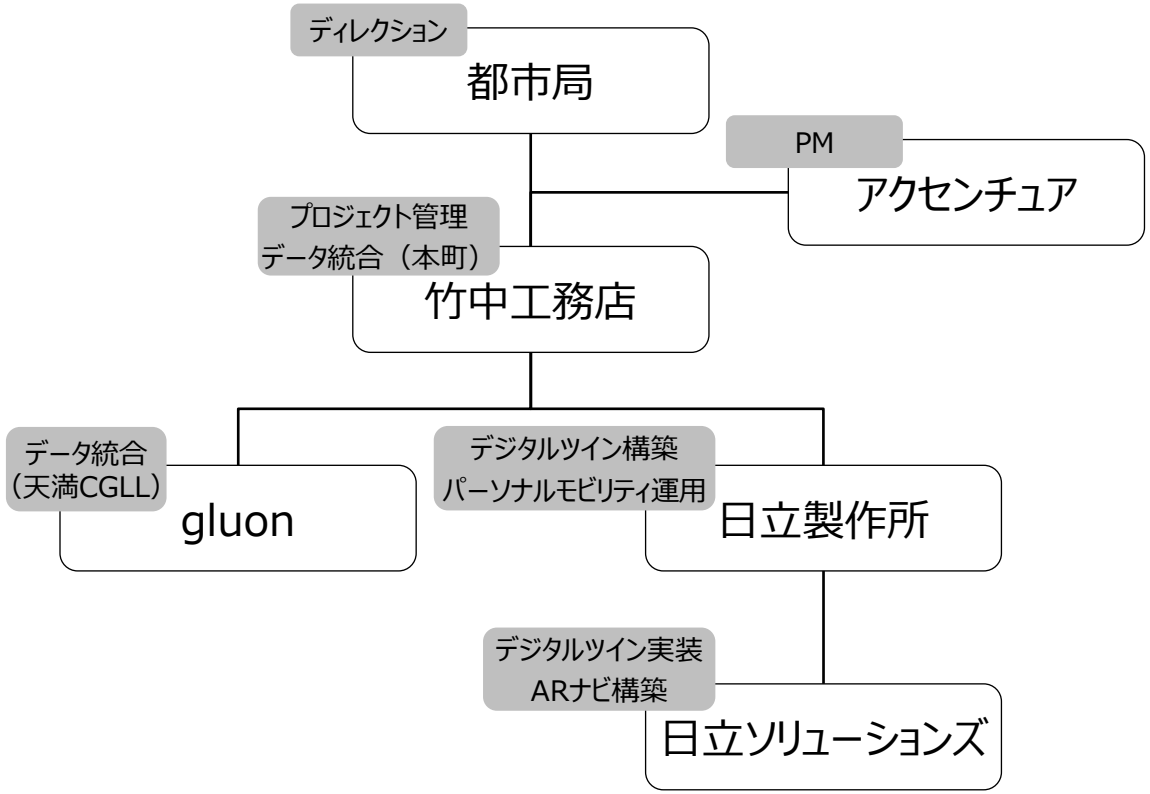
実証仮説	• 3D都市モデルやBIMモデル、点群データなど、データ形式の異なるモデルをいくつかの手法で統合し、その必要工数と実際のアプリケーションとしてパーソナルモビリティ運用とARナビを通じて統合データの精度を明らかにすることで、体系化したモデルの統合手法を構築する。 • 統合したデータを使ったデジタルツイン空間を構築しパーソナルモビリティ運用やARナビで活用することで様々な空間データのリアルタイム連携を実現する。
検証ポイント	• 都市、建物、点群と複数の3Dモデルの特徴の整理、課題点の抽出 • デジタルツイン構築する際の各モデル統合方法について整理・比較検討 • ARナビ表示（案内表示やルート表示）のズレの有無 • パーソナルモビリティ運用での自動走行の可否

実施体制

各主体の役割

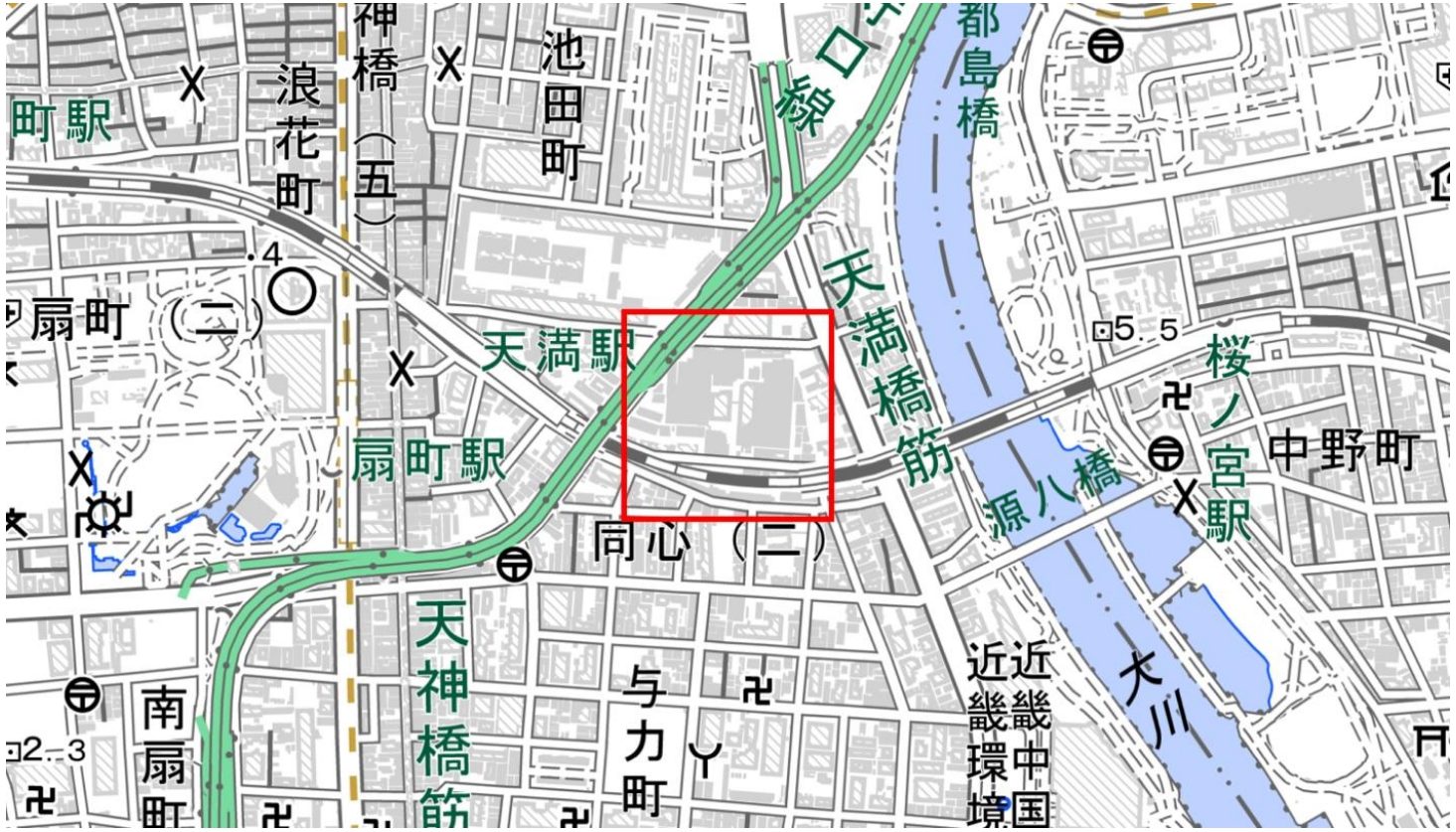
主体	役割
竹中工務店	• プロジェクト管理、とりまとめ • データ統合（本町）
gluon	• データ統合（天満CGLL）
日立製作所	• デジタツイン構築 • パーソナルモビリティ運用
日立ソリューションズ	• デジタルツイン実装 • ARナビ構築
アクセンチュア	• プロジェクトマネジメント

実施体制図



実証エリア｜大阪府大阪市天満周辺

大阪府大阪市天満周辺　面積：約47,000㎡

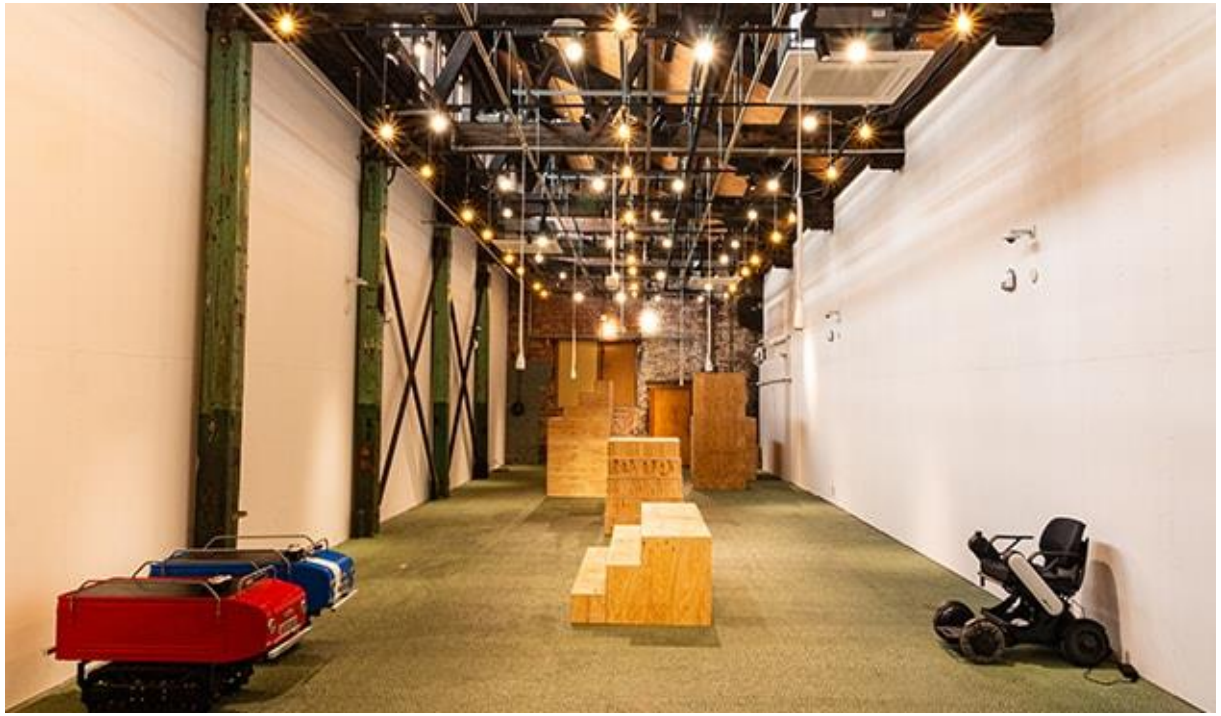


実証エリア｜大阪府大阪市天満周辺　CGLLについて

CGLL（コモングラウンド・リビングラボ）の詳細

項目	詳細
名称	コモングラウンド・リビングラボ（CGLL）
場所	大阪市北区天満橋3-3-5　中西金属工業（株）本社内
概要	• 異業種が集まり、コモングラウンドを試して作れる実験場 - 参画者がデータ／実験結果を互いに提供し、共有実証を進め、技術・運営ノウハウを先行して集積している - 複数の企業や団体が垣根なく議論、実験し、次世代都市の空間情報プラットフォーム実装を探る
コモングラウンドとは	• 「Society5.0」実現に向けた汎用的なインフラとなりえる空間プラットフォーム - 建築や都市の3Dデータをインデックスとして、空間に存在する様々なものをデジタル情報として記述することで、フィジカル空間とサイバー空間をリアルタイムにシームレスにつなぎ、人とロボットが共通認識を持ち得る社会の実現を目指している

CGLL（コモングラウンド・リビングラボ）の様子



*出典：https://www.cgll.osaka/

実証エリア｜大阪府大阪市本町駅周辺

大阪府大阪市本町駅周辺　面積：約17,800㎡

スケジュール

実施事項	令和4年									令和5年		
	4月	5月	6月	7月	8月	9月	10月	11月	12月	1月	2月	3月
1. 諸条件整理（点群撮影許諾など）	←	→										
2. 点群データ取得、サーフェス化、BIM化		←	—	→								
3. データ統合化				←	—	—	→					
4. 仮想空間構築、アプリ開発					←	—	—	→				
5. 統合データ 補正、検証						←	—	—	→			
6. 実証実験							←	→				
7. 報告書作成									←	—	—	→

Ⅰ．実証概要

Ⅱ．実証技術の概要

Ⅲ．実証システム

Ⅳ．実証技術の検証

Ⅴ．成果と課題

活用技術一覧

本実証実験で利用した技術は以下の通り

項目	内容
CloudCompare	• 点群の編集や変換、間引き処理に広く使われるソフトウェア
Metashape	• 点群データからメッシュ生成が可能な点群処理ソフトウェア
Blender	• オープンソースの統合型3DCG制作ソフトウェア
Unity	• スマートフォンなど様々なプラットフォームに対応した高品質な3Dコンテンツ制作を可能とするゲームエンジン（3D制作プラットフォーム）
Unreal Engine	• リアリティを追求した高品質な3Dコンテンツ制作を可能とするゲームエンジン（3D制作プラットフォーム）
USD（Universal Scene Description）	• 様々なソフトウェアやハードウェアに対応した高機能かつ拡張可能な3Dのシーンを記述可能なファイルフォーマット
パーソナルモビリティ WHILL	• 外部機器による制御が可能な車いす型の電動モビリティ

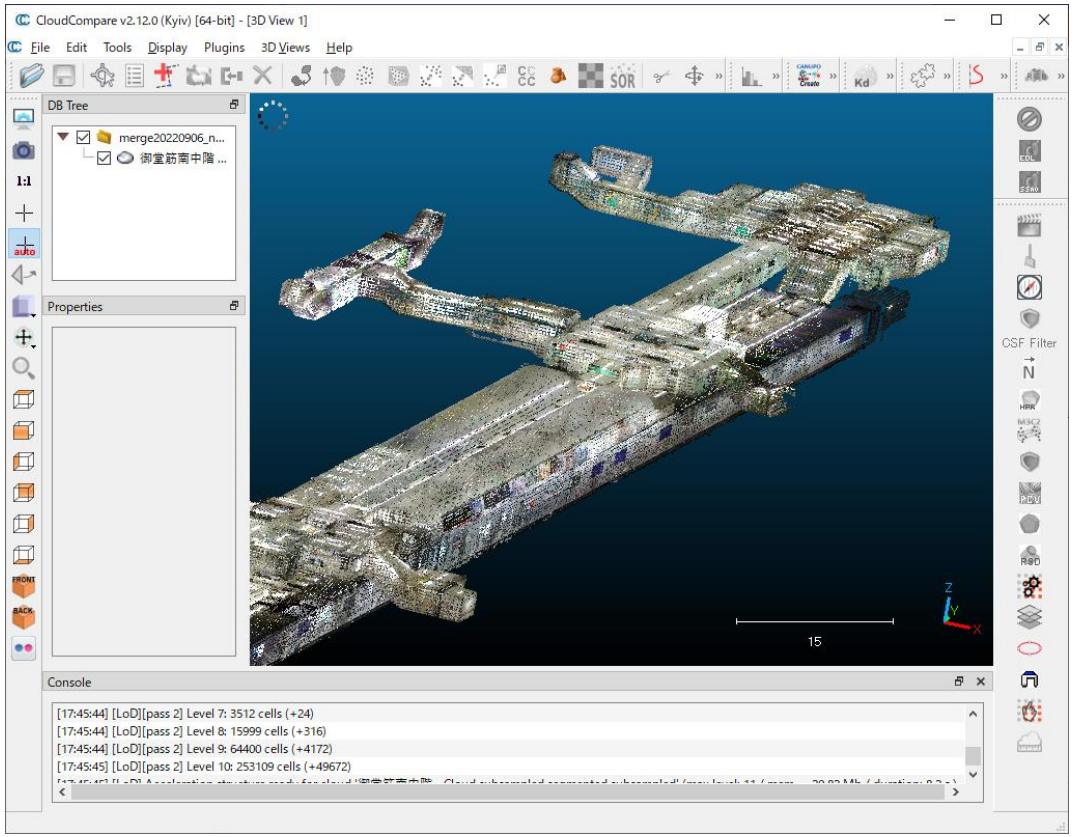
CloudCompareについて

点群の編集や変換、間引き処理に広く使われるソフトウェア

概要

項目	詳細
名称	CloudCompare
概要	• 多様な形式の点群を読み込み、各種処理を行うことができる • 変換したデータを多くの形式で保存可能
主な機能	• 様々な形式の点群ファイルを読み込み、編集を行った上でデータ変換を行う
本ユースケースで利用する機能	• 3Dスキャンした点群データ(LAS)の統合 • 点群データの軽量化 • BIM、3D都市データとの位置合わせ

点群処理画面



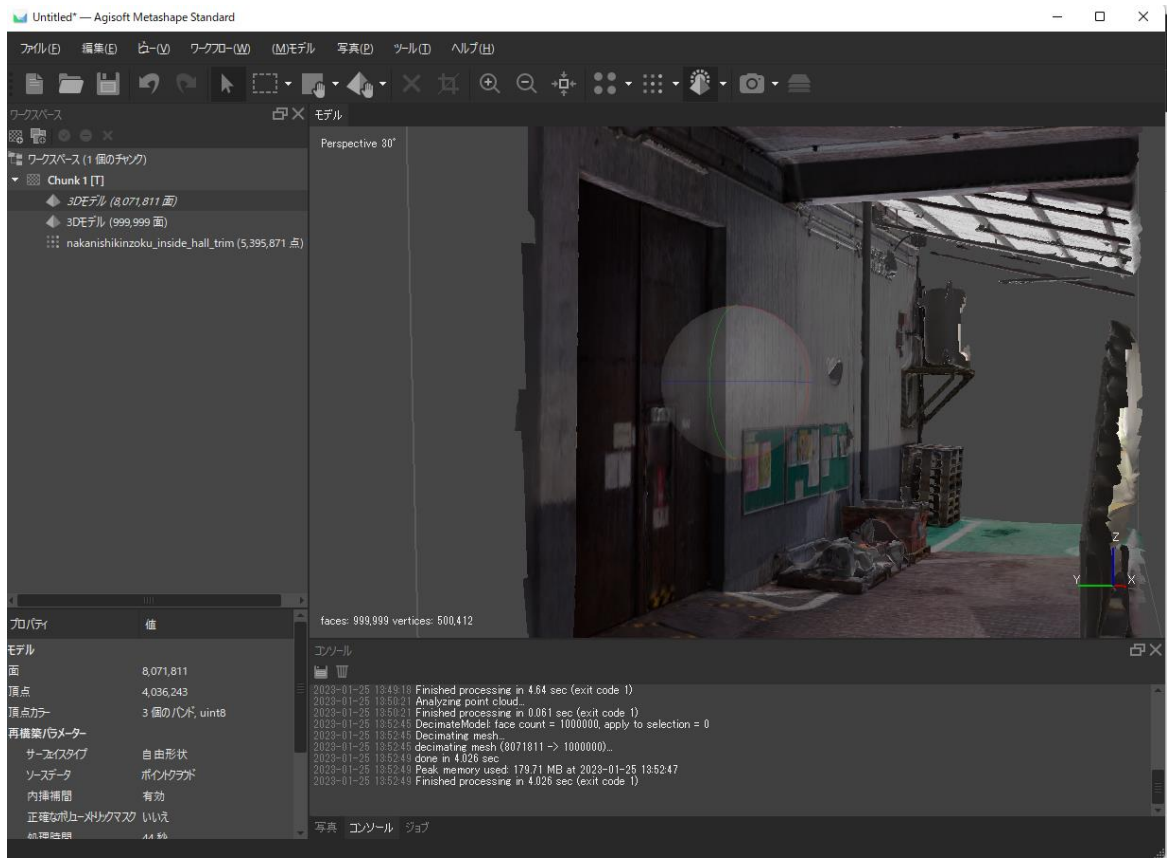
Metashapeについて

点群データからメッシュ生成が可能な点群処理ソフトウェア

概要

項目	詳細
名称	Metashape Standard
概要	• 写真測量用のソフトウェア • 静止画像から、点群、テクスチャーポリゴンモデルなどの生成が可能
主な機能	• 複数枚の静止画像の特徴点から、3Dモデルを生成 • 特徴点を点群に変換し、点群からメッシュを生成。画像からテクスチャを生成
本ユースケースで利用する機能	• 点群データからのメッシュ構築 • メッシュ生成後の浮遊ノイズの除去 • メッシュのリダクション

メッシュ生成画面



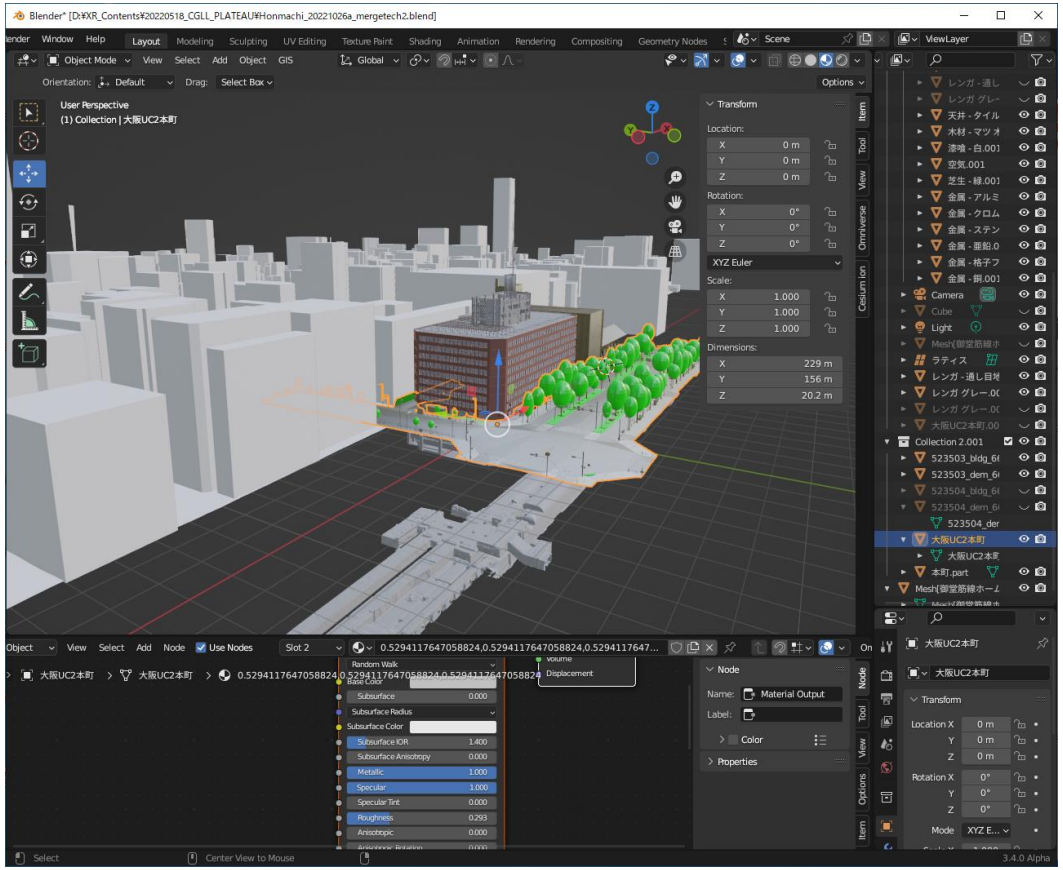
Blenderについて

オープンソースの統合型3DCG制作ソフトウェア

概要

項目	詳細
名称	Blender
概要	• 3Dのモデリング、レンダリング、アニメーション等幅広い用途に利用可能
主な機能	• 3Dモデリング • 3Dレンダリング • データ変換
本ユースケースで利用する機能	• 取り込んだ各種データ(BIM/3D都市モデル/点群の位置合わせ • データの修正 • データの変換と出力

3D処理画面



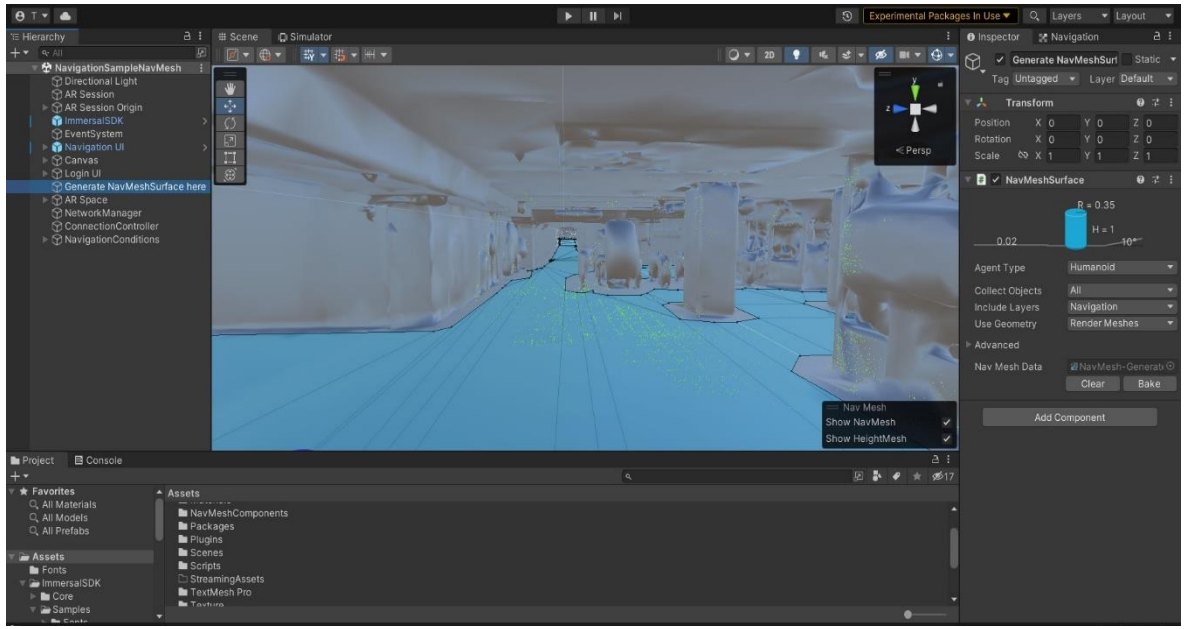
Unityについて

スマートフォンなど様々なプラットフォームに対応した高品質な3Dコンテンツ制作を可能とするゲームエンジン（3D制作プラットフォーム）

概要

項目	詳細
名称	Unity
概要	• Unity Technologiesによって開発・提供されるゲームエンジン（3D制作プラットフォーム） • マルチプラットフォーム対応で様々なデバイス向けの制作が可能
主な機能	• 高品質なリアルタイム3D描画機能 • ゲーム開発向けリアルタイム空間処理機能（経路探索等） • ゲームキャラクタ向け判断制御機能
本ユースケースで利用する機能	• リアルタイム3D描画機能 • ナビメッシュ生成・経路探索機能 • AR描画機能

操作画面



Unreal Engine について

リアリティを追求した高品質な3Dコンテンツ制作を可能とするゲームエンジン（3D制作プラットフォーム）

概要

項目	詳細
名称	Unreal Engine 4.27
概要	• 米 Epic Games社によって開発・提供されるゲームエンジン（3D制作プラットフォーム） • リアリティを追求したグラフィック表現が可能
主な機能	• 高品質なリアルタイム3D描画機能 • ゲーム開発向けリアルタイム空間処理機能（経路探索等） • ゲームキャラクタ向け判断制御機能
本ユースケースで利用する機能	• リアルタイム3D描画機能 • ナビメッシュ生成・経路探索機能

UI例：Unreal Engine Editor

https://docs.unrealengine.com/4.27

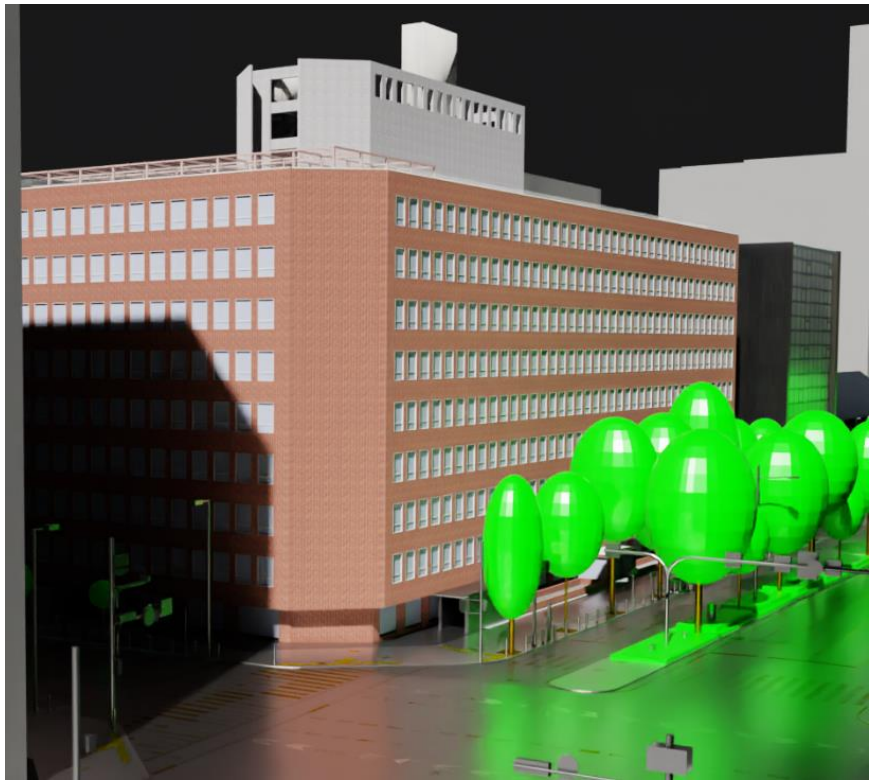
USD（Universal Scene Description）について

様々なソフトウェアやハードウェアに対応した高機能かつ拡張可能な3Dのシーン*を記述可能なファイルフォーマット

概要

項目	詳細
名称	USD（Universal Scene Description）
概要	• 3Dのシーン*を記述可能なファイル形式
主な特徴	• 様々なソフトで利用可能な比較的新しいファイル形式であるが、対応しているソフト、ハードが多い • 新しいファイル形式なので、高機能かつ拡張可能となっている
本ユースケースで利用する理由	• Unity、Unreal Engine双方でのインポートに対応しており、データ統合側で別々のファイルを作成する必要がない • リアルタイムレイトレーシングでデータを確認できるビューワーが用意されているので、高品質なCGでデータの確認ができる

利用イメージ



*シーンには、3Dモデル、レイアウト、アニメーション、仮想カメラなどのジオメトリおよび素材の外観などが含まれる

WHILL Model CRについて

外部機器による制御が可能な車いす型の電動モビリティ

概要

項目	詳細
名称	WHILL Model CR
概要	• ジョイスティックによる直感的な操作で快適な走行が可能な近距離向け電動モビリティの研究開発モデル
主な機能	• ジョイスティック操作による手動走行 • 外部接続端子に機器を接続することで、情報の取得（速度、移動距離等）および制御入力（ジョイスティック入力の代替等）が可能
本ユースケースで利用する機能	• 外部接続端子経由での情報取得（移動距離等） • 外部接続端子経由での制御入力（速度・旋回司令）

外観イメージ*

*出典：https://whill.inc/jp/model-cr

Ⅰ．実証概要

Ⅱ．実証技術の概要

Ⅲ．実証システム

Ⅳ．実証技術の検証

Ⅴ．成果と課題

実証フロー

実証実験では3Dモデルの統合手法を開発、実際にデータ統合を行った後に、アプリケーションとしてARナビ、モビリティを用いた運用の実証を行い、現地で想定した精度内での運用検証を行う

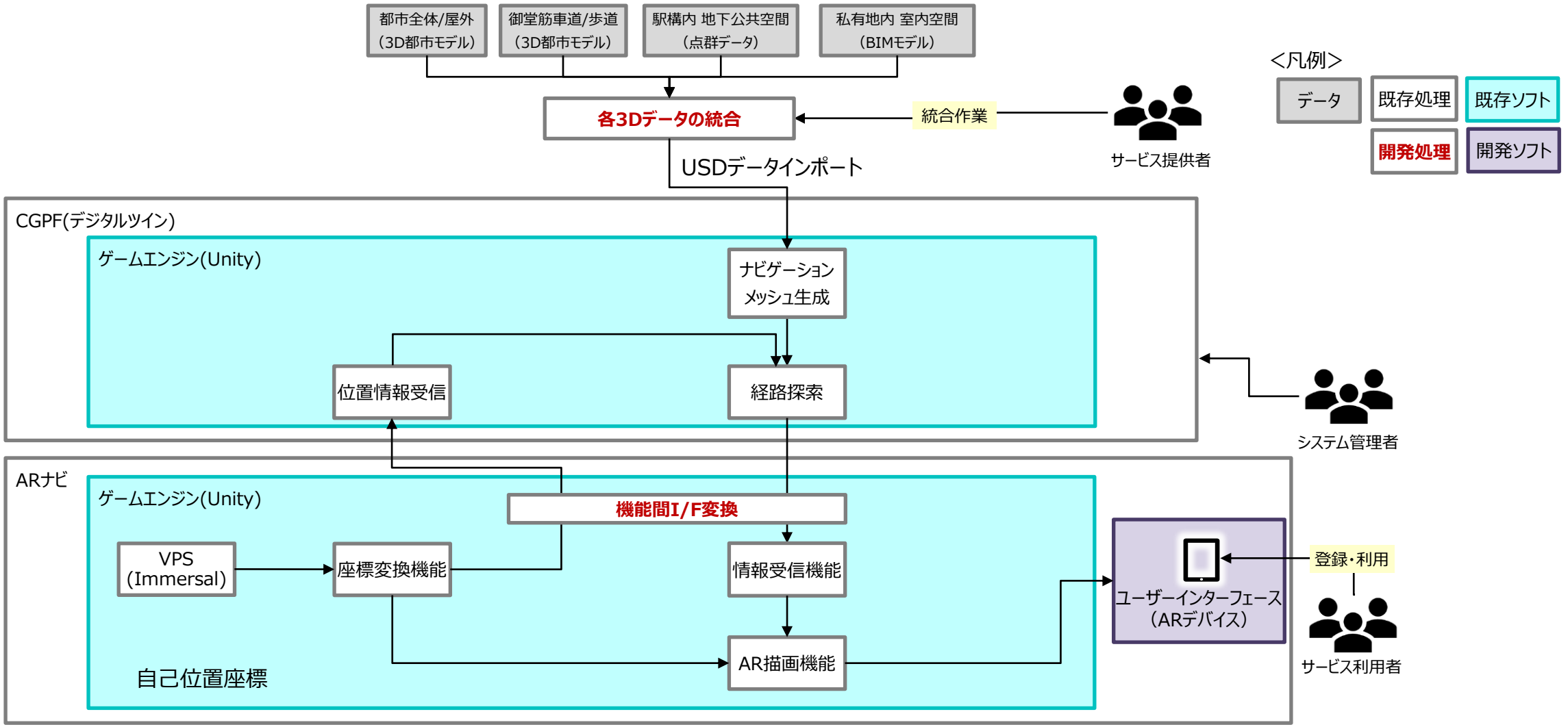
モデル統合手法の開発	• LODの異なるモデル、データ形式の異なるモデル（屋内BIMモデル・点群データと屋外建物情報データ）を統合する手法を構築する • 構築した統合手法を用いてデジタルツイン環境を作成する
モデル統合手法の検証	• 統合手法ごとに下記を検証する - デジタルツイン上のルートと、現実のルートとの誤差検証 - ARナビを実現する上で必要となるデータの検証 - 走行禁止エリア情報等、パーソナルモビリティ運行に必要となる情報の有無を検証
運用検証	• 実空間にてデジタルツイン上で設定したルート案内をもとに空間移動する - モデル結合が上手くいかなかったことで、間違った看板表示等がされてないか確認する - モデル上に表現されていない段差や狭い通路等、バリアフリーなルートとして不適切な事項が無いか確認する
結果比較	• ルート案内、パーソナルモビリティ運用実証におけるゴールまでの到達状況の確認とゴール地点での目標誤差検証を実施する

想定事業機会

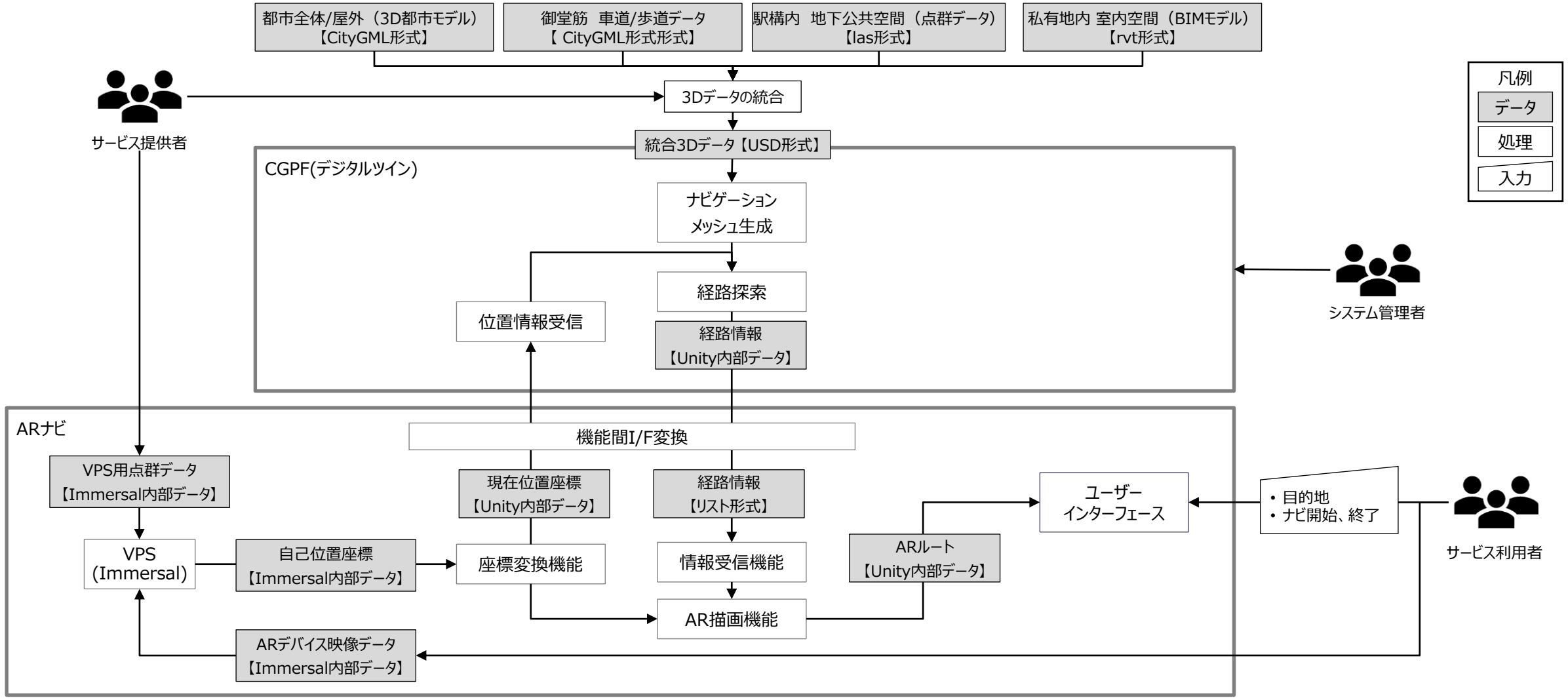
まちづくりの中で空間情報を利用する不動産デベロッパー、自治体、鉄道事業者などが想定利用者となり、各事業者が持つ空間データを提供いただくことで、Webサービスなどの提供が事業機会として見込まれる

項目	内容
利用事業者	• 自治体 • 不動産デベロッパー • 鉄道事業者
提供価値	• 空間（建物内部から建物内部、駅舎などから建物内部）移動のシームレスなナビゲーション • 移動に支援が必要な方々へ、段差や階段を避けたバリアフリー動線等の移動情報の提供
サービス仮説	• 交通弱者ナビゲーションサービス、ロボット移動支援 - 移動のための障害（段差情報、階段、エレベータ利用）を踏まえたナビゲーション提供 • スタンプラリーナビゲーション - 店舗、建物を巡るナビゲーション提供を行い、まちでの周回を促すサービス提供

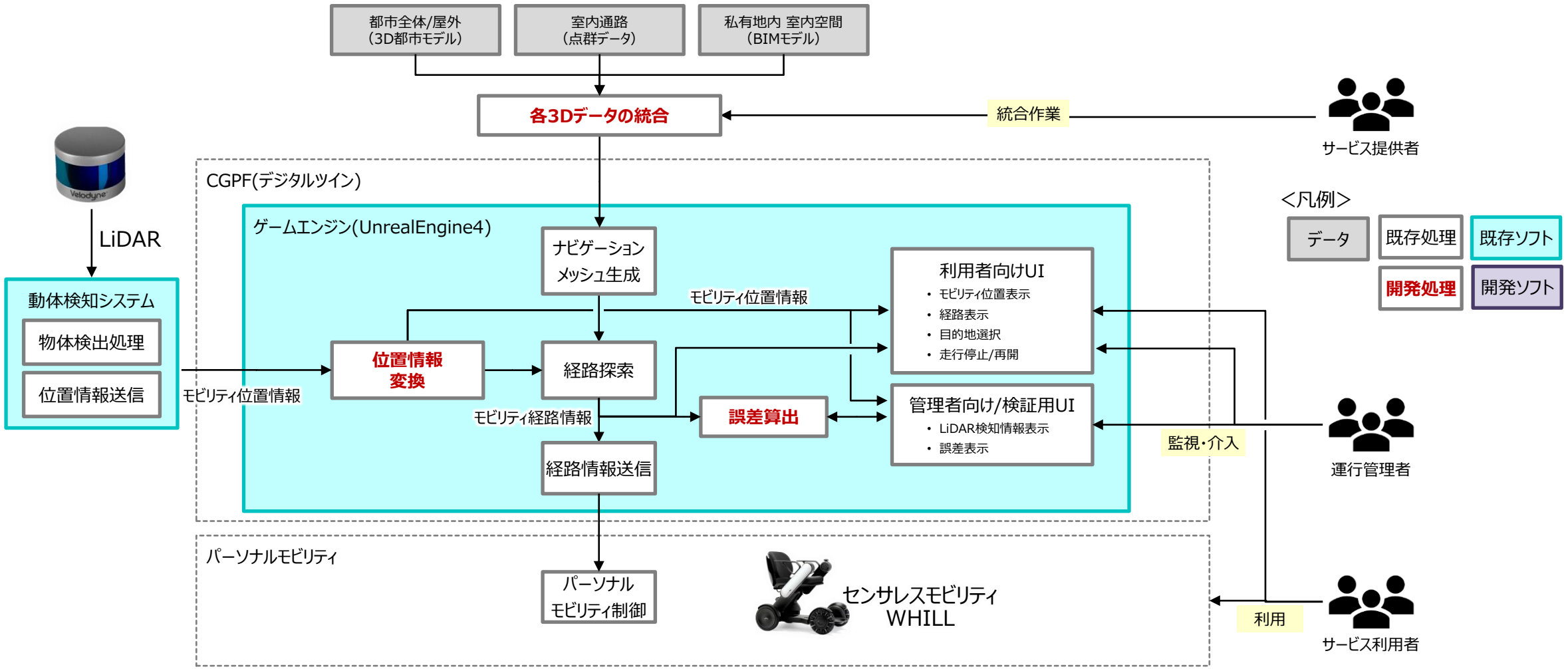
システムアーキテクチャ全体図｜ARナビ



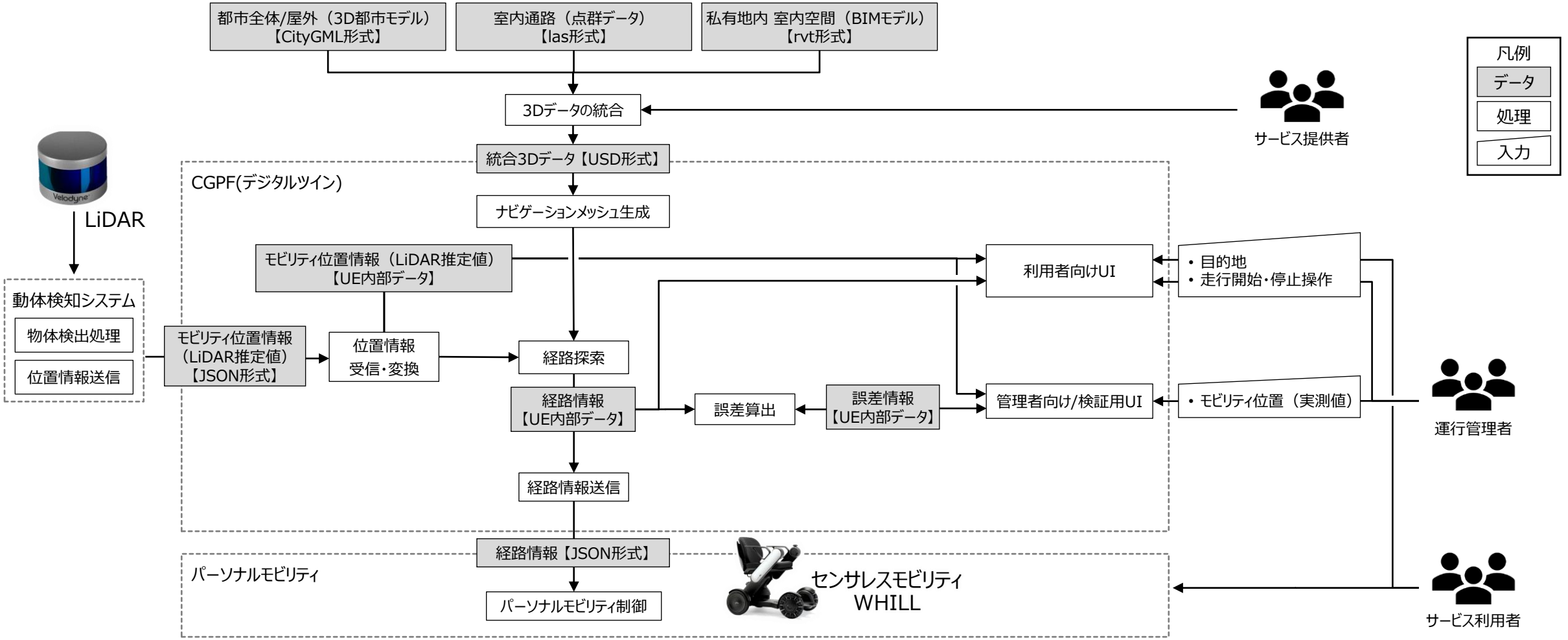
Ⅲ. 実証システム ＞ 3. アーキテクチャ全体図
データアーキテクチャ全体図｜ARナビ



システムアーキテクチャ全体図｜パーソナルモビリティ運用



Ⅲ. 実証システム ＞ 3. アーキテクチャ全体図
データアーキテクチャ全体図｜パーソナルモビリティ運用



システム機能一覧｜ARナビ

＜凡例＞**赤太字：新規開発機能**

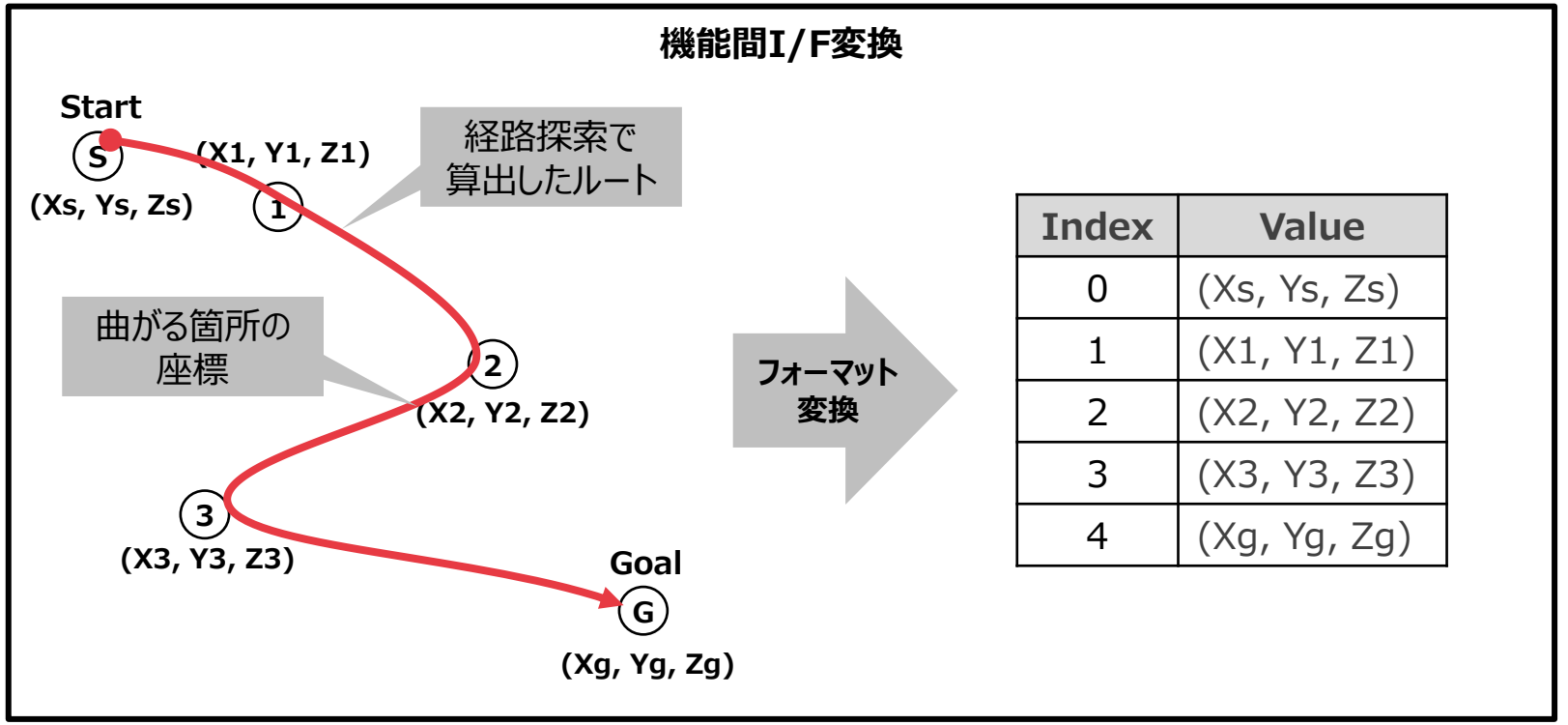
項目	内容
機能間I/F変換	• 既存機能同士を連携するために機能間でデータフォーマットを変換
ナビゲーションメッシュ生成	• 仮想空間上の3Dデータからナビゲーションメッシュ（通行可能領域）を自動生成
経路探索	• ナビゲーションメッシュ上で、ARナビの現在地から目的地までの走行経路を生成
位置情報受信	• ARナビから自己位置情報を取得し、仮想空間に反映
VPS	• カメラ画像を基に自己位置を推定
情報受信機能	• Common Ground PFから経路情報を受信
AR描画機能	• ナビゲーションのルートをARで可視化
座標変換機能	• VPSから取得した自己位置座標をCommon Ground PF側の座標に変換

ARナビ｜機能間I/F変換

CGPF(デジタルツイン)とARナビを連携するために、CGPF(デジタルツイン)の経路探索機能で算出した経路探索結果をARナビのルート表示用のデータフォーマットに変換する

CGPF(デジタルツイン)　　ARナビ

経路探索



フォーマット変換

Index	Value
0	(Xs, Ys, Zs)
1	(X1, Y1, Z1)
2	(X2, Y2, Z2)
3	(X3, Y3, Z3)
4	(Xg, Yg, Zg)

リスト化した座標から
ARでルートを描画

ARナビのルート表示用に、ナビゲーションメッシュ上で経路探索したルートから
スタート、曲がる箇所、ゴールの座標を取得し、リスト形式に変換する

システム機能一覧｜パーソナルモビリティ運用

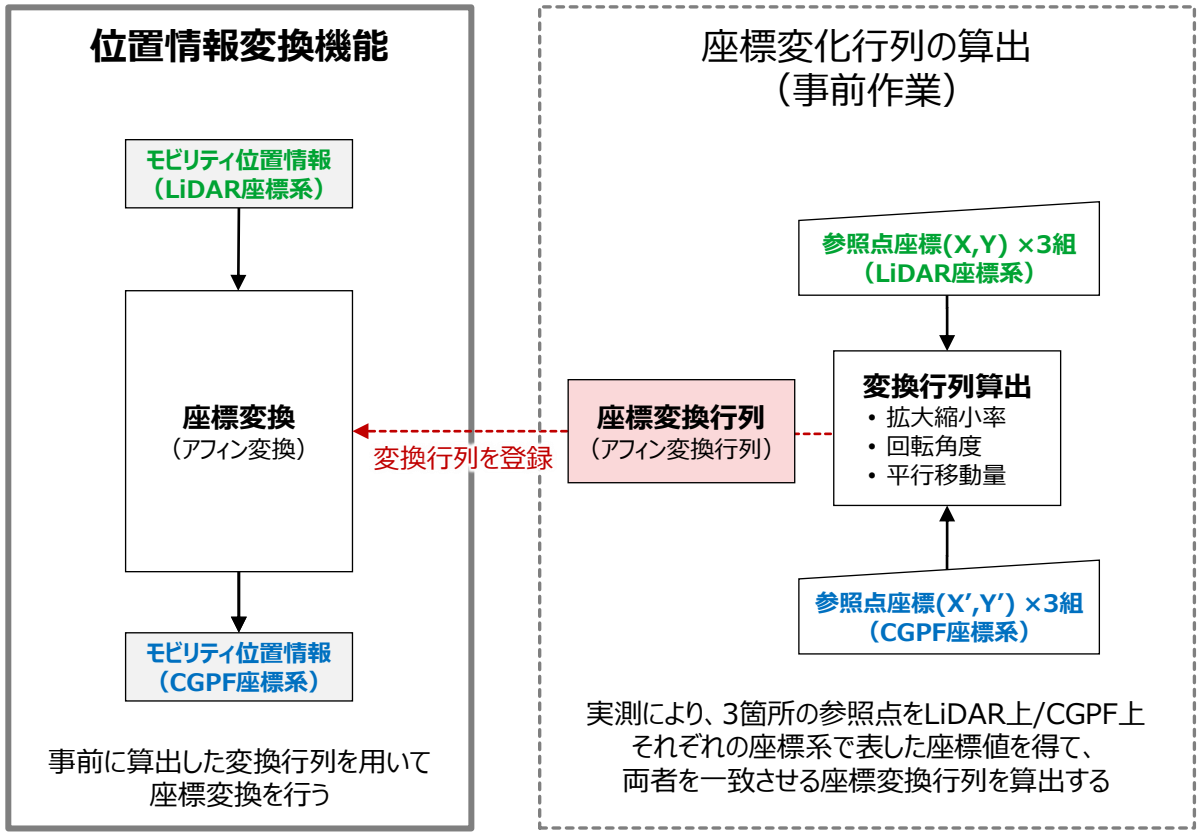
＜凡例＞**赤太字：新規開発機能**

項目	内容
物体検出処理	• 複数LiDARの点群を統合し、移動物体（パーソナルモビリティ）を検出
位置情報送信	• 移動物体（パーソナルモビリティ）の位置を、CGPF（デジタルツイン）に送信
位置情報受信・変換	• モビリティの位置情報を受信し、統合3Dデータを配置したCGPF（デジタルツイン）上の座標に変換
ナビゲーションメッシュ生成	• 統合3Dデータの形状情報を解析し、モビリティの走行可能領域（ナビゲーションメッシュ）を生成
経路探索	• ナビゲーションメッシュ上で、モビリティ現在位置と目的地を結ぶ最短経路を探索
経路情報送信	• 経路情報を、経由地点の座標点列としてモビリティへ送信
パーソナルモビリティ制御	• 経路情報（経由地点の座標点列）を受信し、モビリティを経路に沿って走行させる
誤差算出	• 目標経路に対するモビリティ実測位置の誤差（横方向偏差）を算出
利用者向けUI	• 統合3Dデータの仮想空間上に、モビリティ位置、目標経路を表示 • 目的地、モビリティの走行（停止・開始）を指示する入力を受け付ける
管理者向け/検証用UI	• LiDARで検出したモビリティの位置を、統合3Dデータの仮想空間上に重畳表示 • モビリティ位置の実測値の入力を受け付け、目標経路に対する誤差（逸脱量）を表示

パーソナルモビリティ運用｜位置情報受信・変換

LiDARによる動体検知システムよりモビリティの位置情報を受信し統合3Dデータを配置したCGPF（デジタルツイン）上の座標に変換する

位置座標変換の流れ



アフィン変換

アフィン変換を用いて、LiDAR座標系からCGPF（デジタルツイン）座標系へモビリティ座標の変換を行う

変換後の座標（CGPF座標系）　座標変換行列　変換前の座標（LiDAR座標系）

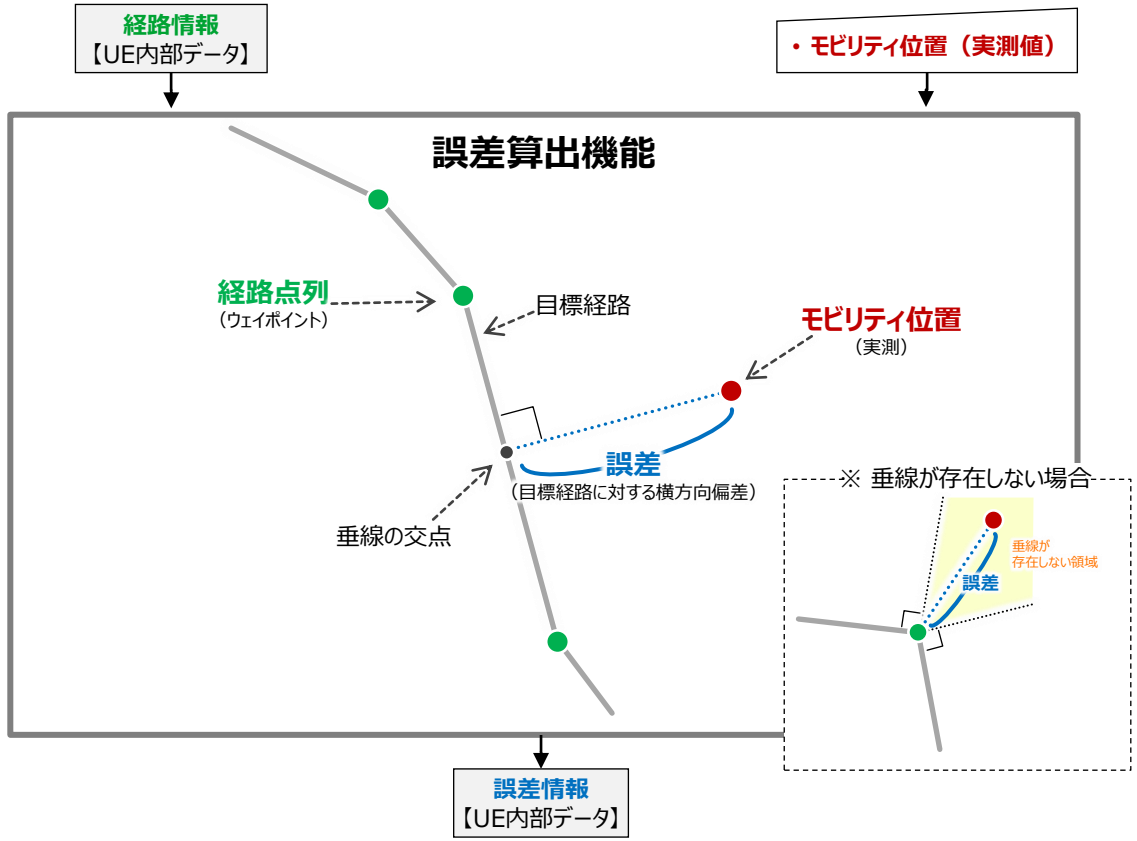
$$\begin{bmatrix} X' \\ Y' \\ 1 \end{bmatrix} = \begin{bmatrix} \lambda\cos\theta & -\lambda\sin\theta & T_x \\ \lambda\sin\theta & \lambda\cos\theta & T_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

λ：拡大縮小率
θ：回転角度
T_x, T_y：並行移動量

パーソナルモビリティ運用｜誤差算出

目標経路とモビリティの実測位置から目標経路に対するモビリティ位置の誤差（横方向偏差）を算出する

誤差算出イメージ



誤差算出方法

① 経路点列（ウェイポイント）の間を線形補完し、目標経路とする

② 目標経路に対して、モビリティ位置（実測）から垂線を降ろし、交点を求める
※複数の経路線分に対して垂線が存在する場合は、最も近いものを採用
※いずれの経路線分に対しても垂線が存在しない場合は、モビリティ位置から最も近いウェイポイントを交点として採用

① モビリティ位置（実測）から垂線の交点までの距離を、目標経路に対するモビリティ位置の誤差（横方向偏差）として出力

3D都市モデル一覧

地物	地物型	属性区分	属性名	内容
建築物LOD3	bldg:Building	空間属性	bldg:lod3Solid	建築物のLOD3立体
			bldg:lod1Solid	建築物のLOD1立体
道路LOD3	tran:Road		tran:lod3MultiSurface	道路のLOD3立体
地形LOD1	dem:ReliefFeature		dem::TINRelief	地形のLOD1立体
都市設備LOD2	frn:CityFurniture		frn:lod2Geometry	都市設備のLOD2立体
植生LOD3	veg:SolitaryVegetationObject		veg:lod3Geometry	植生（単独木）のLOD3立体
	veg:PlantCover		veg:lod3MultiSurface	植生（植被）のLOD3立体

その他　活用データ一覧

活用データ	内容	データ形式	諸元
点群データ（本町）	大阪メトロ御堂筋線本町駅地下部分の点群データ	las	新規スキャン
点群データ（天満）	CGLL及び屋内通路の点群データ		
BIMモデル（本町）	御堂ビル（竹中工務店大阪本店）のBIMモデル	pla	竹中工務店
BIMモデル（天満）	CGLL及び屋内通路のBIMモデル	rvt	CGLL運営委員会
VPS用点群データ	ARナビを利用する範囲をImmersal Mapperで取得した点群データ	（内部データ）	新規スキャン

点群データ｜本町・天満

対象施設内のルート探索のため、点群データを取得した

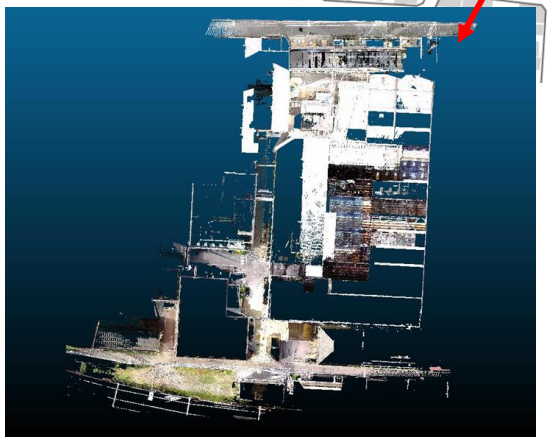
取得方法

項目	内容
実施業者	クモノスコーポレーション
使用機器	Faro Focus
機器の範囲誤差	±1mm@10mm
空間的な誤差	±3~5mm

- 機械は年1回のメーカ校正があり、校正を受けた機器を使用
- 合成ソフトを使用して合成
- 3Dレーザスキャナの特性上、毎回同じポイントをデータとして取得するわけではないため、1点や2点間での比較が難しく空間的な誤差の結果、その空間内に含まれる任意の距離確認もその精度内に収まっていると判断

本町の点群データ

天満（CGLL）の点群データ



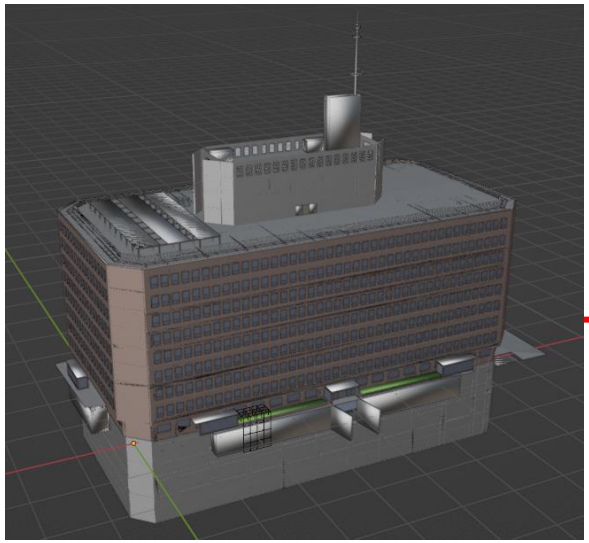
BIMモデル｜本町

対象施設敷地内及び駅構内のルート探索のために、BIMデータを利用

概要

項目	内容
対象建物	御堂ビル（竹中工務店大阪本店）
作成ソフトウェア	ARCHICAD 24
データ形式	.pla
データ精度	±3mm
備考	建物改修の際に作成

データイメージ



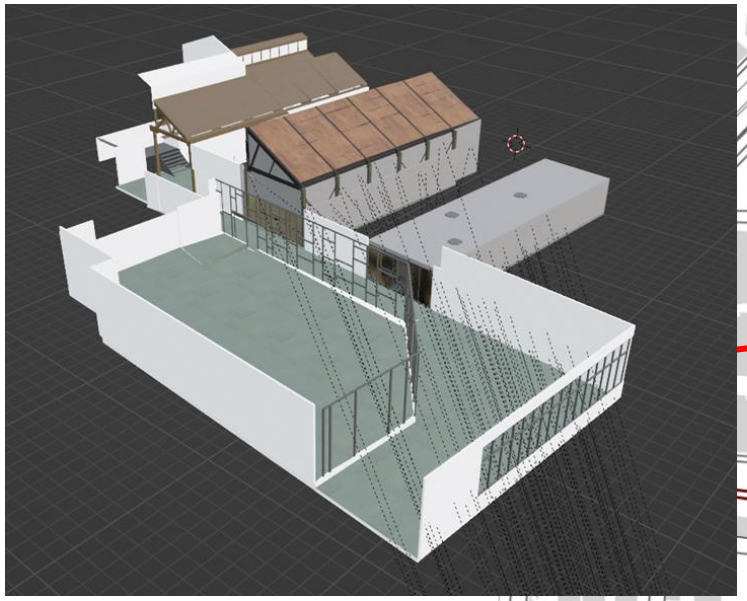
BIMモデル｜天満

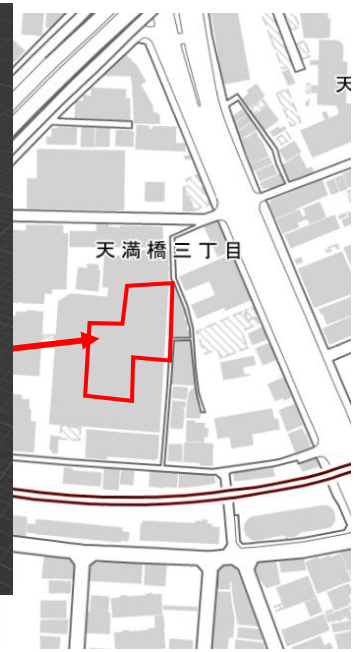
対象施設敷地内及び駅構内のルート探索のために、BIMデータを利用

概要

項目	内容
対象建物	コモングラウンド・リビングラボ（CGLL）
作成ソフトウェア	Revit 2019
データ形式	.rvt
データ精度	±3mm
備考	設計時に作成

データイメージ





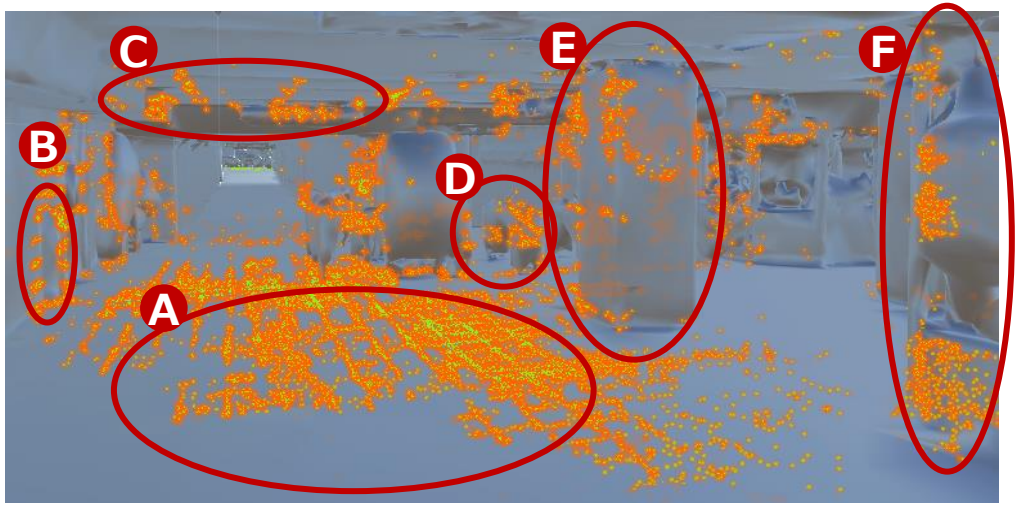
VPS用点群データ｜統合データとの位置合わせ

ARナビアプリユーザの実空間の位置を3D統合データ上に反映させる方法としてVPSを用いた。その詳細について記載する。仮想空間上の統合モデルに対し、自己位置推定用の点群を手動で配置した

VPS用点群データと統合データとの位置合わせ

- VPSによる自己位置推定位置と統合データ上の位置を合わせるため、各撮影ポイントにおけるVPS用点群データを統合データ上に配置し、位置合わせを行う
- VPS用点群データは以下の観点を踏まえて配置した
  - 統合モデルの地面と点群の地面の高さが合っているか
  - 進行方向に対して点群の位置が、統合モデルの各特長の位置に重なるようになっているか
- 10か所の点群撮影ポイントについてそれぞれ位置合わせを実施した

位置合わせの一例：点群撮影ポイント①（本町駅改札前）



#	確認ポイント
A	地面と点群の高さが一致すること
B	壁際の消火器に点群が一致すること
C	天井の看板等に点群が一致すること
D	改札に点群が一致すること
E	支柱の角部分に点群が一致すること
F	エレベータ側面の壁の凹凸に点群が一致すること

VPS用点群データ｜点群撮影ポイント

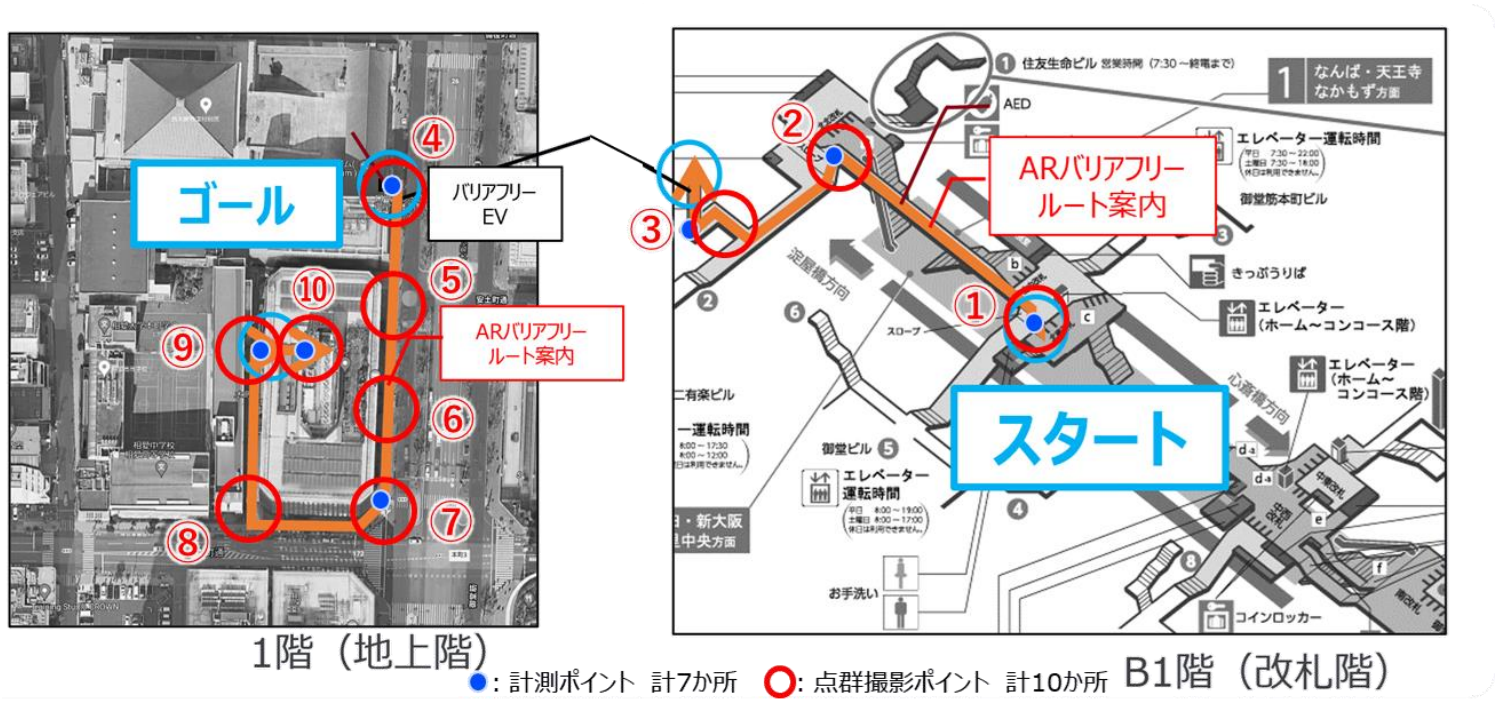
ARナビアプリで利用するVPSのImmersalに必要な点群データを10か所の赤丸の地点でImmersal Mapperを使って取得した

撮影箇所の選び方

次の基準で撮影箇所を10か所選定した

- スタート地点
  - アプリ開始時の自己位置推定が必要なため
- 曲がり角周囲
  - 曲がり角のナビゲーション精度を高めるため
- 直線距離が長い箇所の中間地点
  - 移動距離に応じて、iPadのモーションセンサの誤差が堆積するため

VPS用点群の撮影箇所



データ処理一覧（1/3）

システムに入力するデータ（データ形式）	用途	処理内容	データ処理ソフトウェア	活用データ（データ形式）
統合モデル（本町駅）（手法1）	ARナビゲーションに利用	• 開発した統合手法1によりデータを統合 • 統合データをUSD形式で出力	• FME Desktop • CloudCompare • Blender • Metashape	• 3D都市モデル（CityGML形式） • 点群データ（本町）（las形式） • BIMモデル（本町）（pla形式）
統合モデル（本町駅）（手法4）USD形式		• 開発した統合手法4によりデータを統合 • 統合データをUSD形式で出力		
統合モデル（天満　CGLL）（手法1）USD形式	パーソナルモビリティ運用に利用	• 開発した統合手法1によりデータを統合 • 統合データをUSD形式で出力		• 3D都市モデル（CityGML形式） • 点群データ（天満）（las形式） • BIMモデル（天満）（pla形式）
統合モデル（天満　CGLL）（手法2）USD形式		• 開発した統合手法2によりデータを統合 • 統合データをUSD形式で出力		
統合モデル（天満　CGLL）（手法3）USD形式		• 開発した統合手法3によりデータを統合 • 統合データをUSD形式で出力		

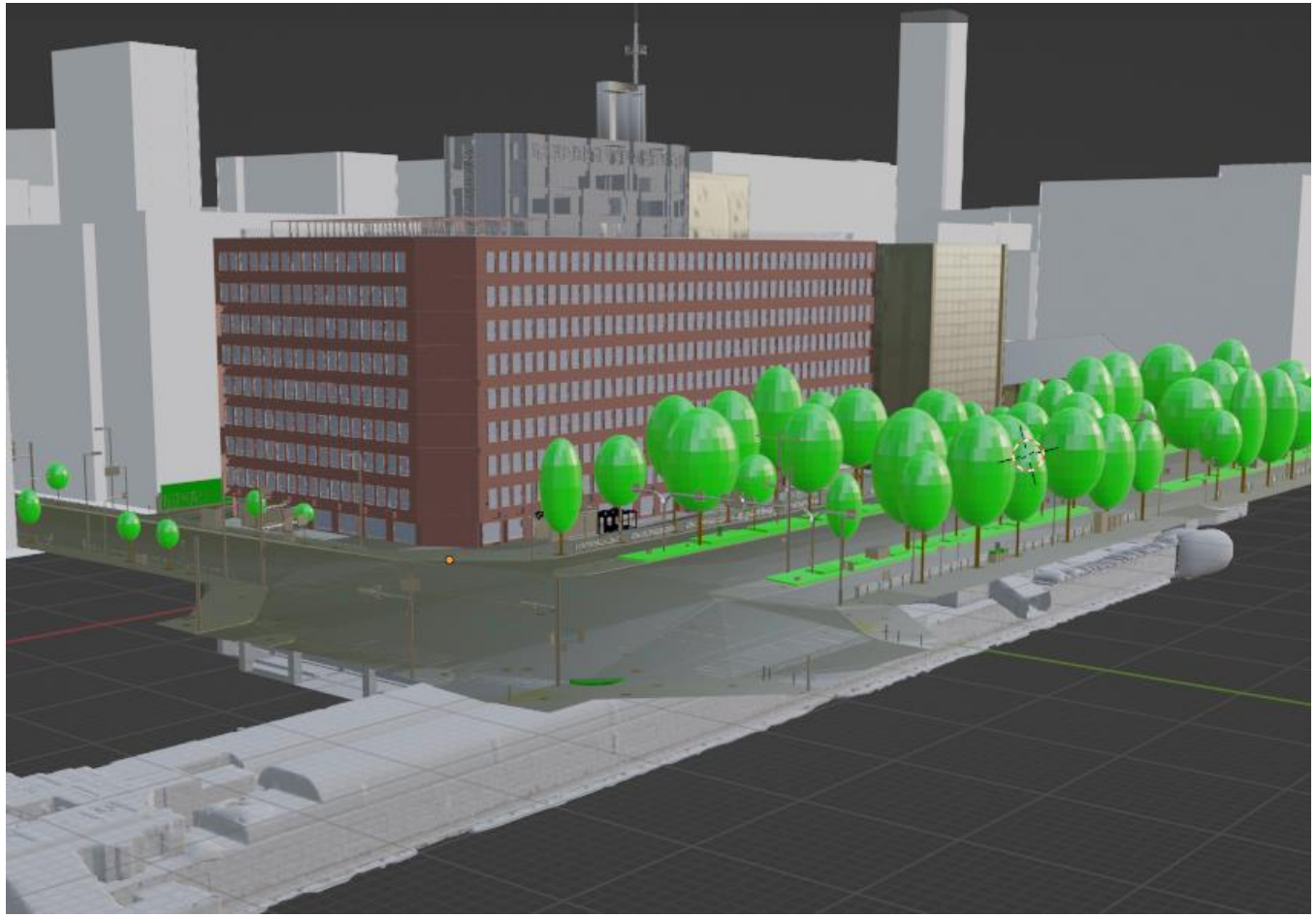
データ処理一覧（2/3）

システムに入力するデータ（データ形式）	用途	処理内容	データ処理ソフトウェア	活用データ（データ形式）
経路情報（Unity内部データ）	ARナビの経路として、UIの画面表示に使用	統合3Dデータ、現在位置座標をもとに、Unityの経路探索機能を用いて、最短経路を生成	Unity	• 統合3Dデータ（USD形式） • 現在位置座標（Unity内部データ）
自己位置座標（Immersal内部データ）	ARナビの推定現在位置として、経路探索に使用	VPSの自己位置推定機能を用いて、自己位置座標を生成	Immersal	• VPS用点群データ（Immersal内部データ） • ARデバイス映像（Immersal内部データ）
現在位置座標（Unity内部データ）	ARナビの推定現在位置として、経路探索に使用	VPSから取得した自己位置座標をCommon Ground PF側の座標に変換	Unity	自己位置座標（Immersal内部データ）
経路情報（リスト形式）	ARナビの経路として、UIの画面表示に使用	Common Ground PF側の経路情報をARナビの処理用にリスト形式に変換	Unity	経路情報（Unity内部データ）
ARルート（Unity内部データ）	ARナビの経路として、UIの画面表示に使用	経路情報をもとに、ARで可視化するルートオブジェクトに変換	Unity	経路情報（リスト形式）

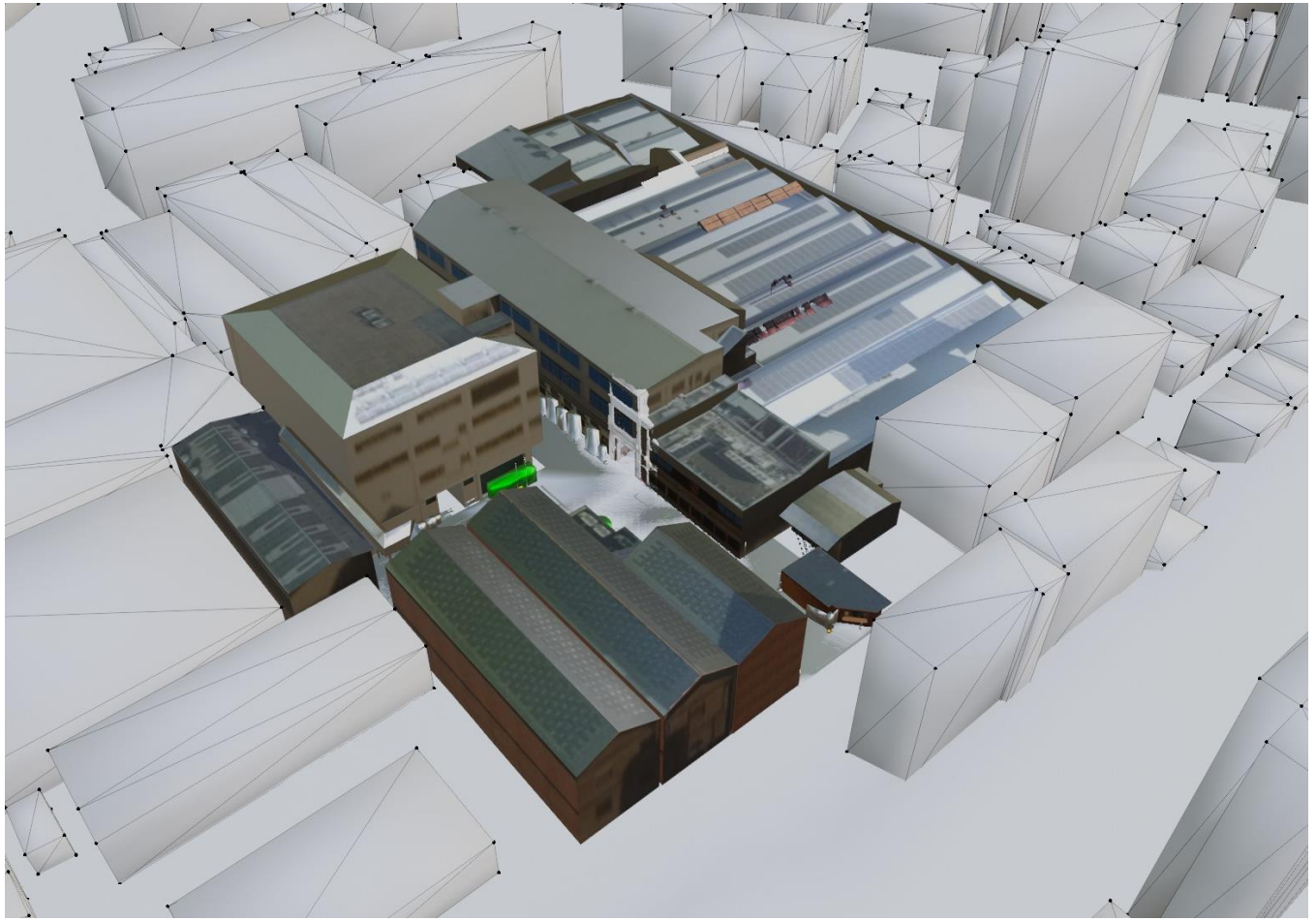
データ処理一覧（3/3）

システムに入力するデータ（データ形式）	用途	処理内容	データ処理ソフトウェア	活用データ（データ形式）
モビリティ位置情報（JSON形式）	モビリティの推定現在位置として、画面表示・経路探索に使用	なし	なし	モビリティ位置情報（動体検知システム出力）
モビリティ位置情報（UE内部データ）	モビリティの推定現在位置として、利用者向けUIの画面表示に使用	動体検知システム（LiDAR）の座標系から、Unreal Engineの座標系へ座標変換	Unreal Engine	モビリティ位置情報（JSON形式）
経路情報（UE内部データ）	モビリティの目標経路として、モビリティ制御、誤差算出および利用者向けUIの画面表示に使用	統合3Dデータをもとに、UnrealEngineの経路探索機能を用いて、最短経路の座標点列を生成	Unreal Engine	・統合3Dデータ（USD形式） ・モビリティ位置情報（UE内部データ）
経路情報（JSON形式）	モビリティの目標経路として、モビリティ制御に使用	UE内部表現の経路点列をJSON文字列に変換	Unreal Engine	経路情報（UE内部データ）
誤差情報	目標経路に対するモビリティの位置誤差として、検証用UIへの表示および記録に使用	目標経路情報と、運行管理者によって入力される誤差情報をもとに、モビリティの横方向偏差を算出	Unreal Engine	• 経路情報（UE内部データ） • モビリティ位置（運行管理者入力）

統合モデル（本町駅）



統合モデル（天満　CGLL）



各統合手法の特徴

手法1は異なるデータに対して同一の手法で統合し作業性の比較を行い、さらに統合に利用するモデルの特徴を踏まえて本町は手法4、天満は手法2と3で統合し検証を行った

統合手法	位置合わせの基準	位置合わせの方法	モデル上でのズレ	接続部の連続性	メリット	その他特徴	検証エリア	
							本町	天満 CGLL
手法1	原点と参照点を一つずつ設定	原点と参照点を合わせる	参照点付近はずれがすくないが、参照点から離れるほどずれは大きくなる	建物やモデルの間の境界で接合部分にずれが生じる	もっとも単純な接合手法のため、工数が少ない	目視での手作業	○	○
手法2	原点一つに対して複数参照点を設定	原点を合わせ、複数の参照点のズレが最小化されるように位置を合わせる	参照点ごとにずれを最小化するため、ズレの大きさは平均的	建物やモデルの間の境界で接合部分にずれが生じる	工数が少なく抑えられる（手法1と手法3との間）	統合のための移動・回転処理は自動で行う	–	○
手法3	点群データごとに原点を設置し、参照点も複数設置	出入口等の精度が必要な個所に原点を置き、参照点調整する	出入口等の精度が必要な個所のズレが小さい	点群データごとに重なる部分は調整幅を設けて接合するためずれが生じにくい	高精度が担保される	異なるモデルと重なる点群が必要	–	○
手法4	（手法2と同じ）			手法2に加えて、モーフィング（二つの要素を繋げて滑らかにする）で接合する	データ上の不連続な接続箇所を取り除くことができる	手法2の基本的な特徴を引き継ぐ	○	–

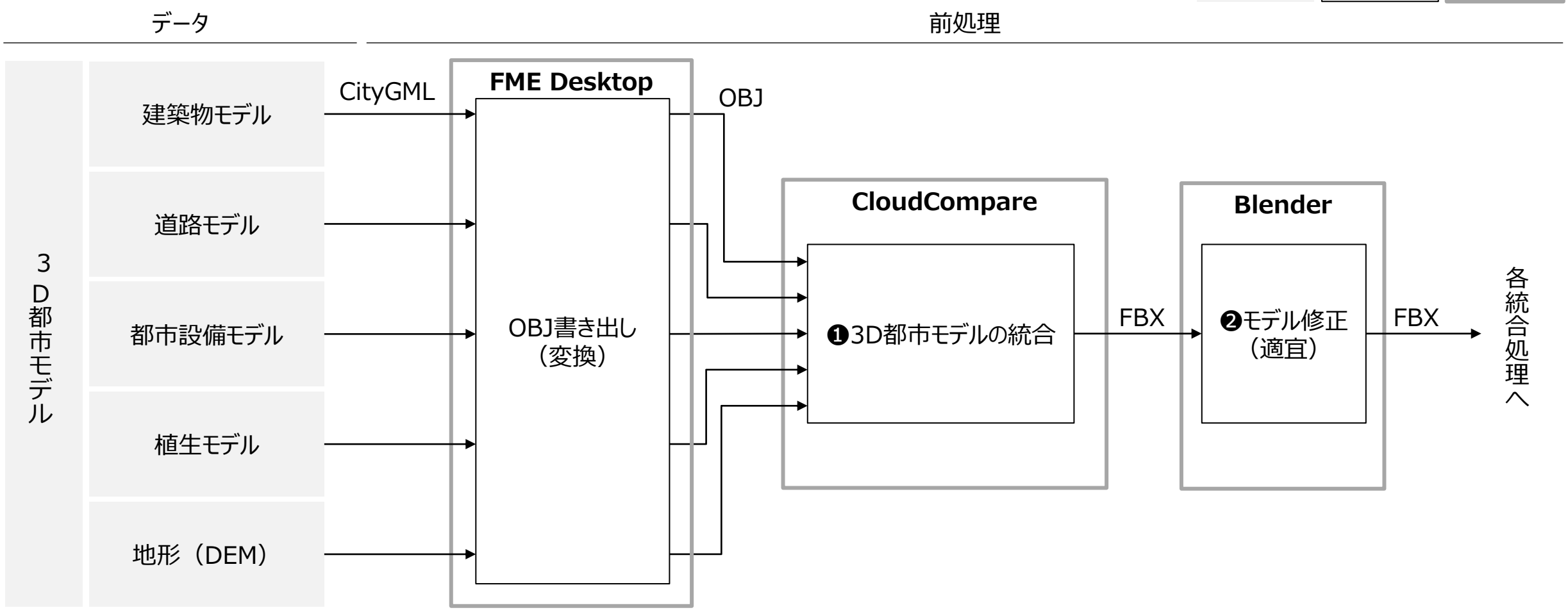
統合手法の全体像

3D都市モデル、点群データ、BIMモデルの3データをそれぞれ前処理した後に、統合手法1~4の手法により各データを統合する

①共通処理｜3D都市モデルの前処理

CityGML形式の3D都市モデルをFME Desktop等でOBJに変換した後、CloudCompareを使いすべての地物を統合する。その後、必要に応じてBlenderで地物間の重なりを除去するなどのモデル修正を行う

データ　処理　ソフトウェア



①共通処理｜3D都市モデルの前処理：❶3D都市モデルの統合

CloudCompareを使い、3D都市モデルの各地物を統合する

処理フロー

3D都市モデルを読み込み

- 変換したobjファイルをCloudCompareに読み込む
  - 座標系が同じため位置合わせは不要

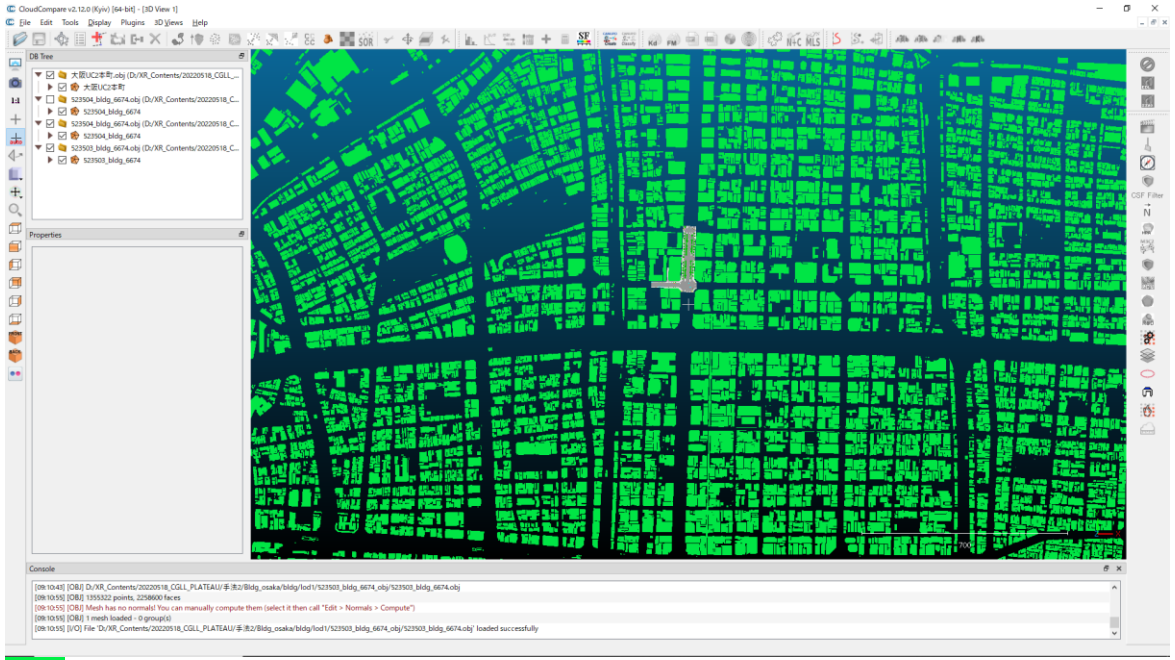
不要データを削除

- LOD1及びDEMモデルは広域のデータとなるので、必要な範囲をSegmentコマンドで切り分け、不要部分を削除

データを統合

- 1ファイルのFBXとして出力

処理のイメージ



3D都市モデルの建物

①共通処理｜3D都市モデルの前処理：❷モデル修正

Blenderを使い、3D都市モデルの地物間の重なり部分およびBIMモデルとの重複を削除する

作業フロー

フラットシェードの適用

- 3D都市モデルを読み込んだ際に、建物が丸く表現されることがあるため、「フラットシェード」を適用する
  - データへの影響はなく、表示方法が変更される

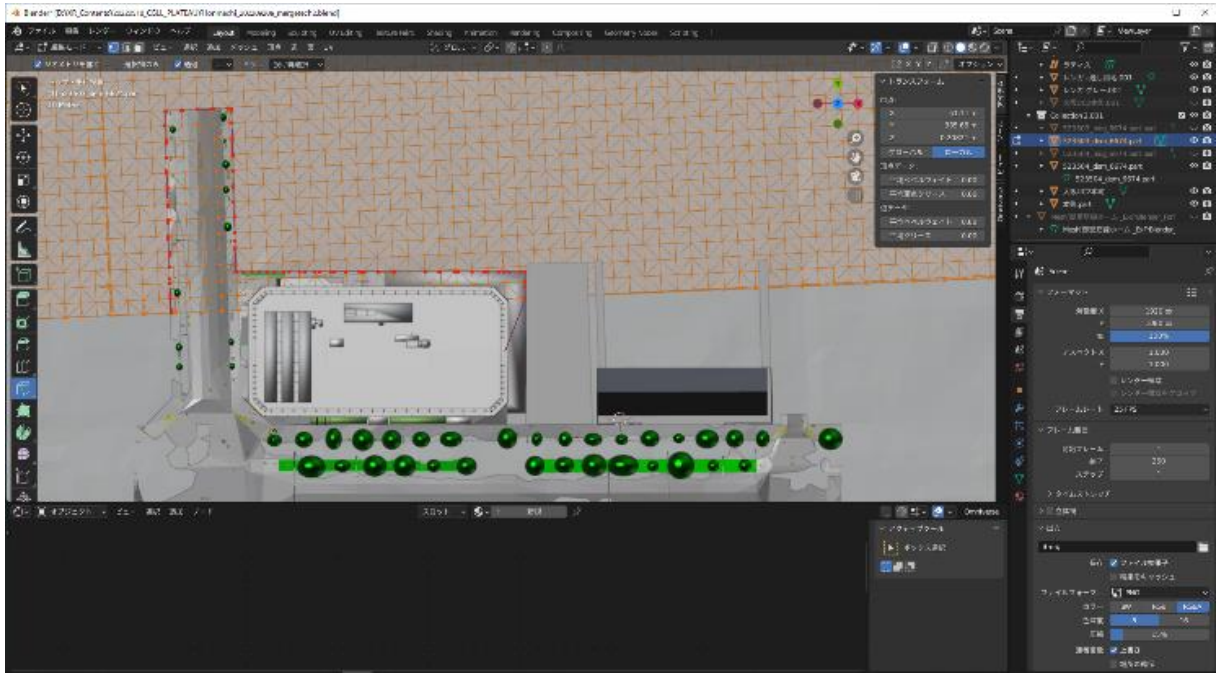
地形と道路の重複を削除

- 3D都市モデルの道路と地形（DEM）が重複する部分を地形から切り抜く

3D都市モデルとBIMデータで重複する建物を削除

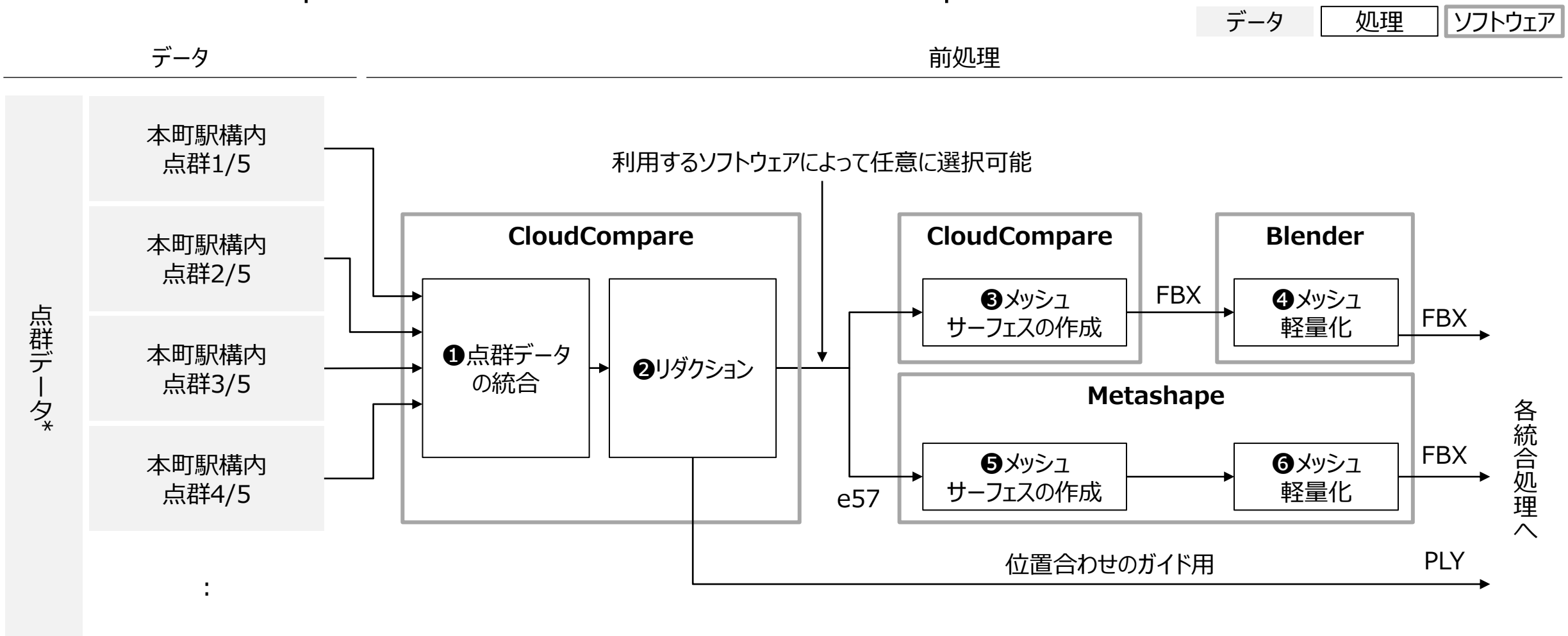
- 3D都市モデルの建物のうち、BIMモデルと重なっている建物を削除する

Blenderでのデータ切り取り



Ⅲ. 実証システム ＞ 5. データ ＞ ②データ処理
①共通処理｜点群データの前処理

複数の点群データの統合とリダクション処理をCloudCompareで実施した後、メッシュサーフェスの作成とメッシュの軽量化をCloudCompare・Blenderの組み合わせもしくはMetashapeによって実施する



*ファイルサイズの制限等で分割された点群データファイルを想定

①共通処理｜点群データの前処理：❶点群データの統合

CloudCompareを使い、複数ファイルに分割された点群データを統合する

作業フロー

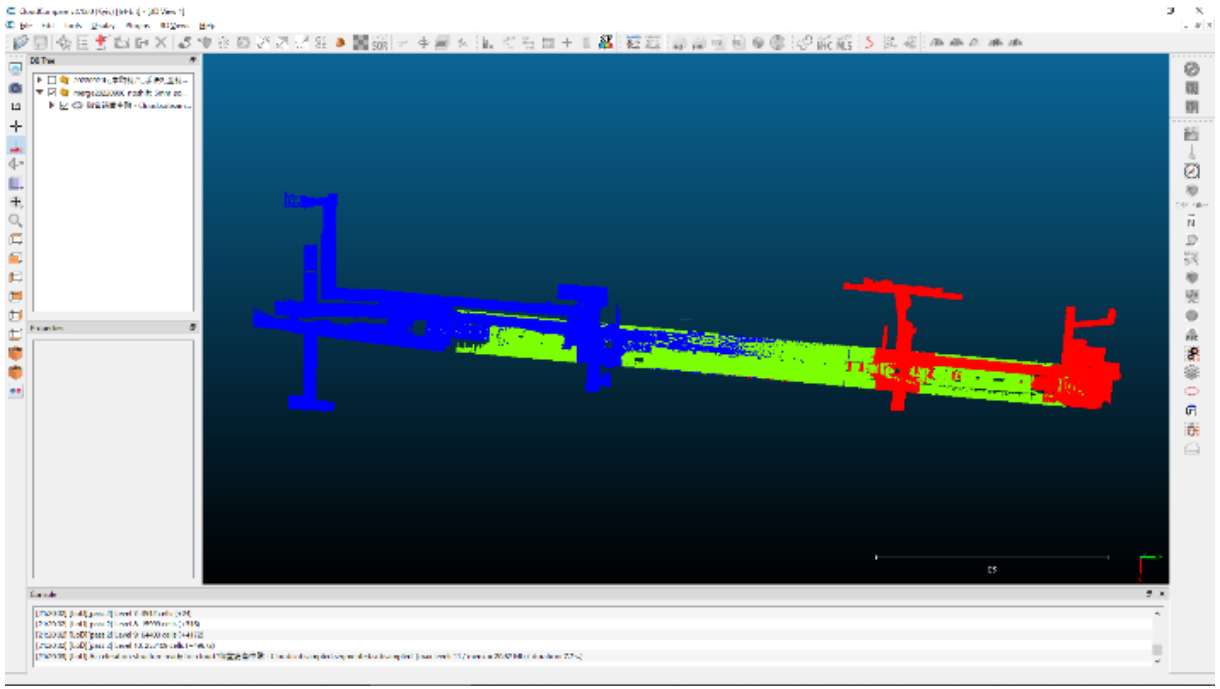
点群データの読み込み

- 分割された点群データファイルをCloudCompareに読み込む

データファイル統合

- ファイルサイズ都合でファイルが分かれているが、同時に計測した点群同士のため位置合わせは不要
  - データ読込時に点群全体を座標原点付近に移動するか確認メッセージが表示されるが、移動は不要

点群読み込みイメージ



大阪メトロ御堂筋線本町駅の点群データの分割イメージ
緑：ホーム、青：南中階、赤：北中階

①共通処理｜点群データの前処理：❷リダクション

点群データはそれぞれ数億点のデータとなっており、そのままでは処理が重いため、CloudCompareを使い、データ読込後に5mm間隔でリダクションし、さらに、統合後に25mm間隔で再度リダクションを行う

処理フロー

5mm間隔でリダクション

- 各データを5mm間隔でリダクション（点群の間引き）を行う
  - Method→「Space」、min.space between pointsで点の間隔を設定

データ統合

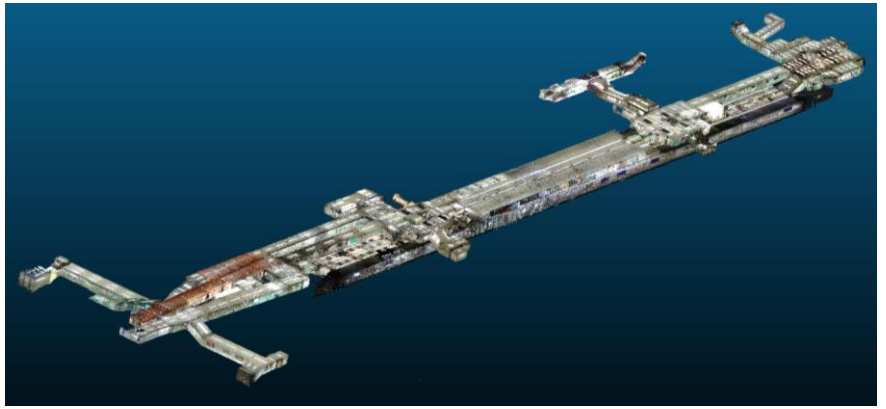
- 3つのデータを1つの点群データに統合

25mm間隔でリダクション

- 統合後の点群データを更に25mm間隔でリダクションを行う

リダクション前後の点の数

	取得時の点の数	統合前のリダクション結果	統合後のリダクション結果
①ホーム	431,997,509	136,708,348	-
②北中階	447,204,459	83,226,055	-
③南中階	938,142,471	167,651,908	-
①～③統合後	-	387,586,311	20,954,545



統合後の本町駅の点群データイメージ

①共通処理｜点群データの前処理：❸メッシュサーフェス化

点群のままでは経路探索に利用できないため、CloudCompareの機能を用いて点群をメッシュ化する

処理フロー

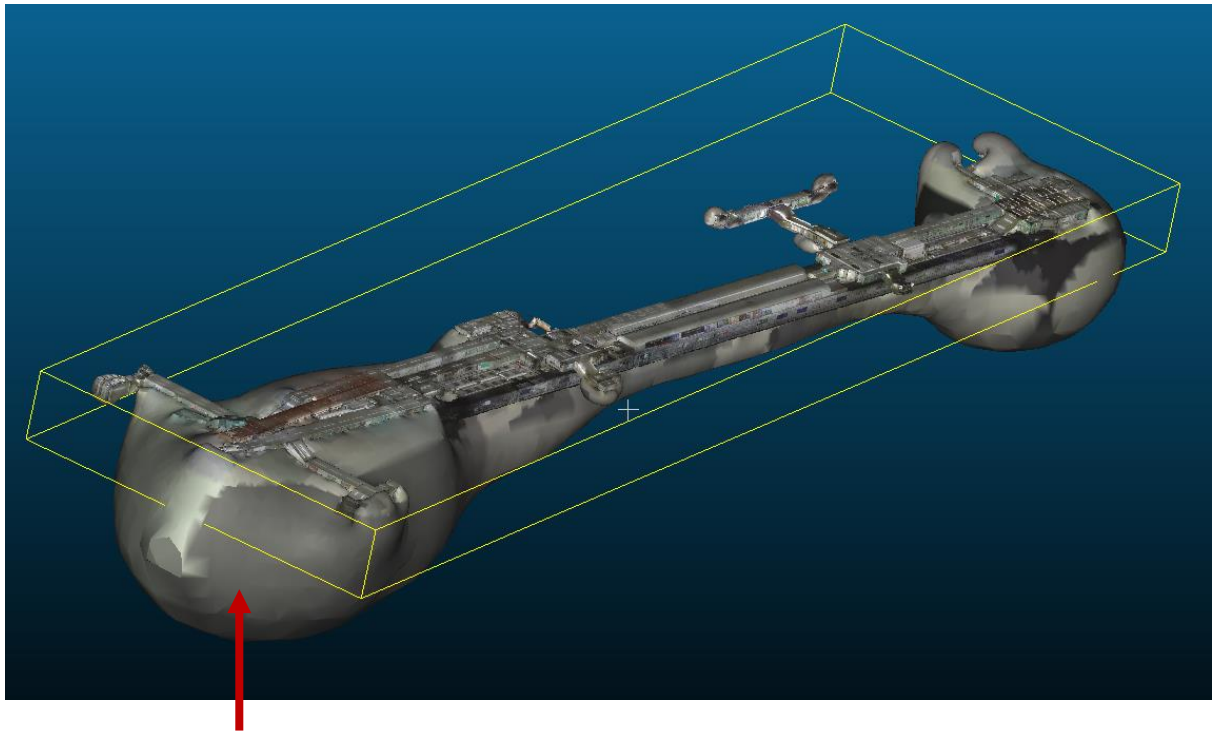
点群のメッシュ化

- Cloud Compareを利用して点群をメッシュ化する（「Poisson Recon」コマンド）
  - メッシュ化により、開口部分等は滑らかな面で繋がれるために大きく円形に膨らんだ面が生成される

不要面の削除

- メッシュ化によって生成された不要な面を分割し、削除する
  - 元の点群データに沿う形で不要面を除去する

CloudCompareで生成されたメッシュ



メッシュサーフェス化によって生成された面を削除する

①共通処理｜点群データの前処理：❹メッシュ軽量化

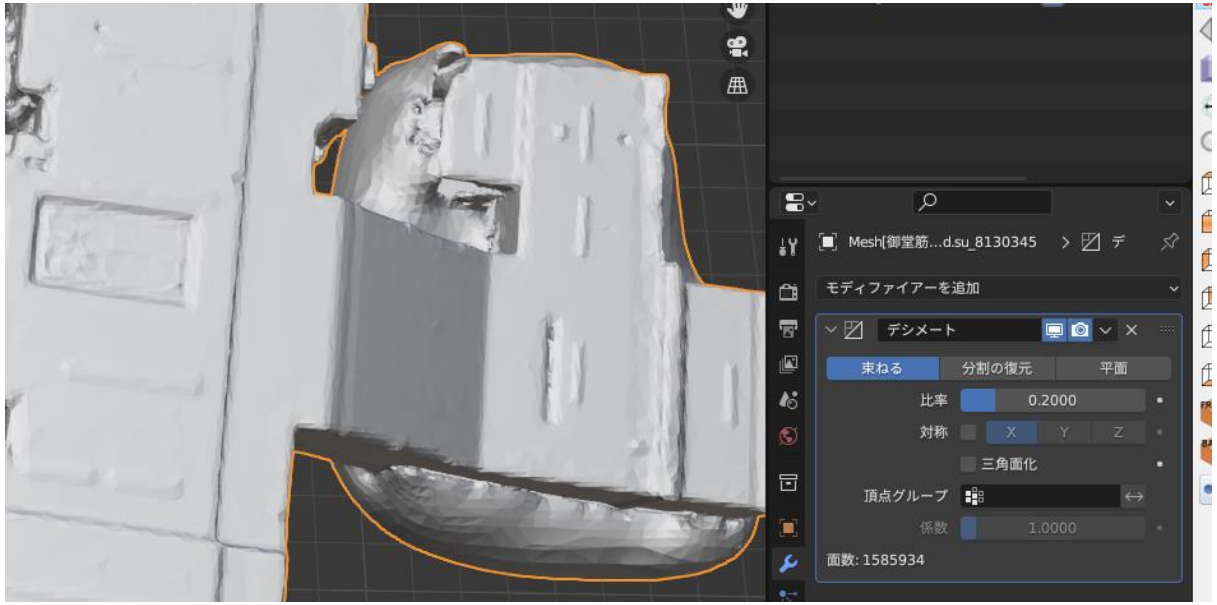
CloudCompareでメッシュ化したデータをBlenderを使ってメッシュの間引きを行う

処理フロー

メッシュの間引き

- CloudCompareでメッシュサーフェス化されたデータをBlenderのDecimateモディファイアー機能によって、メッシュの間引きを行う

Blenderによるメッシュの間引き



①共通処理｜点群データの前処理：❺メッシュサーフェス化（1/2）

CloudCompareとBlenderで実施するパターンとは別に、Metashapeを用いてメッシュサーフェスの生成とメッシュの軽量化を行うことも可能

作業フロー

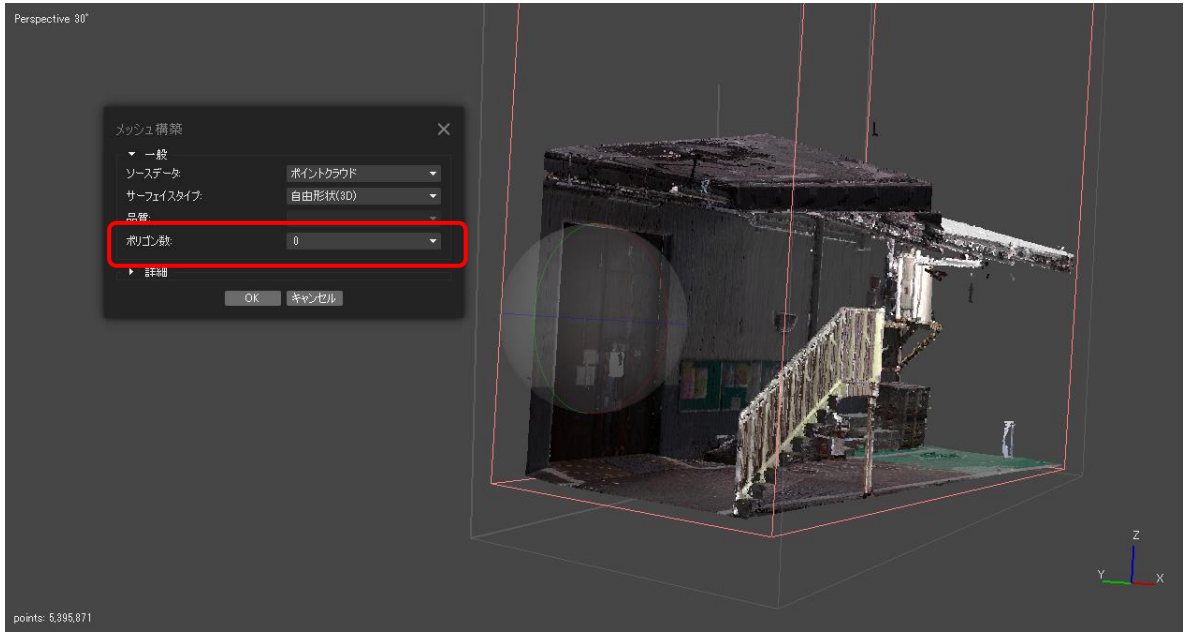
点の間隔・
メッシュ数の設定

- メッシュ化にあたって、点の間隔とメッシュ数の設定を行う
  - 点の間隔：5mm
  - ポリゴン数設定： 0
    （設定値の理由は次頁参照）

メッシュの構築

- 「領域を変形」を選択し、「領域のリセット」を行った後にメッシュを構築する

メッシュサーフェスの生成



①共通処理｜点群データの前処理：❺メッシュサーフェス化（2/2）

データサイズの削減はメッシュサーフェス化後にも実施するため、ここでは点の間隔を揃える意図でデータ削減効果の最も小さい、点の間隔を「5mm」、ポリゴン数設定を「０」とした

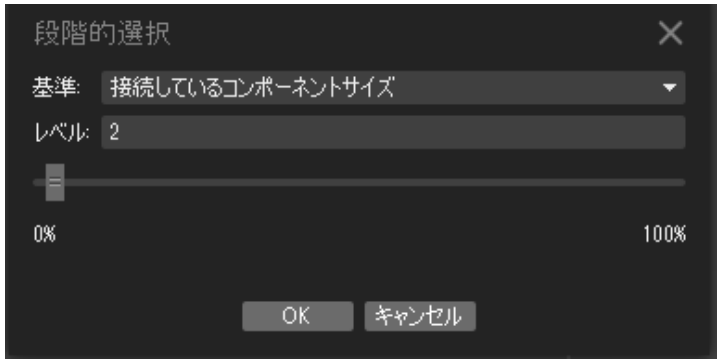
メッシュ構築*時における点の間隔とメッシュ数とデータ量の関係

点の間隔	Original				**5mm**				１０mm				３０mm			
点の数	520万点				**330万点**				105万点				14万点			
ポリゴン数の設定	0	高	中	低	**0**	高	中	低	0	高	中	低	0	高	中	低
メッシュ数	770万	71万	28万	9.6万	**456万**	45万	18万	6万	137万	16.5万	7.2万	2.5万	17.5万	2.1万	0.9万	0.3万
データサイズ(fbx)	132 MB	13.5 MB	5.5 MB	1.9 MB	**79.6 MB**	8.5 MB	3.5 MB	1.2 MB	24.3 MB	3.2 MB	1.4 MB	0.5 MB	3.2 MB	0.5 MB	0.2 MB	0.01 MB

*点群データの一部をトリミングし、メッシュ構築の設定を変えて検証

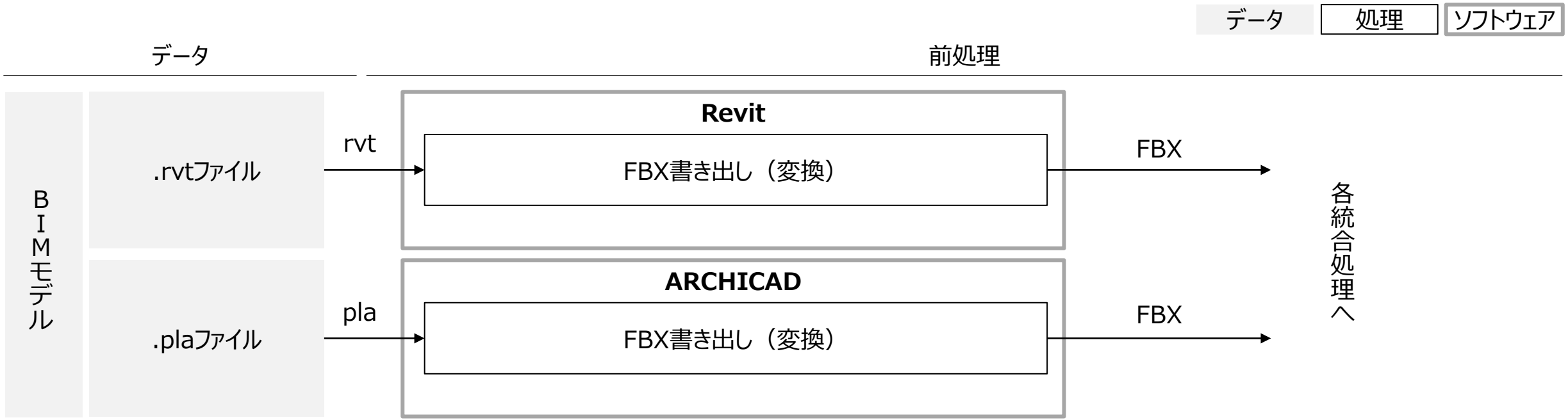
①共通処理｜点群データの前処理：❻メッシュ軽量化

CloudCompareとBlenderで実施するパターンとは別に、Metashapeを用いてメッシュサーフェスの生成とメッシュの軽量化を行うことも可能

作業フロー		設定画面
浮遊ノイズ除去	• メッシュサーフェス生成時に生じる浮遊ノイズを除去する - 浮遊ノイズが除去されるようスライダを使って調整する	
メッシュのリダクション	• リダクション処理によってメッシュ数を調整する	

①共通処理｜BIMモデルの前処理

BIMモデルの前処理はFBXへの変換のみとなるため、RevitやARCHICADなどFBX書き出しに対応したBIMソフトウェアを入力するBIMモデルに合わせて選択する



②統合手法1｜概要

原点と参照点を一つずつ設定して位置合わせを行い統合を行う手法で最も単純なため工数が極小化されるが、参照点から離れる程誤差が拡大する

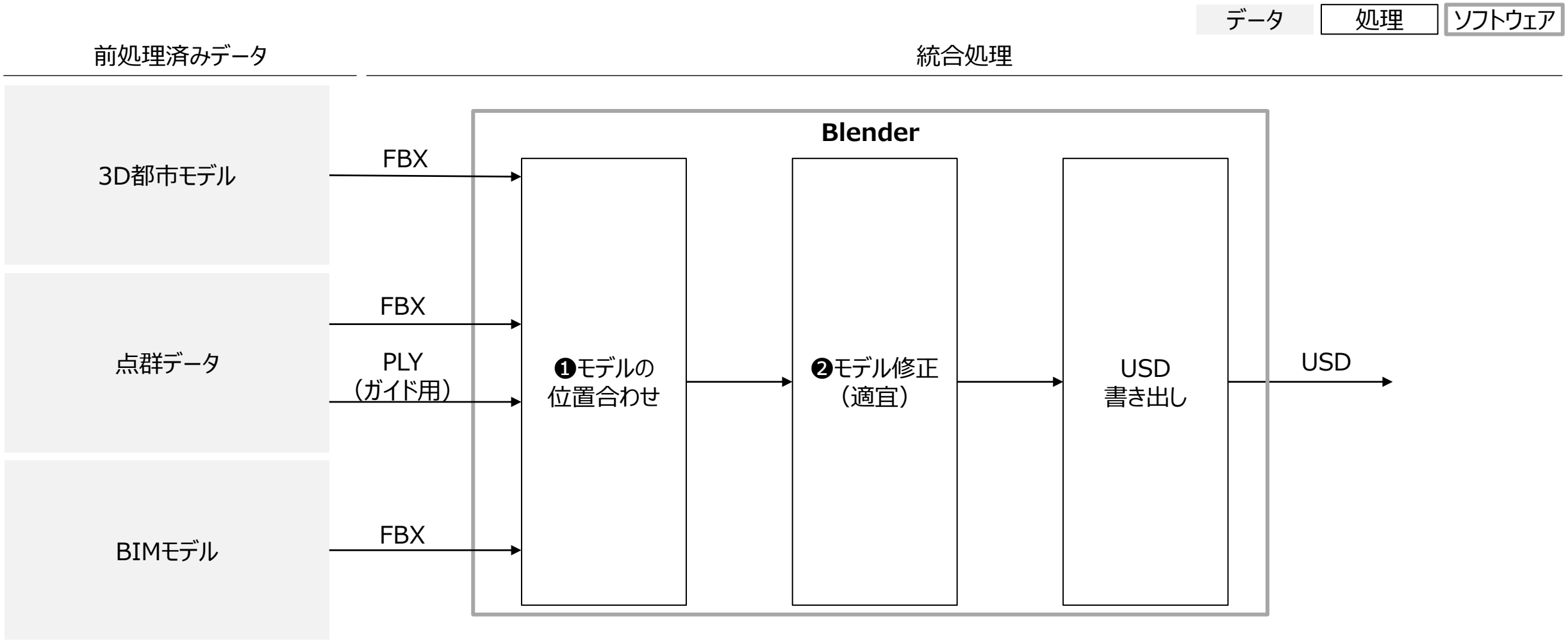
特徴

統合手法	• 原点と参照点を一つずつ設置して統合を行う、もっとも単純な統合手法 • 位置合わせは参照点一か所のみで実施するため、参照点付近のずれは少ないが、参照点から離れるほどズレが大きくなる
特徴	• 少ないモデルの統合で、求められる統合の許容誤差が大きい場合には最も手軽な手法 • 目視、手作業に頼る割合が大きく、作業する人によって結果が異なる

イメージ

②統合手法1｜フロー

Blenderを使い、各データを1組の原点と参照点をもとに回転と移動だけで位置合わせを行って統合する



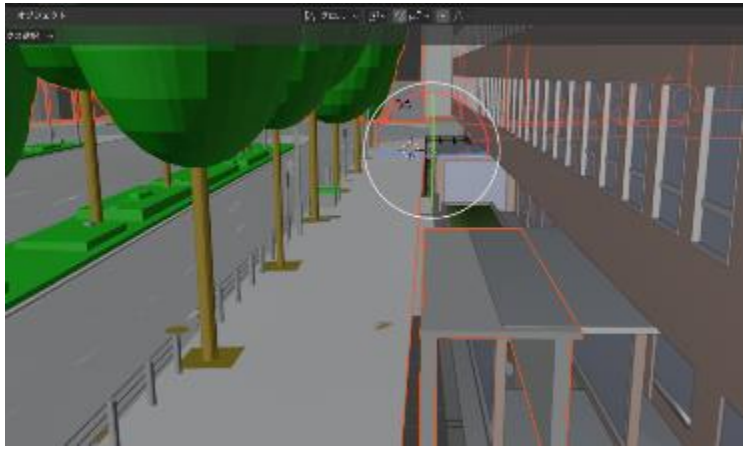
②統合手法1｜❶モデルの位置合わせ

BlenderにてBIM、3D都市モデル、点群を開き、目視で位置合わせを行う（移動、回転）

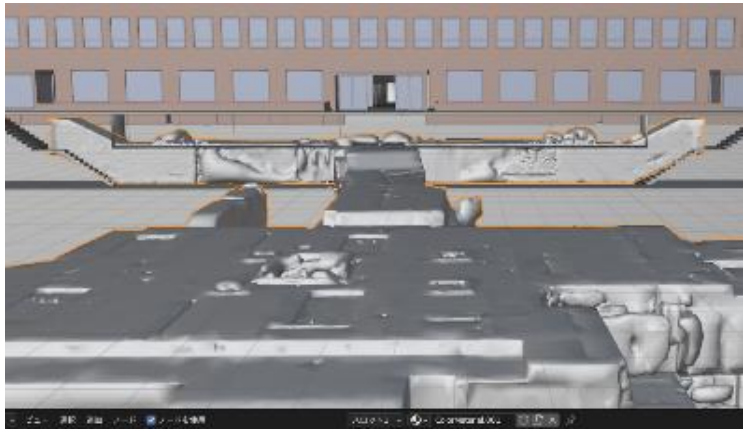
モデルの位置合わせの方法

- 地下鉄入口等、モデル同士の接続部で目印となる地物や形の取りやすい箇所を2点設定し、それぞれ原点、参照点とする
- 原点を重ねるようにモデルを配置する
- 原点を回転中心としてモデルを回転させ、参照点を合わせる
- なお、いずれも目視で位置合わせを行い、各データ間の寸法精度はある程度の範囲に収まっているものとして、各データの拡大縮小は行わない

イメージ



地下鉄入口を原点に設定



地下鉄階段を参照点に設定

②統合手法1｜❶モデルの位置合わせ：点群とメッシュサーフェスの配置

ガイド用の点群データを基準として点群から生成されたメッシュサーフェスを配置する

作業フロー

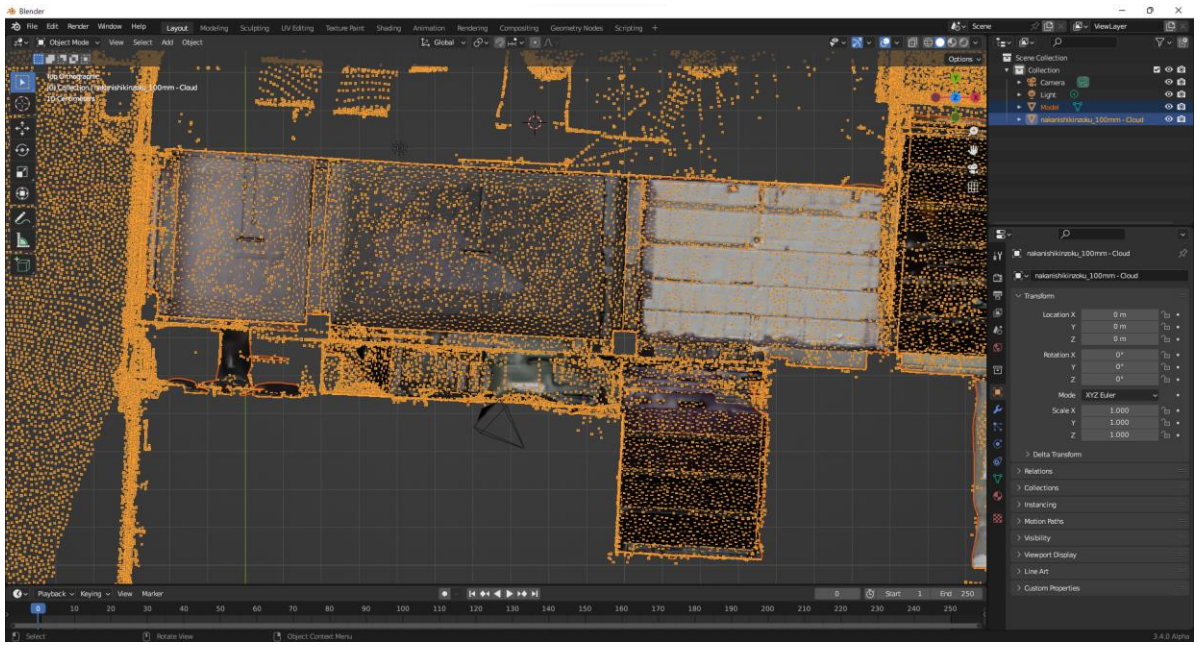
点群とメッシュサーフェスの読み込み

- Blenderでガイド用の点群データと点群から生成されたメッシュサーフェスを読み込む

配置の確認

- 点群データとメッシュサーフェスの位置が重なっていることを目視で確認する

点群読み込みイメージ



オレンジ：点群データ

②統合手法1｜❶モデルの位置合わせ：点群と3D都市モデルの配置

Blenderを用いて点群と3D都市モデルの位置合わせを行う

作業フロー

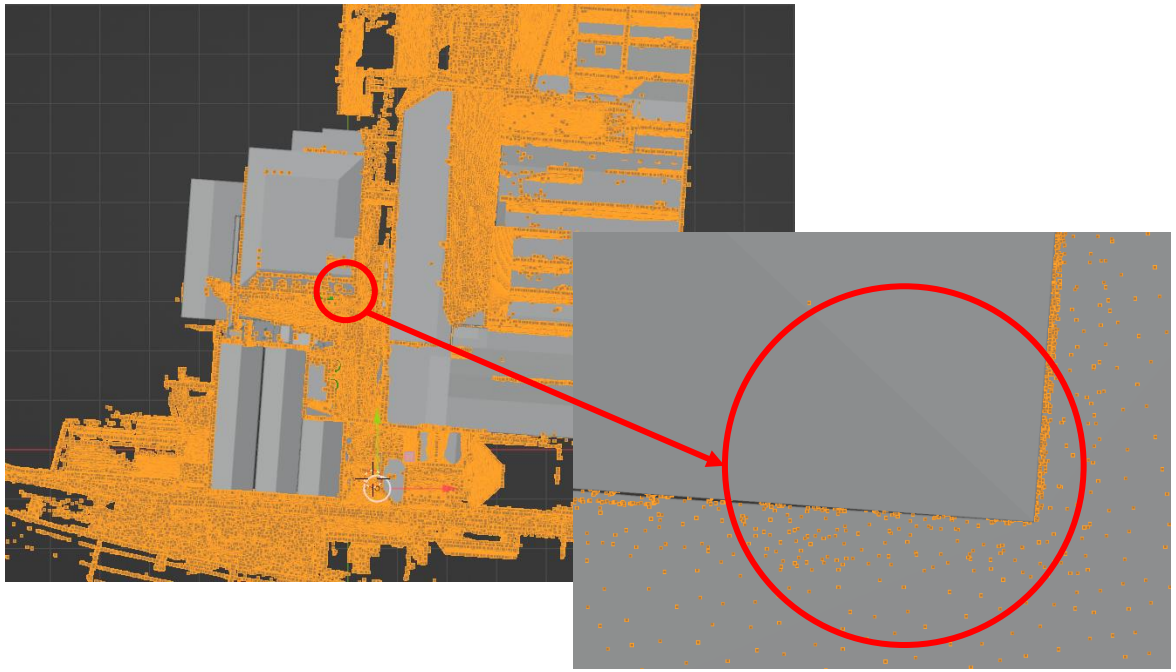
3D都市モデルの読み込み

- 3D都市モデルをBlenderに読み込むと座標系の違いから、位置及び向きが異なって配置されるため、ガイド用点群のあるエリアにモデルを移動する

位置合わせ

- 設定した原点と参照点を使い、ガイド用の点群と目視で位置合わせを行う

点群と3D都市モデルの重なりのイメージ



赤枠：点群が3D都市モデルと離れている状態

②統合手法1｜❶モデルの位置合わせ：点群とBIMモデルの配置

Blenderを用いてBIMモデルの位置合わせを行う

作業フロー

BIMモデルの読み込み

- BIMモデルを読み込む

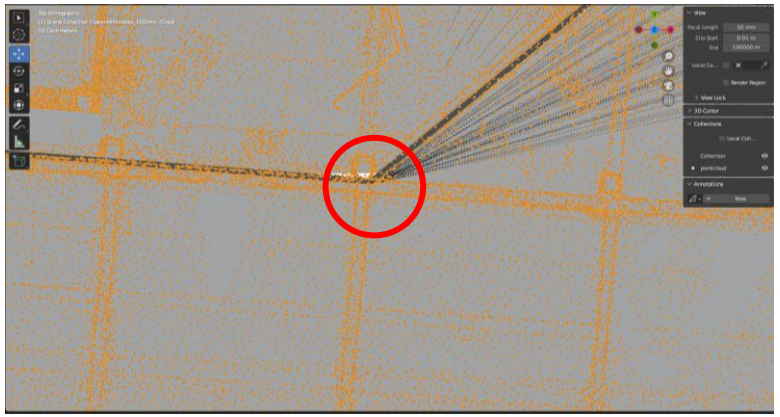
位置合わせ

- 設定した原点と参照点を使い、ガイド用の点群と目視で位置合わせを行う

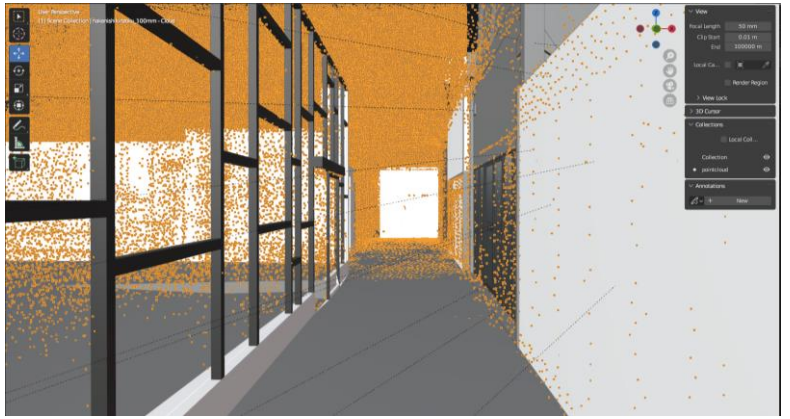
アイレベルでの確認

- BIMモデルに対し、点群は現地の不陸などが反映されるため完全に一致することはないため、アイレベルで確認を行う

位置合わせイメージ



赤枠：
基準点に設定する箇所



アイレベルでの確認

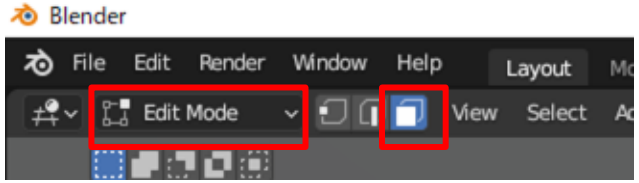
②統合手法1｜❷モデルの編集

3D都市モデルは建物内外の接続箇所にもメッシュが存在するため、不要なメッシュを削除する

作業フロー

不要なメッシュの削除

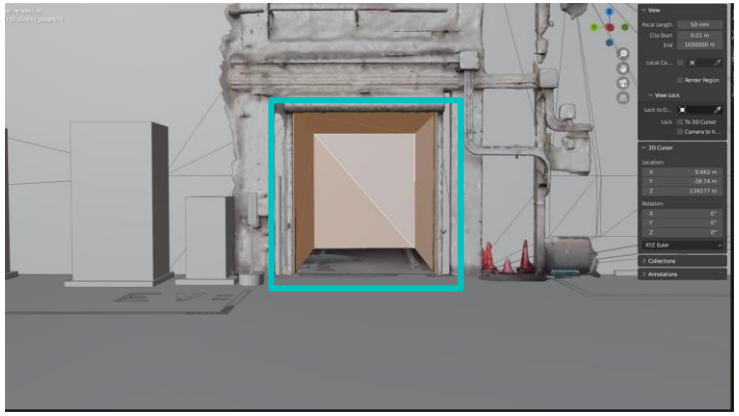
- BlenderをEditモードに変更後、Faceモードにし不要な面を選択する
- DeleteからFaceを選択すると、選択した面のみ削除される



建物内外の接続

- 不要なメッシュを削除したことで建物の内外が接続される

メッシュ情報編集前後



青枠：接続箇所にある不要なメッシュ

建物内外が接続される

③統合手法2｜概要

原点を一つ設定し、複数の参照点で誤差が小さくなるように位置合わせを行いながら統合を行うため、作業工数が増加する一方でズレの大きさは3Dモデル全体で平均化される

特徴

統合手法

- 原点一つに対して、複数の参照点を設置して参照点での誤差が小さくなるように合わせる
- それぞれの参照点でズレを小さくするように合わせるため、ズレの大きさは各所で平均的になる

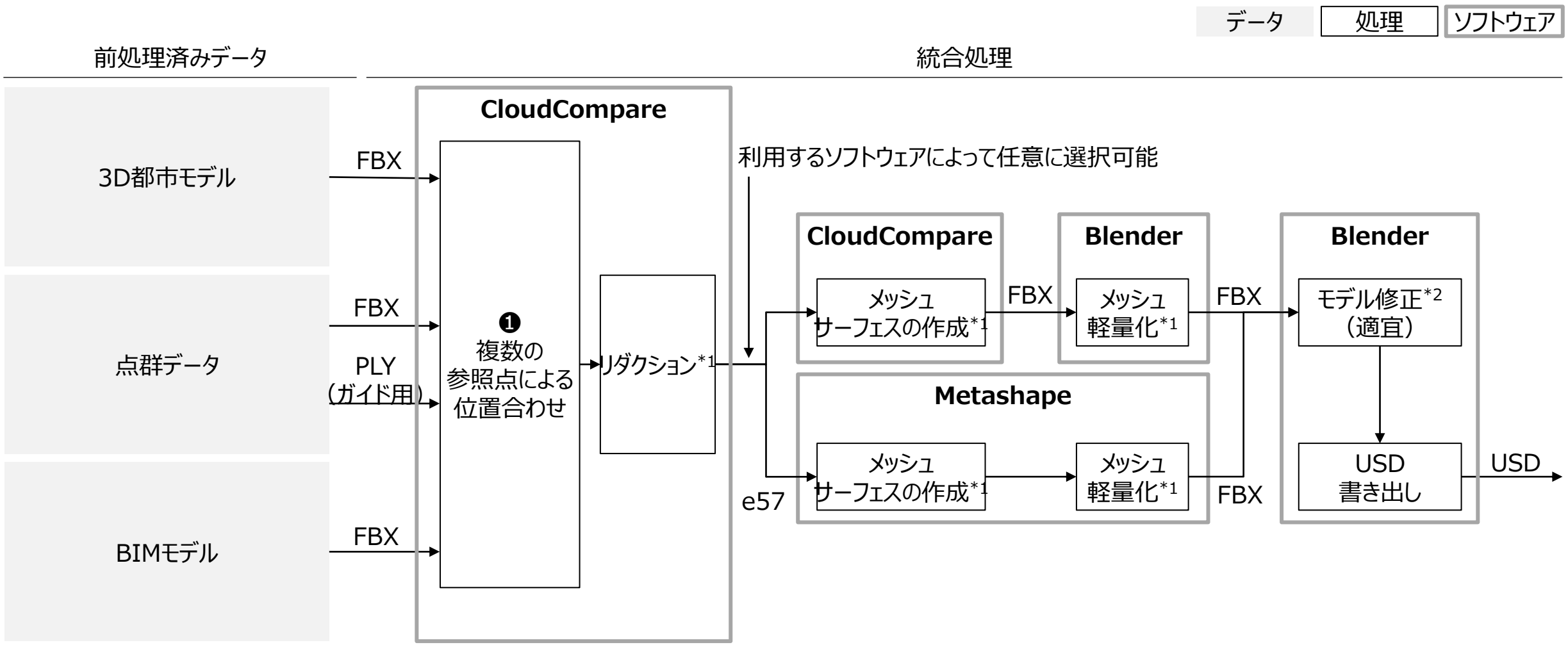
特徴

- 複数の参照点の誤差が小さくなるように合わせるため、ズレの大きさは各所で平均的になる
- モデル間の境界面に形状誤差がある場合、境界が不連続となる
- 移動・回転の処理は自動的に行われる

イメージ

③統合手法2｜フロー

Cloudcompareの機能を用いて複数点の平均から位置合わせを行う



*1 点群データの前処理の同名称の処理と同様
*2 統合手法1の同名称の処理と同様

③統合手法2｜❶複数の参照点による位置合わせ

CloudCompareの「Aligns two clouds by picking」機能*を用いて位置合わせを行う

作業フロー

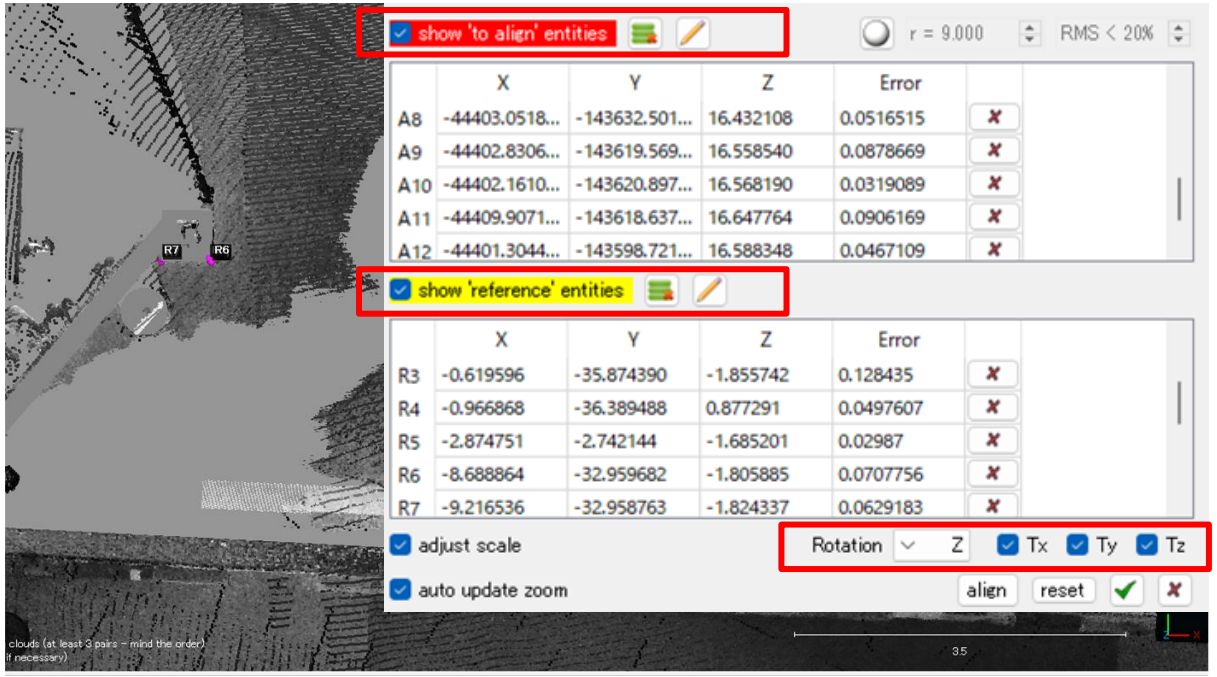
モデルの配置

- 3D都市モデル、点群データ、BIMモデルを読み込む
- 位置を合わせるモデルを基準となるモデルの近くに配置する
  - BIMモデルを基準とする

位置合わせ

- 対象モデルをすべて選択し「Aligns two clouds by picking」を選択し、以下を設定して位置合わせを行う
  - 位置を合わせるモデル
    - 「to align entities」に設定
  - 基準となるモデル
    - 「reference entities」に設定
  - Rotationは「Z」とし、水平面の回転のみにする

位置合わせの設定イメージ



*モデル同士の対象点を複数指示し、対象点同士の距離が最小となるようにモデルを移動し、位置合わせを行うコマンド

Ⅲ. 実証システム ＞ 5. データ ＞ ②データ処理

④統合手法3｜概要

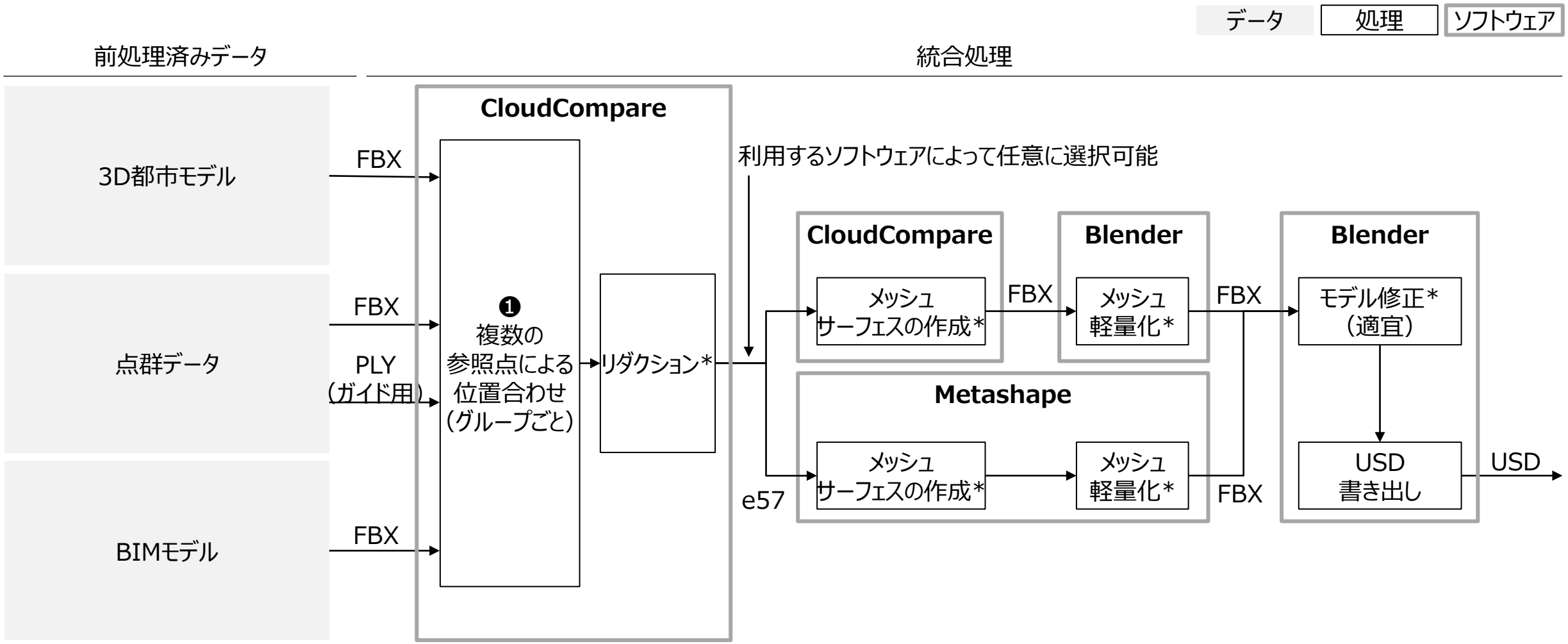
点群データをエリアごとに分割し、それぞれ原点を設定の上一つの参照点を利用し統合する手法で、複数の原点と参照点を調整して誤差を極小化させられる一方で工数は増加する

	特徴
統合手法	• 各モデルごとに原点を設置する • 原点は出入口など精度が求められる箇所を内包したエリアごとにグループ化し、エリアごとの参照点で調整する • グループ化したデータ毎に重なりが微妙にズレることが想定されるが、調整代とみなす
特徴	• 複数の参照点で調整を行うため、時間や工数の手間が増加する • モデル間での重なり合った部分は調整代とみなして統合することで連続的なモデルを構築できる

イメージ

④統合手法3｜フロー

3D都市モデルと点群データのズレが大きい場合を想定し、CloudCompareを使って点群を接続箇所ごとに分割・グループ化し、グループごとに微調整を行う



*点群データの前処理の同名称の処理と同様

④統合手法3｜❶複数の参照点による位置合わせ（グループごと）

CloudCompareにて点群データを読み込み、3D都市モデルとの位置合わせを行う

作業フロー

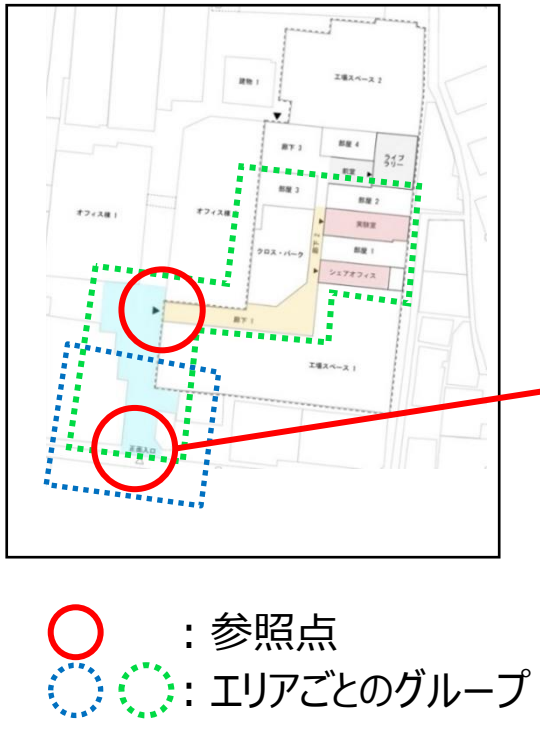
データの読み込み

- 基準としたい地物や形状を含むエリアごとに、原点と参照点を設定する
- エリアごとにグループ化するため、点群を分割するためにメッシュサーフェスの必要な箇所をトリミングする

位置合わせ

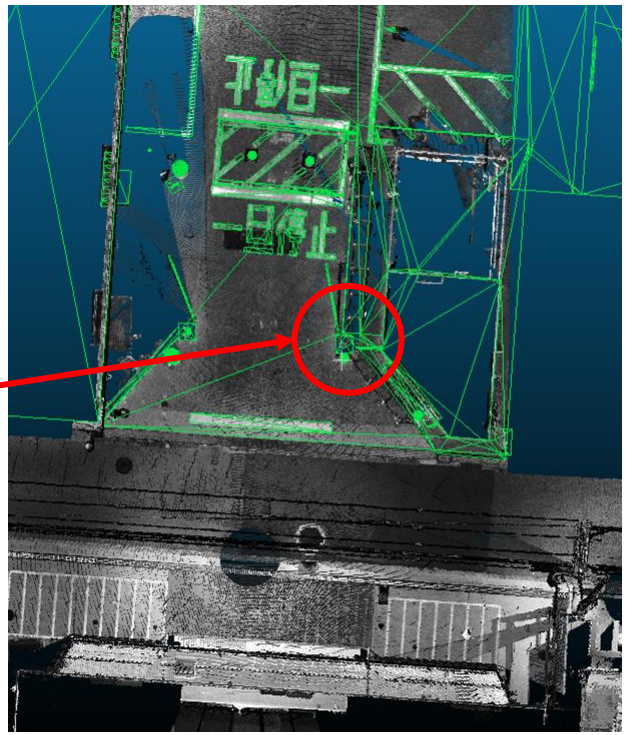
- 分割した点群と3D都市モデルをグループごとに位置合わせする

イメージ



：参照点

：エリアごとのグループ



⑤統合手法4｜概要

手法2を実施した後にモデル間で接続面が非連続になっている個所は局所変形を行い、モデル同士を滑らかに接続する

特徴

統合手法

- 統合手法は手法2と同様
  - 一つの原点に対して参照点を複数設置し、参照点ごとのズレが少なくなるように調整する
- 手法2のままではモデル間の接合面がつながらないため、モーフィングを行うことで複数モデルを統合する
  - モーフィングとは形の異なる接合面のものを自然になるようにつなぎ合わせる技術

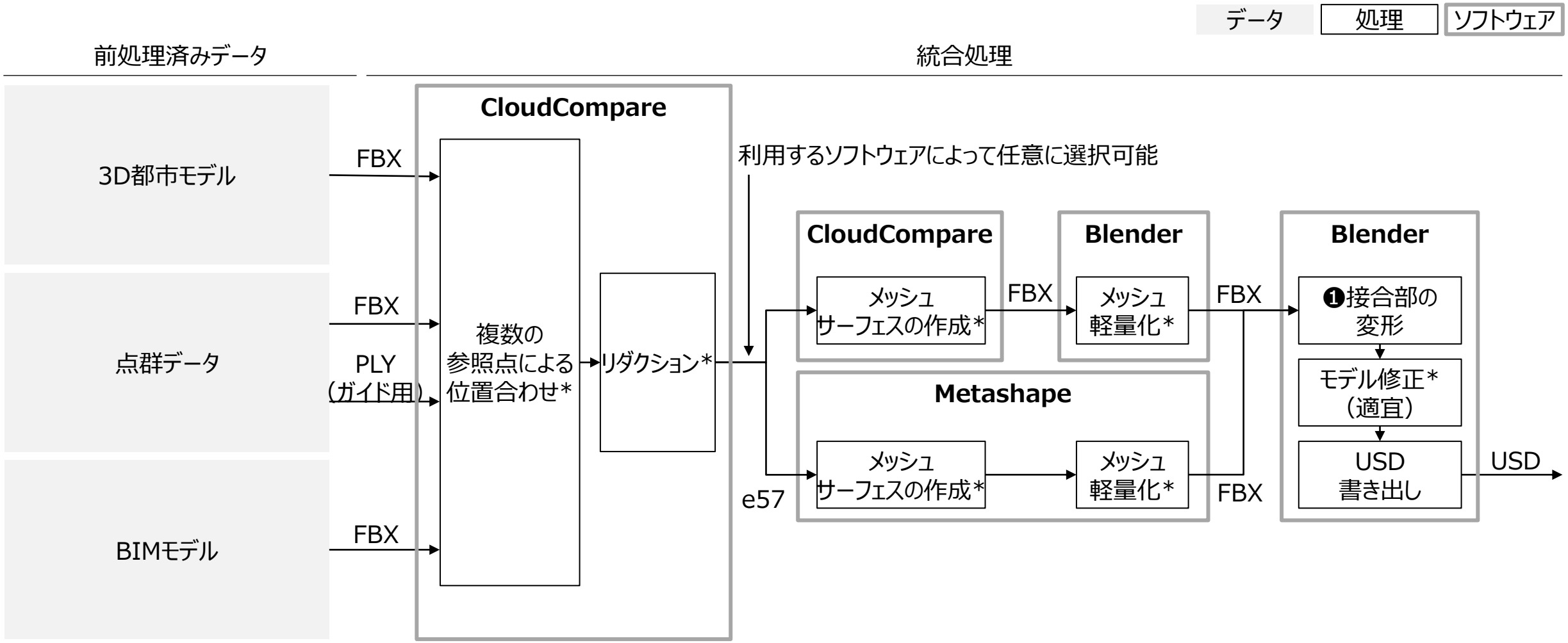
特徴

- 各モデルごとに原点を設置する
- 原点は出入口など精度が求められる箇所を内包したエリアごとにグループ化し、エリアごとの参照点で調整する
- グループ化したデータ毎に重なりが微妙にズレることが想定されるが、調整代とみなす

イメージ

⑤統合手法4｜フロー

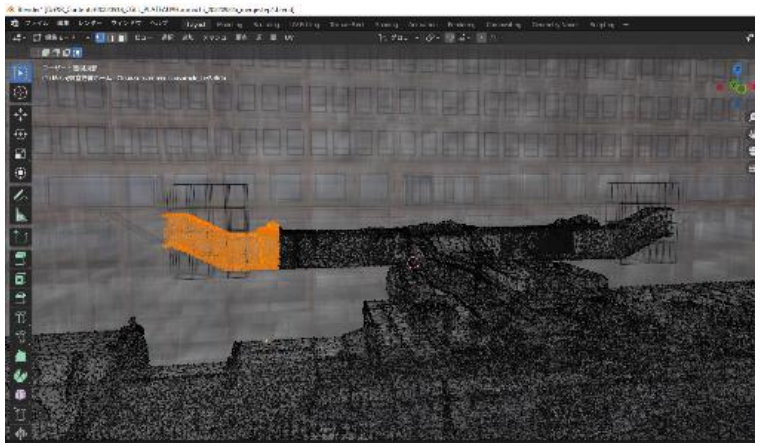
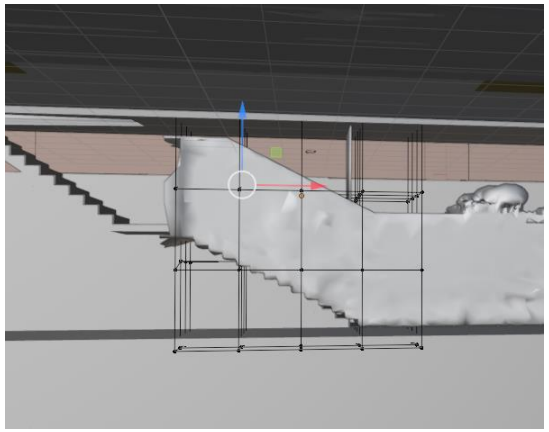
Cloudcompareの機能を用いて複数点の平均から位置合わせを行ったのち、Blenderを使ってモデル変形により接合面の滑らかさを確保する



*点群データの前処理の同名称の処理と同様

⑤統合手法4｜❶接合部の変形

モデル側の不整合等により、移動、回転のみによる位置合わせでは対応できない部分については、「ラティスモディファイアー」機能によって変形を行う

データ接続のためのデータ変形		イメージ
変形対象を選ぶ	• 局所的にズレが発生している箇所において、変形が必要な個所を選択する	 変形対象の選択
モデル間を統合する	• Blenderの「ラティスモディファイアー」機能を使い変形を伴うモデル統合を実施する	 BIMデータの地下鉄入口の階段の段数が実際と異なっていたので、変形によって繋ぎ込みを実施

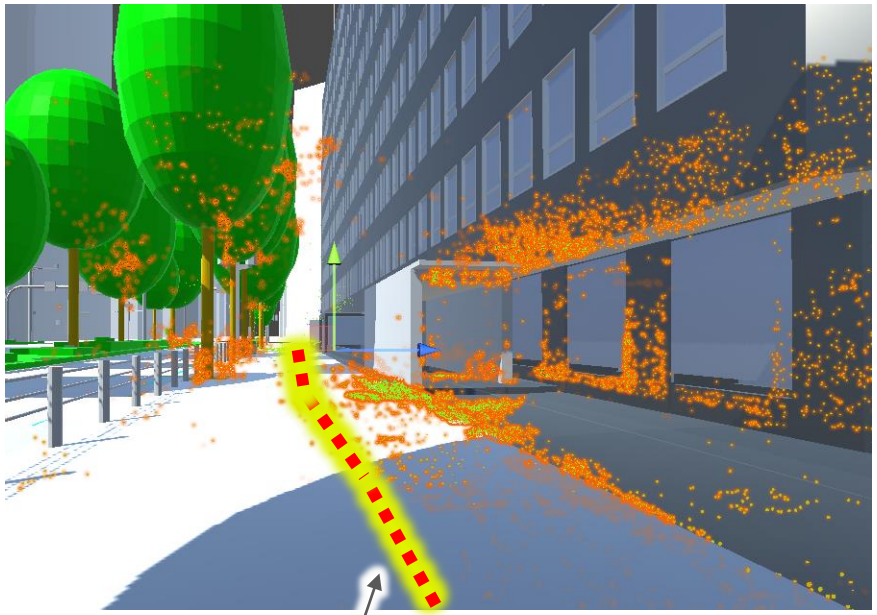
経路情報（Unity内部データ）の生成

ARナビで案内する経路情報を生成する

概要

項目	詳細
概要	統合3Dデータ、現在位置座標をもとに、Unityの経路探索機能を用いて、最短経路を生成
データ仕様	経路情報（Unity内部データ形式）
備考	ARナビの経路として、UIの画面表示に使用する

イメージ



経路を算出

自己位置座標（Immersal内部データ）の生成

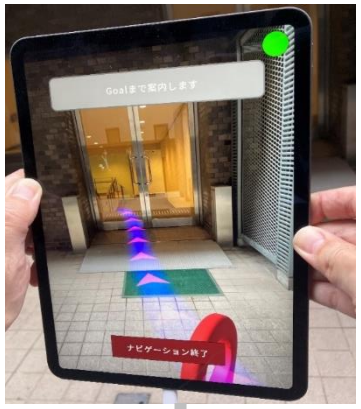
Immersalを利用し、自己位置を推定する

概要

項目	詳細
概要	ARデバイスの映像から、VPSの自己位置推定機能を用いて自己位置座標を生成
データ仕様	自己位置座標（Immersal内部データ形式）
備考	ARナビの推定現在位置として、経路探索・ARナビ表示に使用

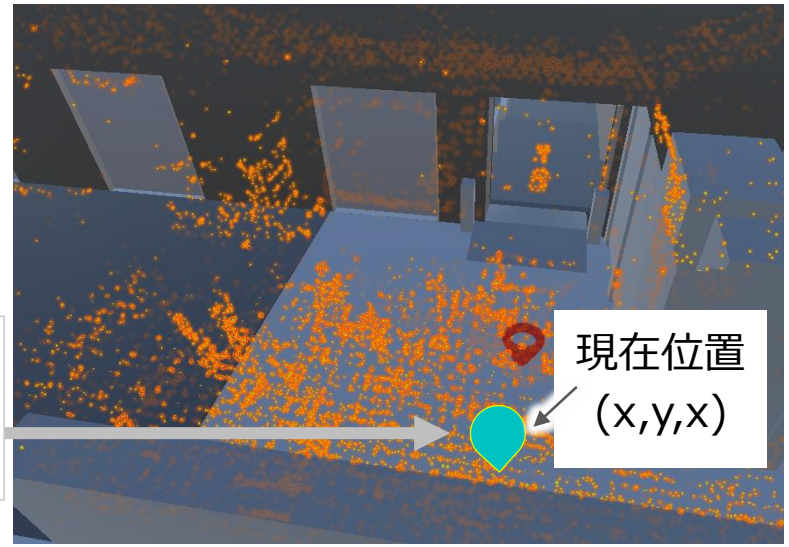
イメージ

ARデバイス



ARデバイスの映像とVPS用点群データを突き合わせ現在位置を推定

VPS用点群データ(オレンジ)



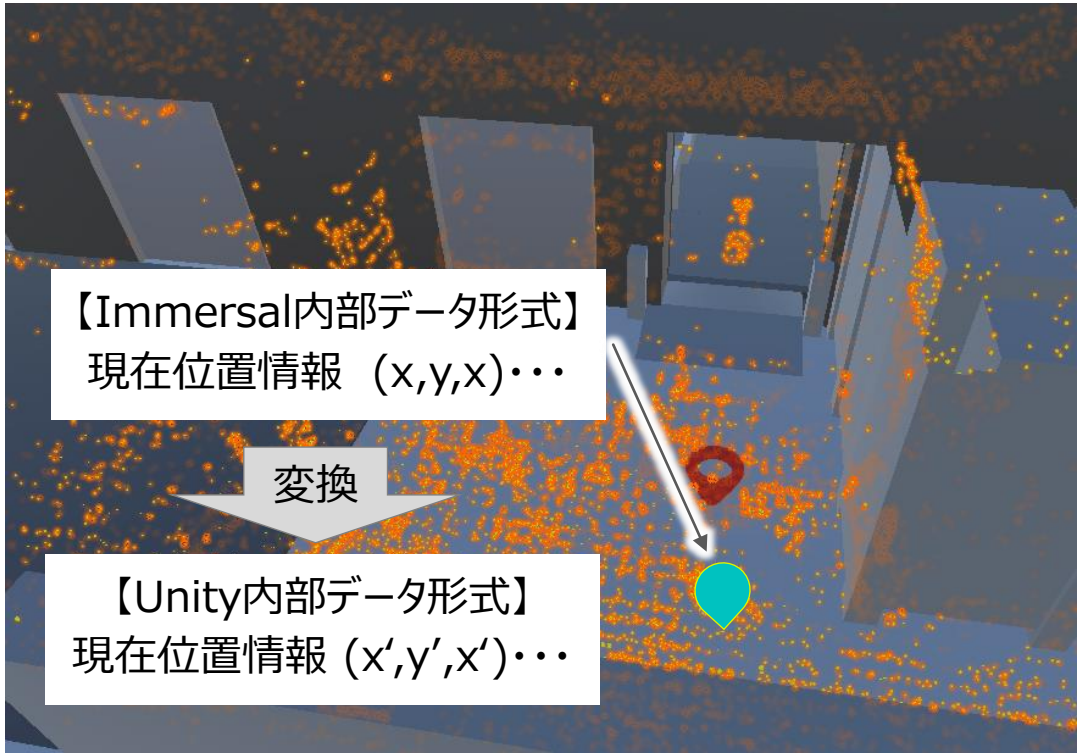
現在位置座標（Unity内部データ）の生成

Immersalで取得した自己位置座標をUnity処理用に変換する

概要

項目	詳細
概要	VPSから取得した自己位置座標をCommon Ground PF側の座標に変換
データ仕様	現在位置座標（Unity内部データ形式）
備考	ARナビの推定現在位置として、経路探索・・ARナビ表示に使用

イメージ



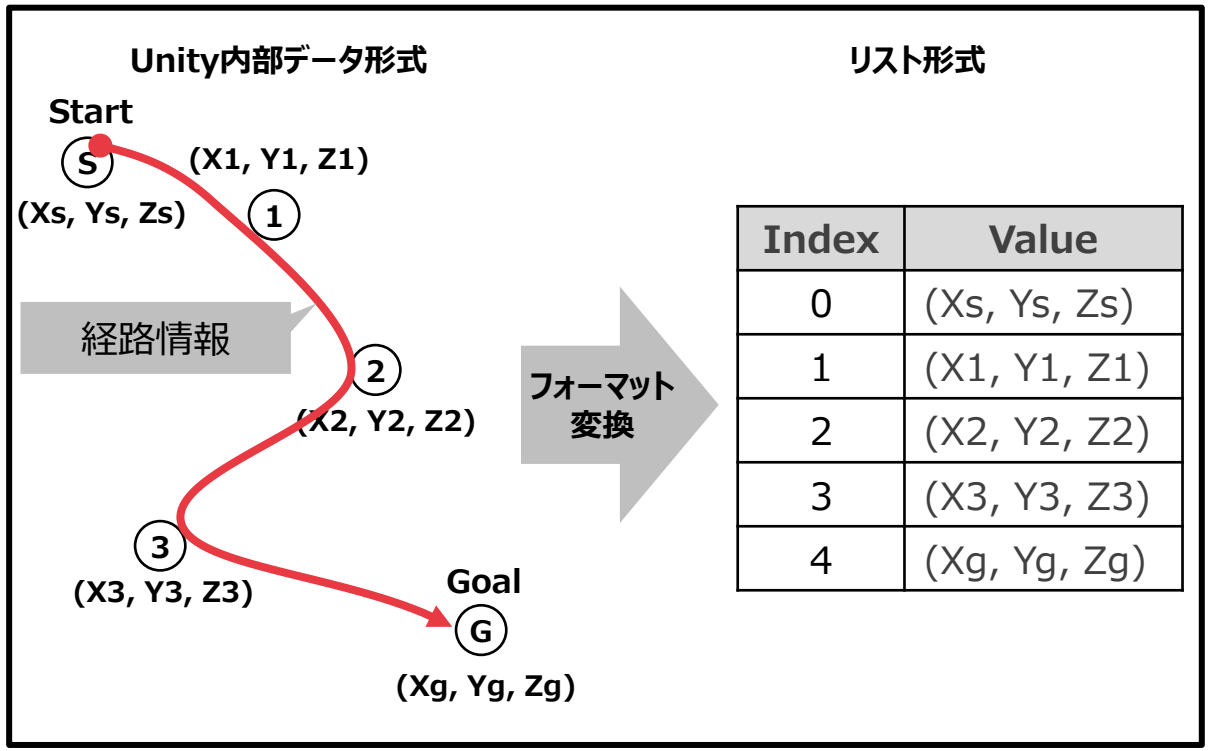
経路情報（リスト形式）の生成

経路探索機能で生成した経路情報をARナビ処理用に変換する

概要

項目	詳細
概要	Common Ground PF側の経路情報をARナビの処理用にリスト形式に変換
データ仕様	経路情報（リスト形式）
備考	ARナビの経路として、UIの画面表示に使用

イメージ



ARルートの生成

経路探索機能で生成した経路情報をARナビ可視化用のオブジェクトに変換する

概要

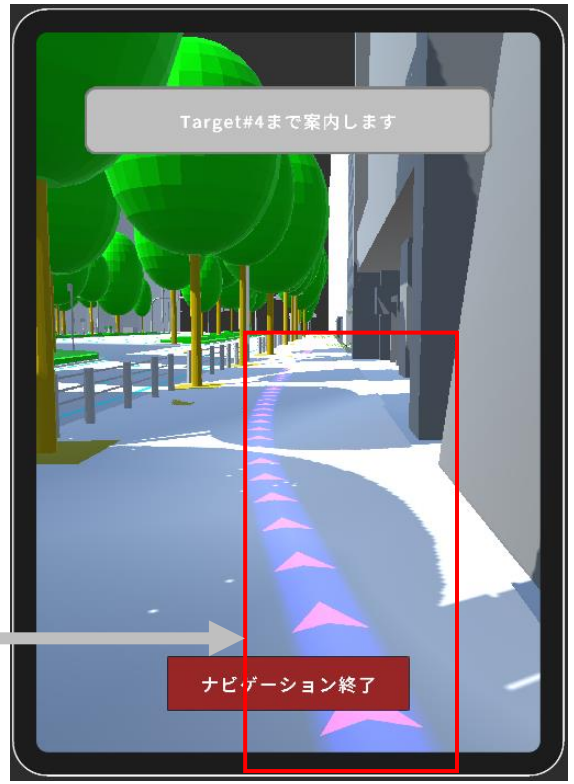
項目	詳細
概要	経路情報（リスト形式）をもとに、ARで可視化するルートオブジェクトに変換
データ仕様	ARナビ用ルートオブジェクト
備考	ARナビの経路として、UIの画面表示に使用

イメージ

リスト形式

Index	Value
0	(Xs, Ys, Zs)
1	(X1, Y1, Z1)
2	(X2, Y2, Z2)
3	(X3, Y3, Z3)
4	(Xg, Yg, Zg)

ARナビで案内用に表示されるオブジェクトに変換



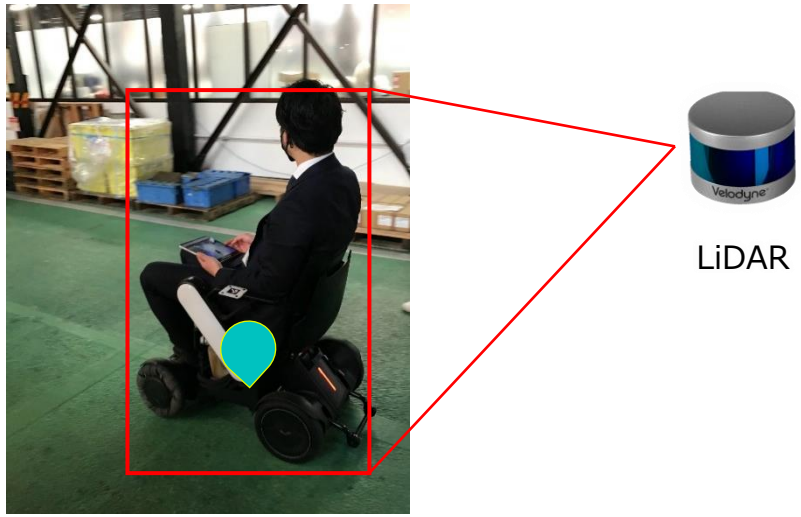
モビリティ位置情報（JSON形式）の生成

LiDARで計測された位置情報を取得

概要

項目	詳細
概要	LiDARで計測されたモビリティ位置情報をJSON形式で取得
データ仕様	JSON形式の平面座標
備考	モビリティの推定現在位置として、画面表示・経路探索に使用

イメージ




```
{
   “Time”: "2022/11/29 14:29:57.347“,
   “ID”: 160128170221387,  # 動体ID
   “X”: 40991.347656,        # 位置X
   “Y”: 4847.094727,         # 位置Y
   “Dir”: 161,                  # 方位角
}
```

WHILL位置・姿勢のJSON文字列表現

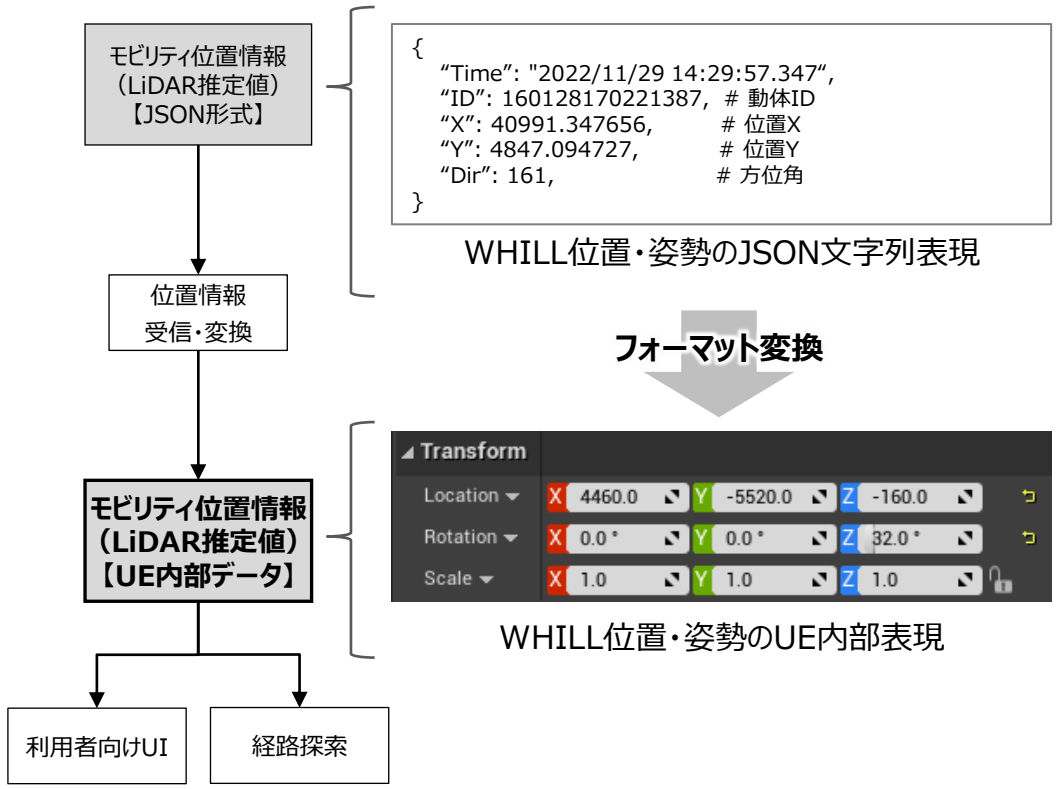
モビリティ位置情報（UE内部データ）の生成

動体検知システムから受信した位置情報を、UE内部表現へ変換

概要

項目	詳細
概要	• 動体検知システム出力のモビリティ位置情報（JSON形式）をもとに、CGPF（デジタルツイン）座標系へ変換し、UE内部で保持
データ仕様	• CGPF（デジタルツイン）座標系での位置（X, Y, Z）および回転（Pitch, Roll, Yaw）を表現する6変数をUE内部で保持
備考	• モビリティの推定現在位置として、画面表示や経路探索に使用

イメージ



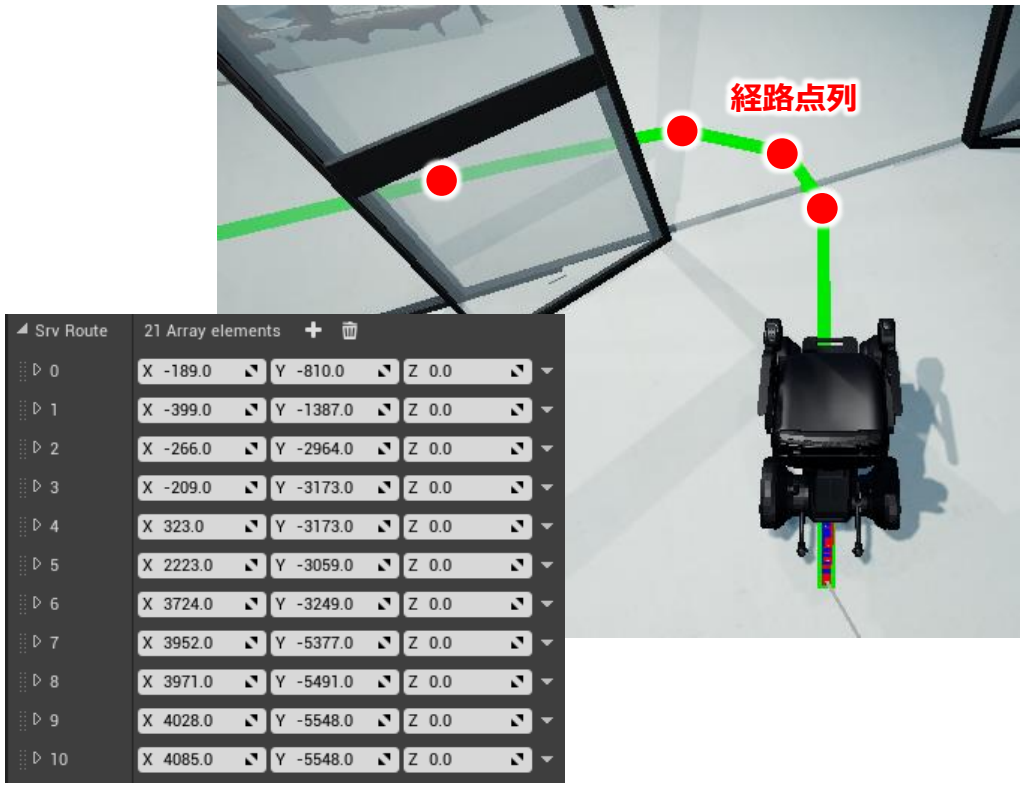
経路情報（UE内部データ）の生成

モビリティ目標経路を生成し、UE内部表現で保持

概要

項目	詳細
概要	• 統合3DデータをもとにUnrealEngineの経路探索機能を用いて、最短経路の座標点列を生成
データ仕様	• CGPF（デジタルツイン）座標系での経路点列の位置（X, Y, Z）の配列をUE内部で保持
備考	• モビリティの目標経路として、モビリティ制御に使用

イメージ



経路点列のUE内部表現

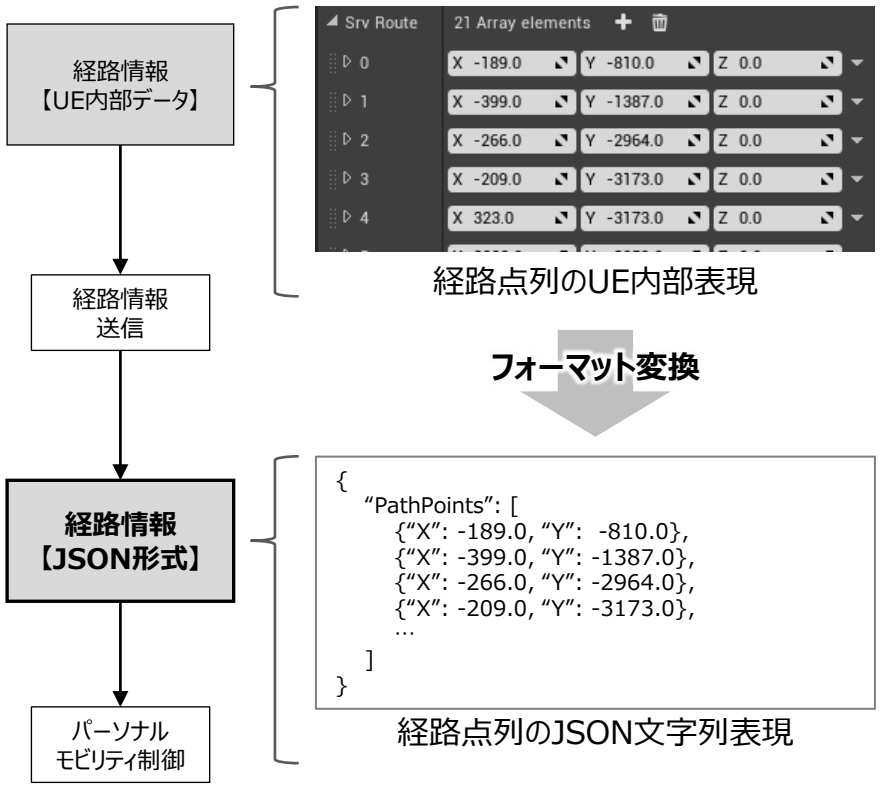
経路情報（JSON形式）の生成

モビリティへ送信・制御するため、UE内部表現の経路情報をJSON文字列表現に変換

概要

項目	詳細
概要	• UE内部表現の経路点列をJSON文字列に変換
データ仕様	• CGPF（デジタルツイン）座標系での経路点列の位置（X, Y, Z）の配列をJSON文字列で表現
備考	• モビリティの目標経路として、モビリティ制御に使用

イメージ



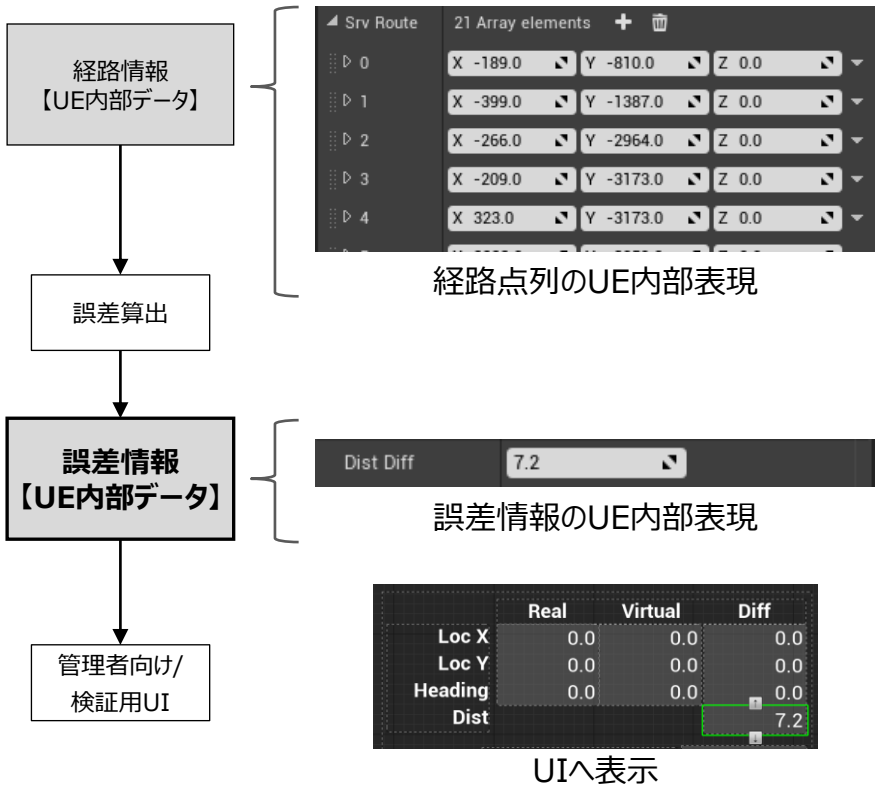
誤差情報の生成

モビリティの誤差を算出し、UE内部表現で画面表示に使用

概要

項目	詳細
概要	• 目標経路情報と、運行管理者によって入力される誤差情報をもとに算出したモビリティの横方向偏差を、UE内部表現として保持
データ仕様	• CGPF（デジタルツイン）座標系での経路点列の位置（X, Y, Z）の配列をJSON文字列で表現
備考	• 目標経路に対するモビリティの位置誤差として、検証用UIへの表示および記録に使用

イメージ



出力データ一覧

出力データ	内容	データ形式
-	-	-

ARナビアプリ｜機能一覧

ARナビアプリの画面表示および機能は以下のとおり

ナビ開始画面

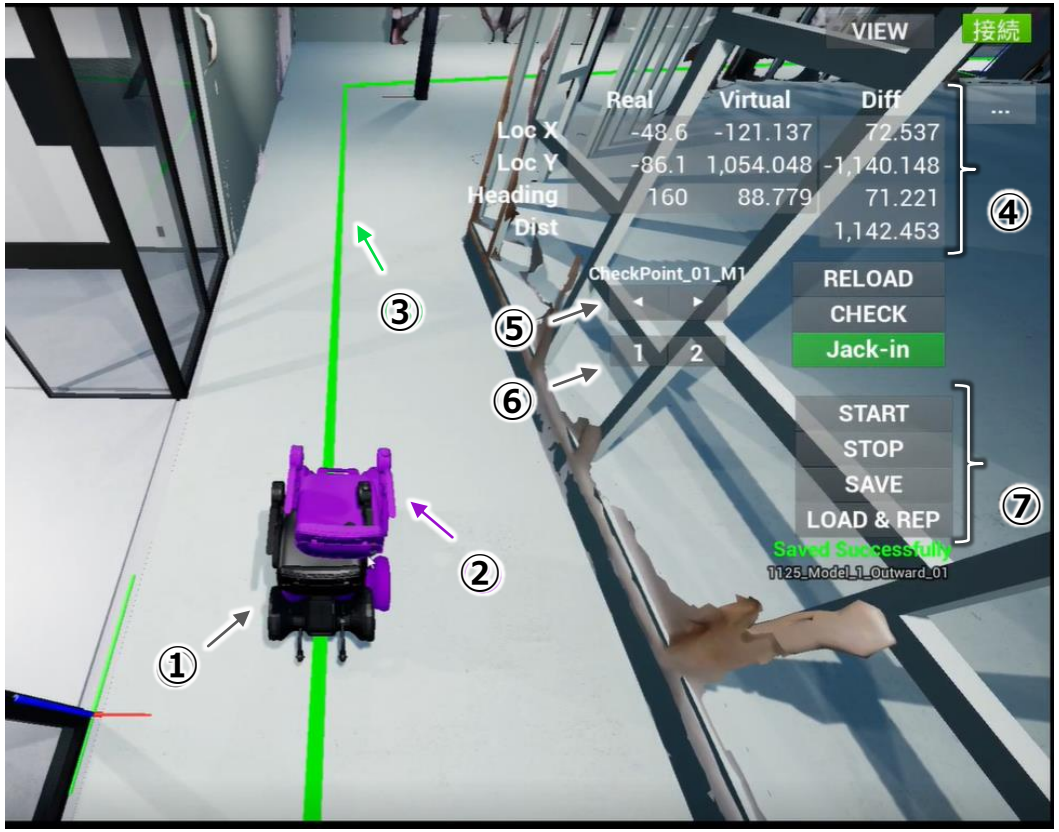
ナビ案内中画面

目的地到着画面

機能名	説明
① ナビ開始	• 以下ボタンをタップすることで案内を開始する - 中間地点ありの場合：「目的地経由⇒ゴール」 - 中間地点なしの場合：「ゴール」
② 案内メッセージを表示	• 案内中にステータスメッセージを表示する
③ 移動経路表示	• カメラ映像にルート表示を重ねて、案内表示を行う - 移動ルート：矢印 - 目的地：ピン
④ 到着メッセージを表示	• 目的地到着のメッセージを表示
⑤ ARナビの終了	• ボタンをタップすることで案内を終了する
⑥ 次の目的地へのナビ開始	• 中間地点到着後、再度案内を開始する

パーソナルモビリティ運用アプリ｜機能一覧

パーソナルモビリティ運用アプリの画面表示および機能は以下のとおり

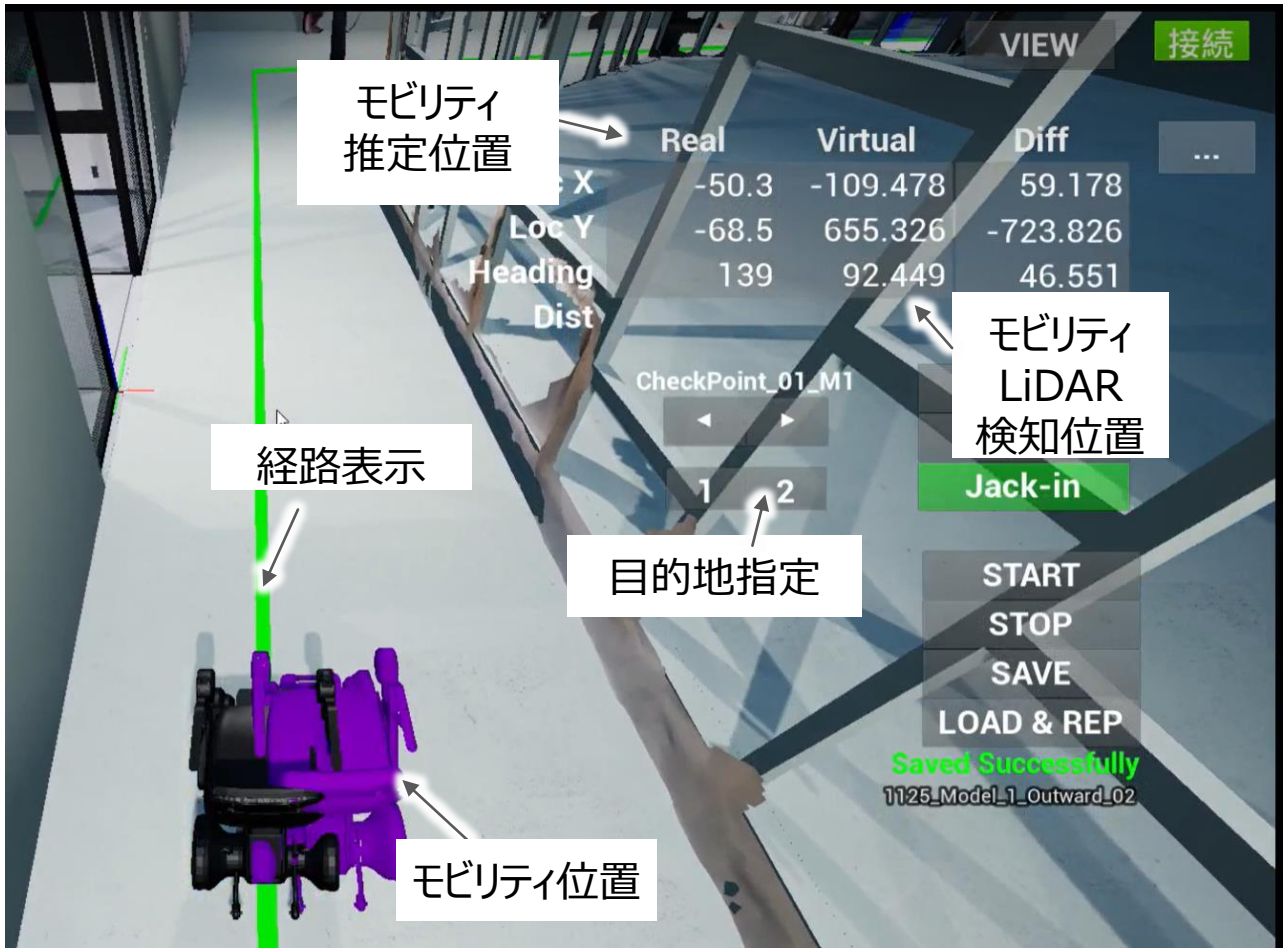


機能名	説明
① モビリティ位置表示	• モビリティのオドメトリとLiDARにより推定されたモビリティ位置を表示
② LiDAR検知情報表示	• LiDARにより検知されたモビリティ位置を表示
③ 経路表示	• モビリティ目標経路を表示
④ 誤差表示	• 誤差計測ポイントにおいて、モビリティ位置の誤差を表示
⑤ 誤差計測ポイント選択	• 誤差計測ポイントを選択肢から指定
⑥ 目的地選択	• 目的地を選択肢から指定、走行開始
⑦ ロギング	• モビリティ位置、目標経路、誤差等のログ記録を開始・終了
目的地指定（任意位置）	• 任意位置をマウスクリックすることで、目的地として指定・走行開始
走行停止・再開（キーボード操作）	• キーボード操作により、任意タイミングでモビリティ停止・走行再開能

パーソナルモビリティ運用アプリ｜目的地までの走行

目的地を設定すると経路が表示され、自己位置推定を行いながら目的地まで走行を行う

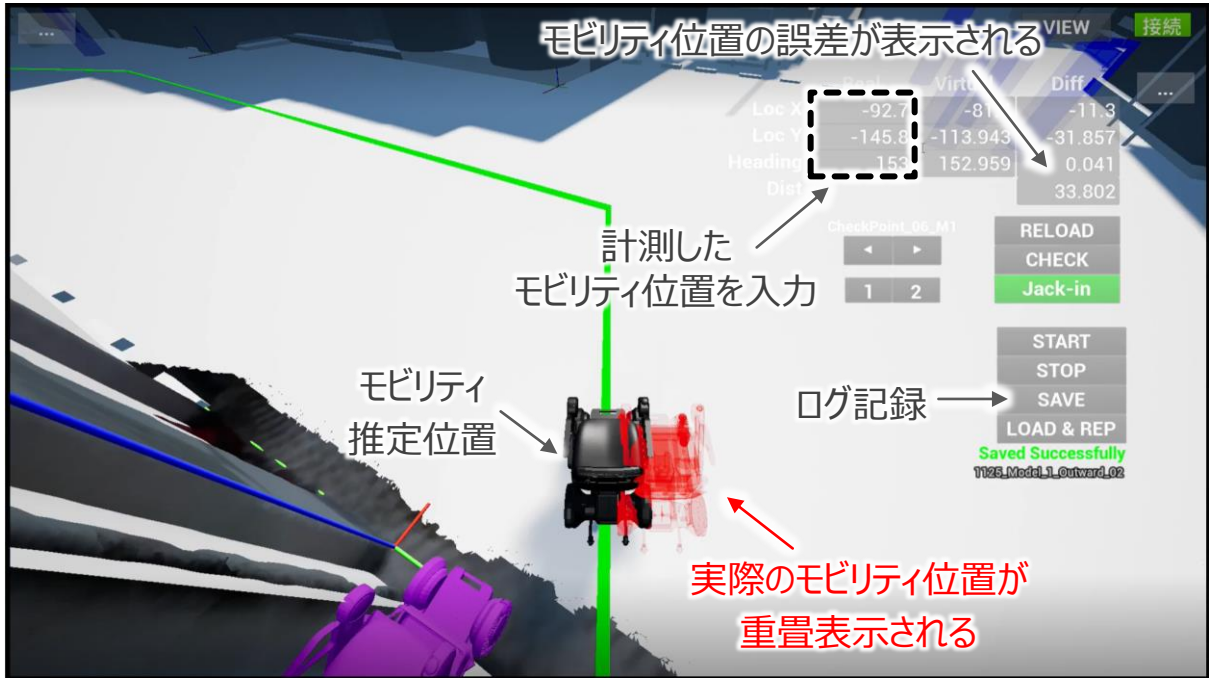
目的地を設定	• 事前に登録された候補から選択、または3Dマップ上の任意地点をクリック
目的地までの経路が表示	• 3Dマップ上に目的地までの経路が表示され、モビリティ移動開始
推定位置を表示	• 制御アプリ上での推定位置およびLiDARによる検知位置がそれぞれ表示される
誤差範囲拡大時は停止	• 走行中に推定位置と検知位置の誤差が拡大し、危険な場合等はキーボード操作でWHILLを停止させる



パーソナルモビリティ運用アプリ｜誤差の計測

誤差を計測するためのUIが備えられており、メジャー等で計測した数値を入力することで誤差の表示と記録が可能

目的地・経由地 位置計測	• モビリティの実際の位置をメジャー等で計測し、アプリUIへ入力
計測結果を 表示	• 3Dマップ上にモビリティ実位置が重畳表示される
推定位置 との誤差表示	• モビリティ推定位置と実位置の誤差がアプリUIに表示される
ログ記録	• モビリティ推定位置、実位置、目標経路の情報をログに記録する



システムテスト結果一覧

システムテストの結果は以下のとおり

対象	試験項目	確認内容	結果
ARナビアプリ	統合モデルの取り込み	• 統合モデルのUSDファイルをUnity上にインポートし、外観・スケールが正しいか	合格
	ナビメッシュの生成	• Unity上にてナビメッシュを生成し、生成したナビメッシュが、モデル統合の境界面等で途切れることなく、連続しているか	合格
	経路探索	• Unity上にて、ナビメッシュを用いて最短経路探索を実施し、スタート地点からゴール地点まで連続した経路が生成されるか	合格
	画面確認	• 開発したARナビアプリの「ナビ開始画面」「ナビ案内中画面」「目的地到着画面」が想定通り動くか	合格
	案内確認	• ゲームエンジン上で生成したルート情報を元にARナビアプリで目的地までの案内をする際に、経路から大きく逸脱することなく、ゴールに到達するか	合格
パーソナルモビリティ運用アプリ	統合モデルの取り込み	• 統合モデルのUSDファイルをUnreal Engine上にインポートした際に、外観・スケールが正しく反映されているか	合格
	コリジョン（衝突判定）の付与	• Unreal Engineの機能を用いて、統合モデルにコリジョンを生成し、床・壁・柱等の形状とコリジョン形状が一致しているか	合格
	ナビメッシュの生成	• Unreal Engine上にて、コリジョン情報を用いてナビメッシュを生成し、生成したナビメッシュが、モデル統合の境界面等で途切れることなく、連続しているか	合格
	経路探索	• Unreal Engine上にて、ナビメッシュを用いて最短経路探索を実施し、スタート地点からゴール地点まで連続した経路が生成されるか	合格
	モビリティの走行	• モビリティを経路に追従走行させ、モビリティが経路から大きく逸脱することなく、ゴールに到達するか	合格
	誤差の算出・表示	• 誤差計測ポイント付近において、実際のモビリティ位置を計測し、入力し、アプリ上の推定モビリティ位置との誤差が算出・表示されるか	合格

Ⅰ．実証概要

Ⅱ．実証技術の概要

Ⅲ．実証システム

Ⅳ．実証技術の検証

Ⅴ．成果と課題

検証概要

目的に合わせた3Dモデルを適切なデータ統合手法で実施するために、統合手法ごとのジオメトリ精度と統合工数を比較検証し、各手法に適するシチュエーションを検討する

項目	内容
検証目的	• ナビゲーションやパーソナルモビリティ運用などアプリケーションの目的に応じた3Dモデルが必要となり、過不足のない工数・精度の統合データを作成するために統合手法ごとの差分を検証する - 各3Dモデルでは統合される元データのLODが異なるためジオメトリ精度にもバラツキがあるため、単純なモデル統合を行うとモデル境界にて大きなズレが発生してしまうことが想定される
検証方法	①ジオメトリ精度検証 - 各3Dモデル統合手法毎にジオメトリ精度に差があるかを測定し、各デジタルツイン活用アプリに利用できるジオメトリ精度であるかを検証する ②3Dモデル統合の工数検証 - 各3Dモデル統合手法のジオメトリ精度と作業工数算出を行う

検証ポイント

前処理と統合処理の2つの観点で各ポイントについて評価を行う

観点	ポイント
前処理	• メッシュ変換/軽量化 - 用途を満たし、現実的な処理時間で作成できるメッシュ数の目安 • LOD3とLOD1の統合 - 位置合わせの要否 - 重複部切り取り作業
統合処理	• モデル位置合わせ - 用途を満たすだけの精度を保持しつつ、作業が容易な位置合わせの手法の比較検討 • モデル修正 - 経路探索を行うために必要な修正作業の洗い出し • データ出力 - 経路探索を行うために必要なデータサイズと精度

Ⅳ. 実証技術の検証 ＞ 1. 統合データの検証 ＞ ②検証結果
結果サマリ

統合手法	作業時間		精度比較	示唆
	本町	天満		
手法1	8.0h	6.25h	• 重なり代が少ない場合は、離れが大きくなる（本町） - 199mmのずれが発生 • 各データの重なり代が確保できる場合には、他の手法との大きな精度の差はない（天満）	• 操作する点群の数が少ないため、作業時間は短く抑えることができ、ARナビ等20cm程度の誤差が許容される用途であれば問題がないことが確認された • 手法1はデータ同士の重なり代が大きいケースで利点を生かした統合となる - 重なり代が大きいと調整が効くため、精度を維持しつつ作業時間を短くできる
手法2	-	7.5h	• 3つのデータでの建物データの精度に差があるため、複数の参照点の抽出が難しい箇所があった - 3D都市モデル（LOD3)およびBIMモデルはディテールが省略されている一方、点群は実際のディテールが表現される	• 手法2は建物のディテール表現に差がないケースで利点を生かした統合となる - 複数の参照点を設定する必要があるため、モデル間で精度の差が少ない必要がある
手法3	-	8.5h	• CGLLは3D都市モデルと点群のズレが小さかったため、他手法と精度の差はなかった	• 手法3は3D都市モデルと点群データ間での誤差が大きいケースで利点を生かすことができる - モデル間でのズレを調整しながら統合できる点が利点となるため、誤差が少ないと工数が多くなる一方で精度は上がらない
手法4	8.0h	-	• 複数の平均によるため、箇所によっては手法1よりも精度が落ちるおそれがある • 局所的にズレが発生している箇所では、その部分だけを手法4を用いて変形し統合することで精度を高めることができた	• ズレが平均化されるので、最大誤差は手法1と比べて小さい • 一方で、精度が低いモデル同士の統合では全ての個所にある程度の誤差がある状態になる

ジオメトリ精度｜統合モデルの精度の比較（本町）

2手法ともに200mm程度の精度でデータ統合ができていることを確認できた

統合手法1

- 基準とした接続部において2モデル間のズレは199mm程度となった

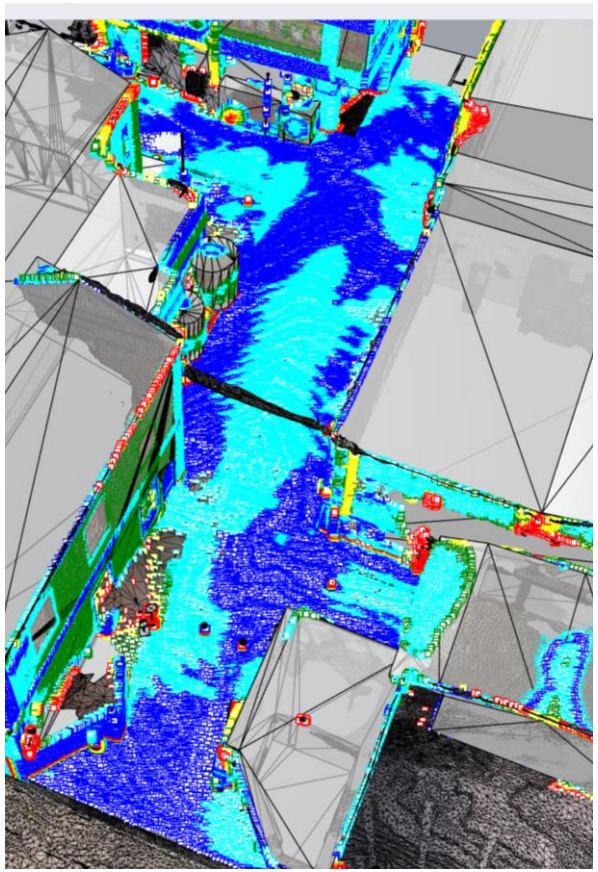
統合手法4

- 基準とした接続部ではズレは98mm程度となった
- 手法1で基準点としていた個所においても121mm程度のズレがあった
  （手法1では199mm程度のズレがあった箇所）

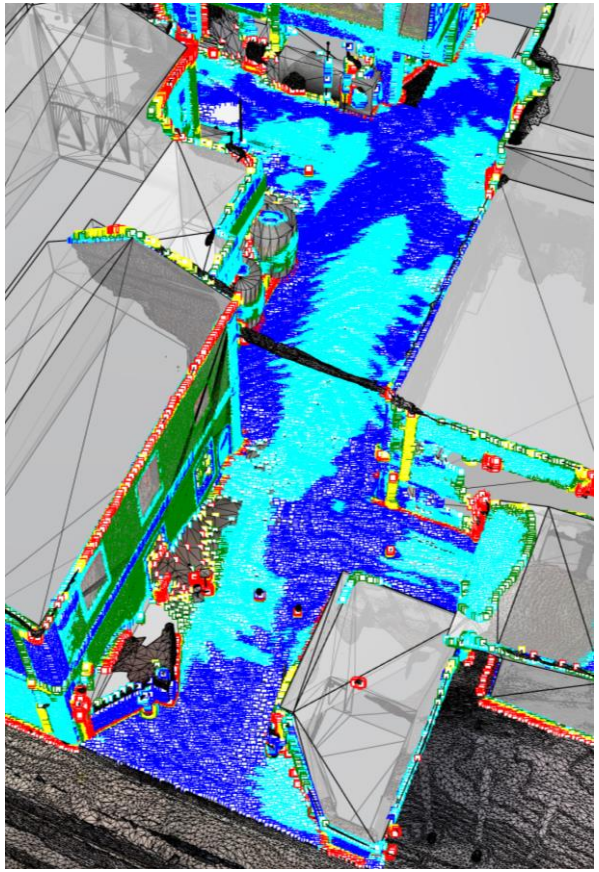
ジオメトリ精度｜統合モデルの精度の比較（天満CGLL）

倉庫面である地面は不陸の影響はあるものの、3手法ともに30mm前後の精度で統合ができていることが確認できた

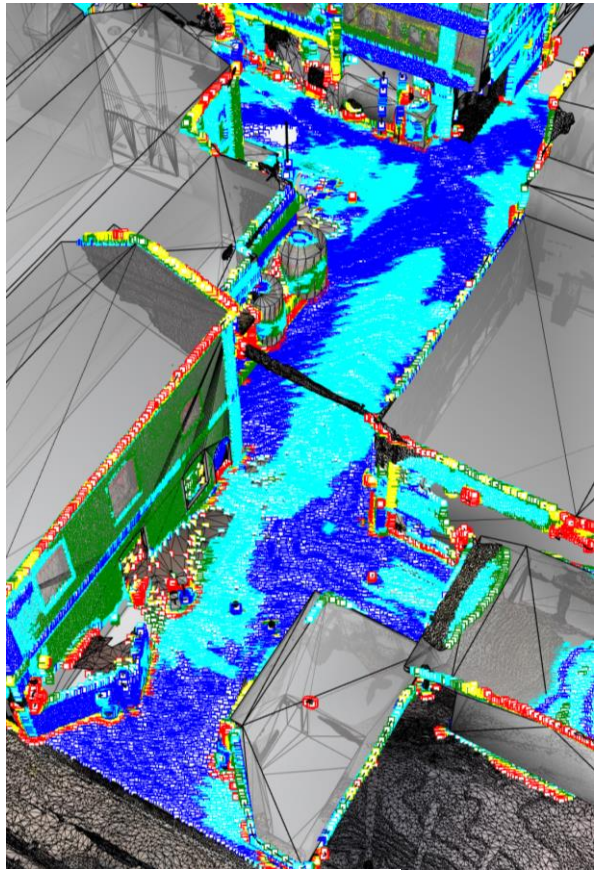
統合手法1



統合手法2



統合手法3



- 天満CGLLは、対象エリアの点群があるため、ガイドに用いた敷地内および建物内の点群データを基準に精度検証を行う
- Rhinocerous 7の「PointDevitation」という機能を用いて偏差の分布により評価を行う

色*1	誤差寸法
	300～500mm
	～300mm
	～210mm
	～120mm
	～30mm

*1 本検証の誤差目標が±300mm以内のため、赤を300mm、青を10%の30mmに設定

3Dモデル統合の工数｜各手法の作業時間の比較

手法2と手法4においては、位置合わせをCloudCompareを用いて半自動で行うため、最も単純な手法1と比較すると差異が小さい一方で、点群をグループ化する必要のある手法3が突出して時間が掛かかっている

エリア	統合手法	前処理			統合処理		小計		合計
		3D都市モデル	点群データ	BIMモデル	位置合わせ	その他	前処理	統合処理	
本町	手法1	2.0h	2.5h	0.5h	2.0h	1.0h	5.0h	3.0h	8.0h
	手法4	2.0h	2.5h	0.5h	1.5h	1.5h	5.0h	3.0h	8.0h
天満	手法1	0h	3.25h	0.25h	1.25h	1.5h	3.5h	2.75h	6.25h
	手法2	0h	3.25h	0.25h	2.0h	2.0h	3.5h	4.0h	7.5h
	手法3	0h	4.25h	0.25h	2.0h	2.0h	4.5h	4.0h	8.5h

3Dモデル統合の工数｜統合手法1（本町）の作業時間

項目	作業内容	ソフトウェア	作業時間
3D都市モデル統合	3D都市モデル(LOD3/LOD1)読み込み/必要部分の切出し/ファイル書き出し	CloudCompare	1.5h
3D都市モデルの重なり部分削除	LOD1とLOD3で重なっている部分があるので、LOD1の重なり部分を切り出して削除する	Blender	0.5h
点群初期処理	点群統合/リダクション/メッシュサーフェス化	CloudCompare	2.0h
メッシュ軽量化	点群から変換したメッシュサーフェスをDecimateモディファイアでリダクションし、軽量化する	Blender	0.5h
BIMモデルの準備	ファイル書き出し	Archicad	0.5h
位置合わせ	3D都市モデル、BIM、メッシュサーフェスを手順１にて目視で位置合わせを行う	Blender	2.0h
モデル編集	３D都市モデルの入り口のメッシュを削除、不要なメッシュの削除、微調整、USDで書き出し	Blender	1.0h

3Dモデル統合の工数｜統合手法1（天満）の作業時間

項目	作業内容	ソフトウェア	作業時間
点群初期処理（ガイド用点群）	点群の読み込み/点群のリダクション 100mm/PLYで書き出し	CloudCompare	0.25h
点群初期処理 （メッシュ化用点群）	点群の読み込み/点群のリダクション 5mm 不要な点群（入り口、荷物など）の削除/e57で書き出し	CloudCompare	1 .0h
メッシュサーフェスの生成	点群の読み込み/メッシュの構築/穴を塞ぐ/浮遊ノイズの除去/ メッシュのリダクション/FBXで書き出し	Metashape	2.0h
BIMモデルの準備	ファイル書き出し	Revit	0.25h
3D都市モデルの読み込み	原点付近に移動	Blender	0.25h
3D都市モデルと点群の位置合わせ	基準点の設定、移動と回転により位置合わせ	Blender	0.5h
メッシュサーフェスとBIMモデルの 位置合わせ	基準点の設定、移動と回転により位置合わせ、ガイド点群を非表示後、微調整を行う	Blender	0.5h
モデル編集	3D都市モデルの入り口のメッシュを削除、メッシュを全て表面に変換、不要なメッシュの削除、微調整、USDで書き出し	Blender	1.5h

3Dモデル統合の工数｜統合手法2（天満）の作業時間

項目	作業内容	ソフトウェア	作業時間
点群初期処理 （メッシュ化用点群）	点群の読み込み/点群のリダクション 5mm 不要な点群（入り口、荷物など）の削除	CloudCompare	1.0h
メッシュサーフェスの生成	メッシュの構築、穴を塞ぐ、浮遊ノイズの除去、メッシュのリダクション、FBXで書き出し	Metashape	2.0h
BIMモデルの準備	ファイル書き出し	Revit	0.25h
メッシュサーフェスの位置合わせ	メッシュサーフェスの読み込み、FBXで書き出し	CloudCompare	0.25h
点群と各モデルの位置合わせ	移動と回転で大まかに位置合わせ、複数の参照点をもとに位置合わせ、微調整、メッシュ生成が必要な箇所のみをトリミング、e57で書き出し	CloudCompare	2.0h
モデル編集	インポートする際に縮尺の確認、BIMデータの置き換え、3D都市モデルの入り口のメッシュを削除、メッシュを全て表面に変換、不要なメッシュの削除、微調整、USDで書き出し	Blender	2.0h

3Dモデル統合の工数｜統合手法3（天満）の作業時間

項目	作業内容	ソフトウェア	作業時間
点群初期処理 （メッシュ化用点群）	点群のリダクション、不要な点群の削除e57で書き出し	CloudCompare	2.0h
メッシュサーフェスの生成	メッシュの構築、穴を塞ぐ、富裕ノイズの除去、メッシュのリダクション、FBXで書き出し	Metashape	2.0h
BIMモデルの準備	ファイル書き出し	Revit	0.25h
メッシュサーフェスの位置合わせ	メッシュのインポート、FBXで書き出し	CloudCompare	0.25h
3D都市モデルを基準に点群データを位置合わせ	3D都市モデルと点群データを読み込み、点群データを基準となる箇所ごとにグループ化、全ての点群を同時に移動と回転で位置合わせ、グループごとに位置合わせ、点群を統合、点群の間引き、不要な点群を削除、e57で書き出し	CloudCompare	2.0h
モデル編集	インポートする際に縮尺の確認、BIMデータの位置合わせ、3D都市モデルの入り口のメッシュを削除、メッシュを全て表面に変換、不要なメッシュの削除、微調整、USDで書き出し	Blender	2.0h

3Dモデル統合の工数｜統合手法4（本町）の作業時間

項目	作業内容	ソフトウェア	作業時間
3D都市モデル統合	3D都市モデル(LOD3/LOD1)読み込み/必要部分の切出し/ファイル書き出し	CloudCompare	1.5h
3D都市モデルの重なり部分削除	LOD1とLOD3で重なっている部分があるので、LOD1の重なり部分を切り出して削除する	Blender	0.5h
点群初期処理	点群統合/リダクション/メッシュサーフェス化	CloudCompare	2.0h
メッシュ軽量化	点群から変換したメッシュサーフェスをDecimateモディファイアでリダクションし、軽量化する	Blender	0.5h
BIMモデルの準備	ファイル書き出し	Archicad	0.5h
位置合わせ	3D都市モデル、BIM、メッシュサーフェスを複数の参照点をもとに位置合わせ	CloudCompare	1.5h
モデル編集	3D都市モデルの入り口のメッシュを削除、不要なメッシュの削除、微調整	Blender	1.0h
モデル接合部変形	モデルの接合部で離れが大きい部分について、局所変形を行なって位置合わせを行う（手法４）	Blender	0.5h

検証内容｜概要

統合手法1、統合手法4で統合されたモデルを使い、どのモデルがARナビに適するかを検証する

検証内容

項目	内容
目的	ARナビによる3D統合モデルの精度の評価
実施日時	2022年11月22日（火）13時～16時
天候	晴れ
実施場所	大阪メトロ御堂筋線本町駅周辺（B1Fから地上階を経て竹中工務店本社に至るエリア）
検証内容	• ARナビ案内ルートをARナビアプリの指示のもと移動し、想定ルートおよび計測ポイントとの差分を測定する • KPI(±2.5m以内)と比較することで、3D統合データの統合手法の評価を行う

検証条件詳細

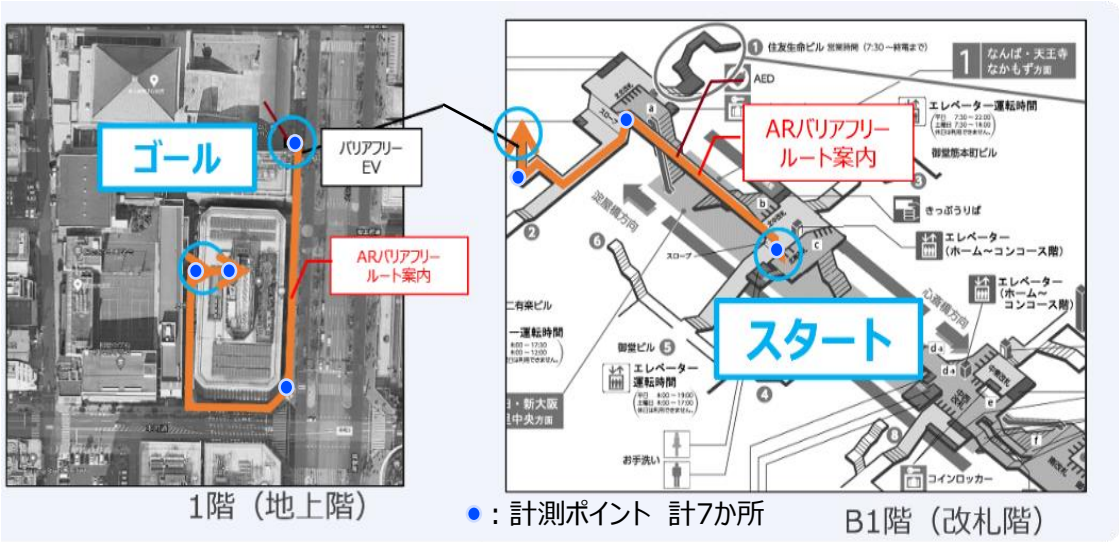
項目	条件
検証モデル	①統合手法1で作成された3Dモデル ②統合手法4で作成された3Dモデル *ARナビアプリは3Dモデルごとに用意する
検証時間帯	• 正午～夕方までの時間帯 - 影や光源がない影響により自己位置推定の点群が認識しづらくなる時間帯は避ける
端末位置	• 操作者の胸の前で端末を安定させた状態を保つ
歩行速度	• 車いすを想定した速度（3~4km/h）を意識して歩く
測定者	• 測定者の違いによる影響を無くすため同一人物が測定する
自己位置推定方法	• 事前に取得した点群から自己位置を推定する • 自己位置推定のために立ち止まるなどの行為は行わない - 例外として、スタート地点およびエレベータ降車地点のみ立ち止まり自己位置推定を行う

検証内容｜精度の評価

3D統合モデルの精度について、ARナビの運用に支障が無いか実際の案内ルート上に計測ポイントを設定し、許容可能誤差をKPIとして評価を行う

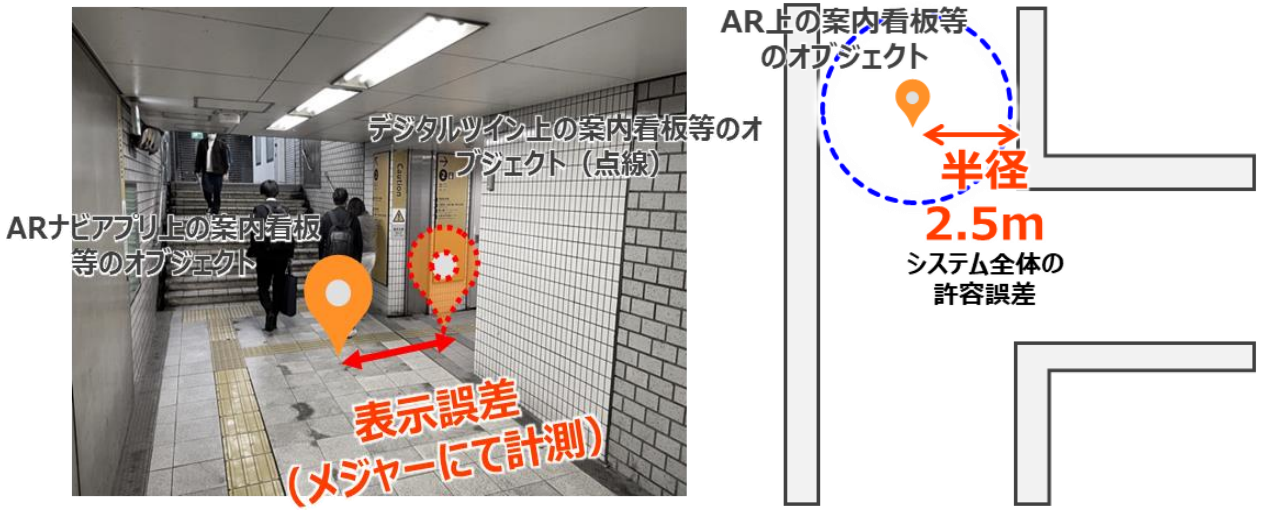
精度の評価方法

- ナビルートが分かれる通路にて利用者を正しいルートに導くことが可能な範囲を許容可能誤差としてKPIとする
- 案内ルート上で分岐が発生する箇所の通路幅は約5mとなるため、半径2.5m以内をARナビシステムの許容誤差とする



ARナビ案内ルートおよび計測ポイント

誤差の測定イメージ



誤差測定方法

検証結果｜統合手法別結果サマリ

いずれのモデル統合手法もARナビには十分なモデル精度を有することが分かった

統合手法	検証結果	判定	示唆
統合手法1	最大0.41m （検証ポイント③）	○	• 誤差の最大値は、いずれの統合手法においてもKPIで定めた±2.5m以内を達成した - いずれの手法も検証ポイント③では他ポイントに比べ誤差が比較的大きくなったものの、VPSの仕組みによるもので、KPIを達成していることから実用上の問題はない • いずれのモデル統合手法も、本検証におけるARナビには十分なモデル精度を達成していたと考えられる - 誤差の平均値は手法1で0.21m、手法4で0.23mとなっておりARナビのナビゲーションの矢印もほぼ通路の真ん中に表示される状態であるといえる
統合手法4	最大0.90m （検証ポイント③）	○	

検証結果｜各計測ポイントでの誤差

検証ポイント③での誤差が手法1では0.41m、手法4では0.90mと最大となり、7か所のポイントの平均値は両手法でほぼ同等の精度であり、どちらを用いてもARナビアプリで必要な精度には達すると考えられる

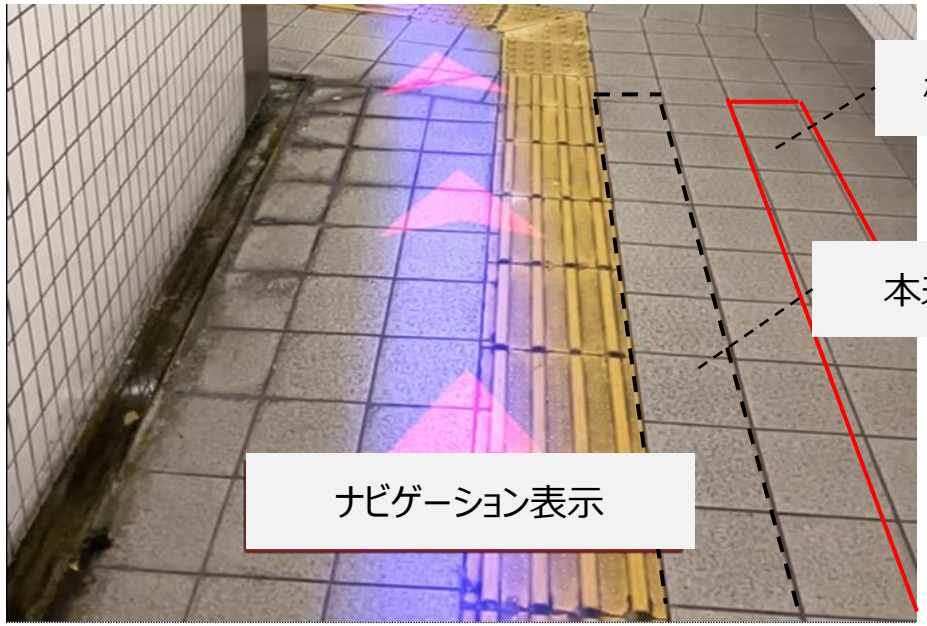
		①	②	③	④	⑤	⑥	⑦	(m)
		本町駅 改札前	**B1階 経由地点**	**B1階 エレベータ前**	**地上階 エレベータ前**	**竹中本社ビル 角**	**竹中本社ビル 裏口前**	**竹中本社ビル エレベータホール前**	**平均値**
手法1	**平均値**	**0.12**	**0.17**	**0.29**	**0.12**	**0.30**	**0.26**	**0.23**	**0.21**
	1回目	0.10	0.33	0.29	0.16	0.30	0.17	0.24	0.23
	2回目	0.07	0.09	0.41	0.07	0.28	0.29	0.28	0.21
	3回目	0.17	0.10	0.16	0.13	0.33	0.31	0.17	0.20
手法4	**平均値**	**0.06**	**0.06**	**0.61**	**0.14**	**0.28**	**0.19**	**0.24**	**0.23**
	1回目	0.08	0.02	0.21	0.09	0.22	0.27	0.29	0.17
	2回目	0.06	0.06	0.72	0.10	0.29	0.14	0.16	0.22
	3回目	0.05	0.09	0.90	0.24	0.33	0.17	0.26	0.29

検証結果｜個別に確認された課題（1/2）

VPSが苦手とするパターン模様によって誤差の値が大きくなっていたと考えれられるため、自己位置推定には固有の特徴を持つオブジェクトを利用することで精度を担保することが可能と考えれられる

確認された課題

- 検証ポイント③ではKPIである2.5mを達成はしているものの、他ポイントに比べ誤差が比較的大きくなった
- 両手法でも、検証ポイント③が最大誤差の発生したエリアとなった



考察

原因

- エリアの床が同じ模様の繰り返しとなっており、特徴が似ている空間が続くことで自己位置推定機能の誤検知だと考えらえる
  - 上記根拠として、本来のタイル位置とは異なる位置にタイルが検知されたことでタイルに沿って左右にナビゲーションルート表示がぶれる挙動を確認した

対策案

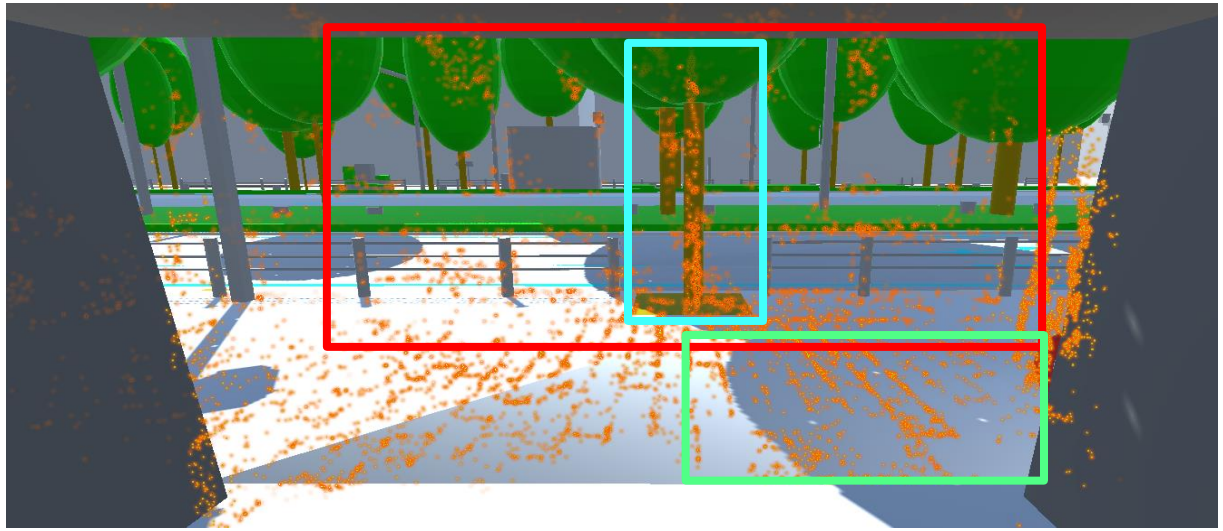
- 同じ模様が繰り返される床や壁のエリアの点群で自己位置推定を行わない
- エリアで一意なオブジェクトやポスターの点群を取得する

検証結果｜個別に確認された課題（2/2）

VPS用点群データの不足から自己位置推定に時間を要するという課題が確認されたが、より精緻な点群データを取得する、UXによりユーザーのアクションを誘発し特徴点を捉えられる場所に移動させるなどの対策が考えられる

確認された課題

- 検証ポイント④のエレベータで地下から地上へ移動後に自己位置推定した際、想定以上の時間がかかった
  - 通常は自己位置推定に10秒程度で済むところを3分間を要した



想定していた自己位置推定に必要な点群
実際に自己位置推定に必要だった点群
その他の自己位置推定可能な点群

考察

原因

- 周囲のガードレール前に人だかりがありカメラが自己位置推定する際の障害物となったと考えられる
  - ガードレール前に人だかりがない場合は、木の周辺にカメラを向けることで自己位置推定が可能であった
  - 自己位置推定は木の点群のみ（青枠部分）を利用していると想定していたが、周辺のガードレール部分の両方の点群（赤枠部分）を利用していたと考えられる

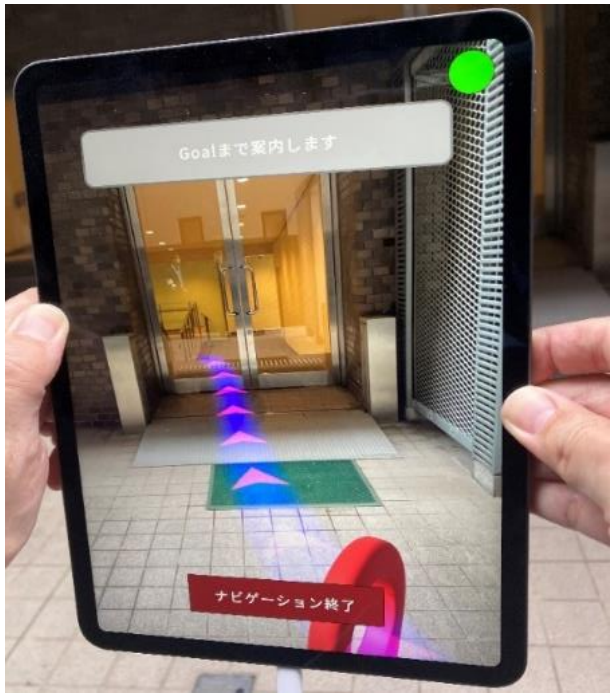
対策案

- VPS用点群データを広範囲に渡って取得する
  - 緑枠部分の地面の点群のみでも自己位置推定が可能であることを確認済み
- 自己位置推定ができない場合に、周囲を見回す、少し移動して特徴のあるものを見つけるなど、アクションを促すメッセージを表示することで、UXとしての品質を担保する

検証結果｜実証実験の様子

ARナビの実証実験の様子は以下の通り

ARナビの様子



誤差測定の様子

検証内容｜検証概要

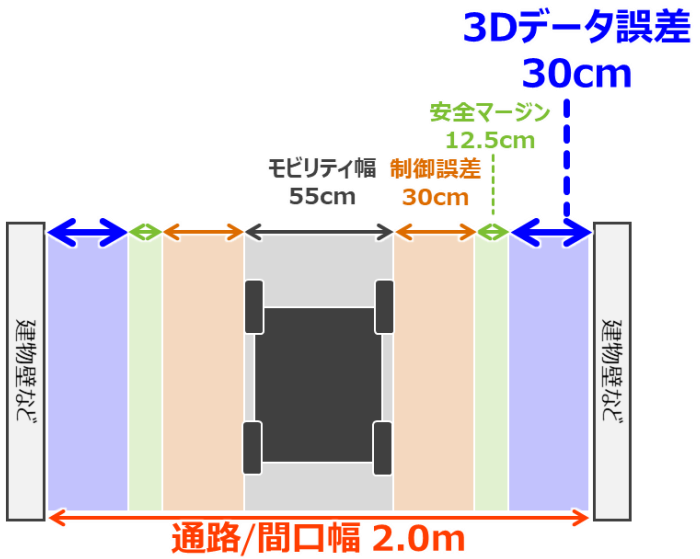
統合手法1～3で統合されたモデルを使い、どのモデルがパーソナルモビリティ運用に適するかを検証する

検証概要

目的	パーソナルモビリティによる3D統合モデルの精度の評価
実施期間	2022年11月24日（木）
実施場所	大阪市天満周辺 CGLL中西金属工業（株）本社内
検証内容	• パーソナルモビリティ運用実証において、デジタルツイン上のルートと現実のルート(走行軌跡)との誤差を複数の経由地点とゴール地点で検証する - 誤差目標とするKPIは60cm（±30cm）以内に設定
検証対象の統合手法	• 統合手法1 • 統合手法2 • 統合手法3

KPIの考え方

- モビリティ幅や制御誤差を加味して3Dデータに許容される誤差は30cmと算出し、KPIとする
  - 通路/開口幅：対象エリアの実測値から2.0mに設定
  - 制御誤差：モビリティの制御仕様から30cmに設定
  - 安全マージン：モビリティ幅の約20%をマージンとして12.5cmに設定



検証内容｜ナビメッシュの設定

パーソナルモビリティ運用アプリが正しくモビリティをガイドできるよう、検証内容と現地の状態に合わせて、ナビメッシュの設定を調整した

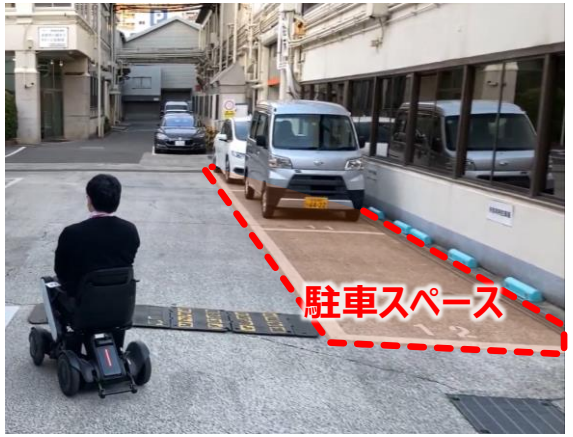
ナビメッシュの生成設定

- モビリティの仕様をもとに、以下の通りナビメッシュの生成設定を行いナビメッシュを生成した

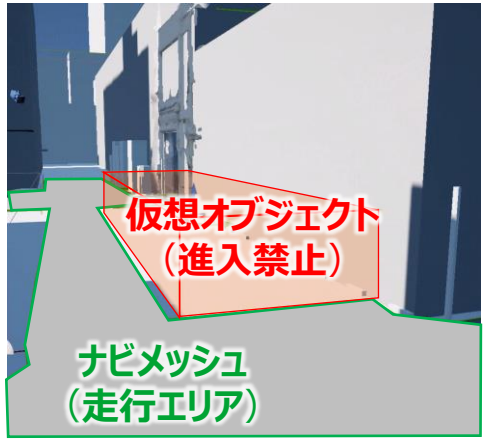
項目	内容
エージェント半径	（モビリティの横幅55cm+安全マージン12.5cm）÷2
エージェント登坂力	10度（モビリティ仕様より）
エージェント乗越段差	2cm • 自律走行において安全に乗り越えが行えるよう考慮 • モビリティ仕様の5cmは手動操作における最大性能であり、段差の形状や、進入角度・速度によっては乗り越えできない場合があることを想定

屋外のナビメッシュ設定

- 検証エリア現地屋外では、統合モデルに存在しない駐車スペース・駐車車両が存在していた
- モビリティが駐車スペースに進入することのないよう、進入禁止エリアを示す仮想オブジェクトを配置し、ナビメッシュの再生成を行い、ナビメッシュが自動的に編集され、駐車スペースに進入しない安全な経路の算出が可能となる



現地の状況（屋外）



編集後のナビメッシュ

屋内のナビメッシュ設定

- 屋内における、統合モデルに存在しない什器や資材の仮置エリア等についても、同様に対処を行った

ナビメッシュ生成結果（屋内通路）

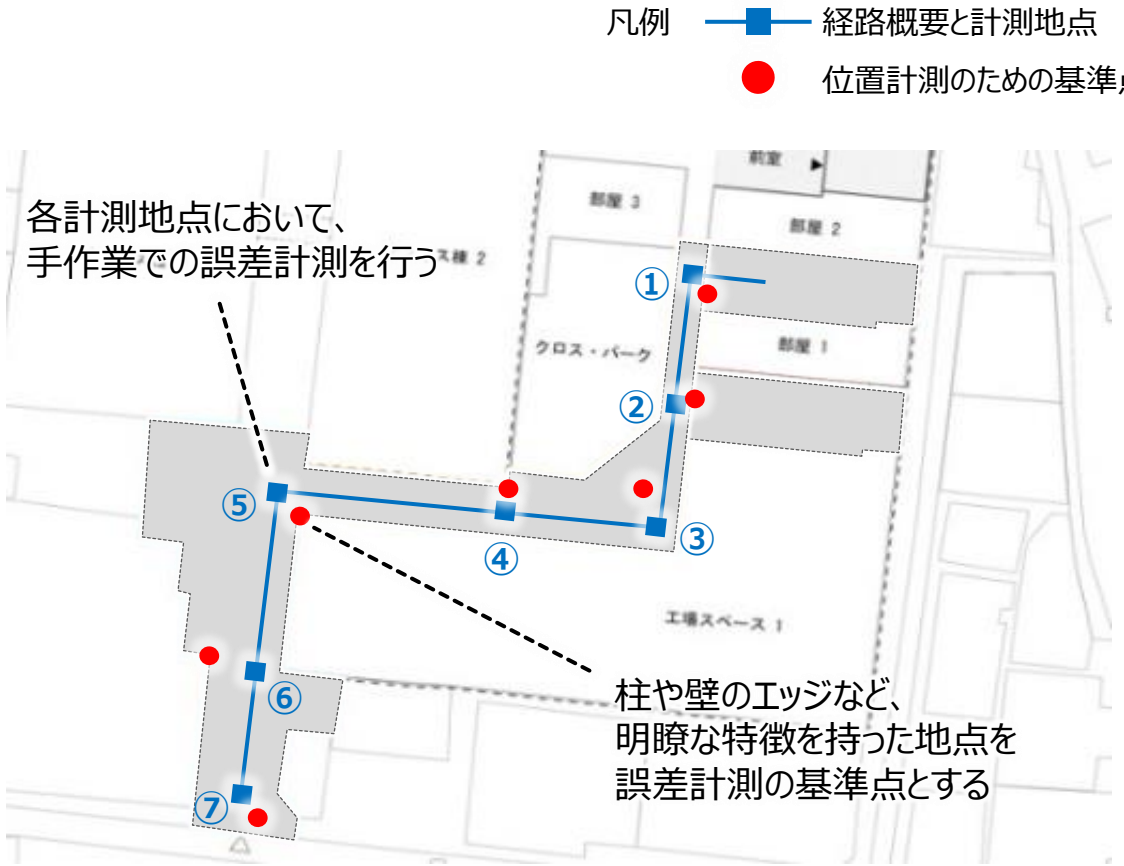
検証内容｜走行ルートと計測地点

CGLLの室内から屋外にかけてのルートを設定し、ルート上の明瞭な特徴点を持つ7か所を基準点として設定し、計測を行う

検証方法

- 建物の柱や壁エッジ部分など、明瞭な特徴を持った7地点を基準点として設定する
- 計測地点①付近をスタート/ゴール、⑦付近を折り返し地点として、モビリティを往復走行させる
- モビリティが各基準点付近に到達時、モビリティを一時停止させ、基準点に対するモビリティの位置を計測・記録する

モビリティ位置計測地点および基準点の配置

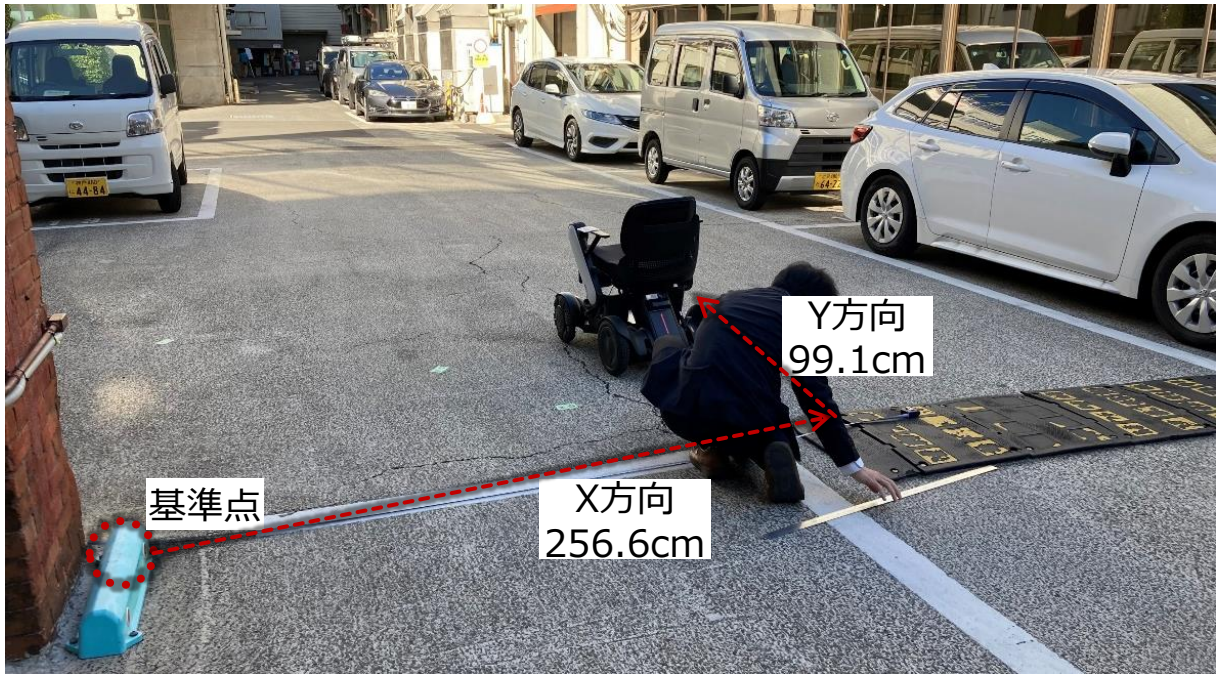


検証内容｜位置の測定

現実と仮想空間で合わせた基準点からの距離をメジャーを使い測定し、システム上に入力すること誤差の算出に利用する

実モビリティ位置の測定

- 建物の柱や壁エッジ部分など明確な特徴をもつ基準点に対する相対位置をメジャー等で計測する



仮想空間上でのモビリティ表示

- 計測した実モビリティ位置を制御アプリに入力し、仮想空間上でのモビリティ位置と重ね合わせる

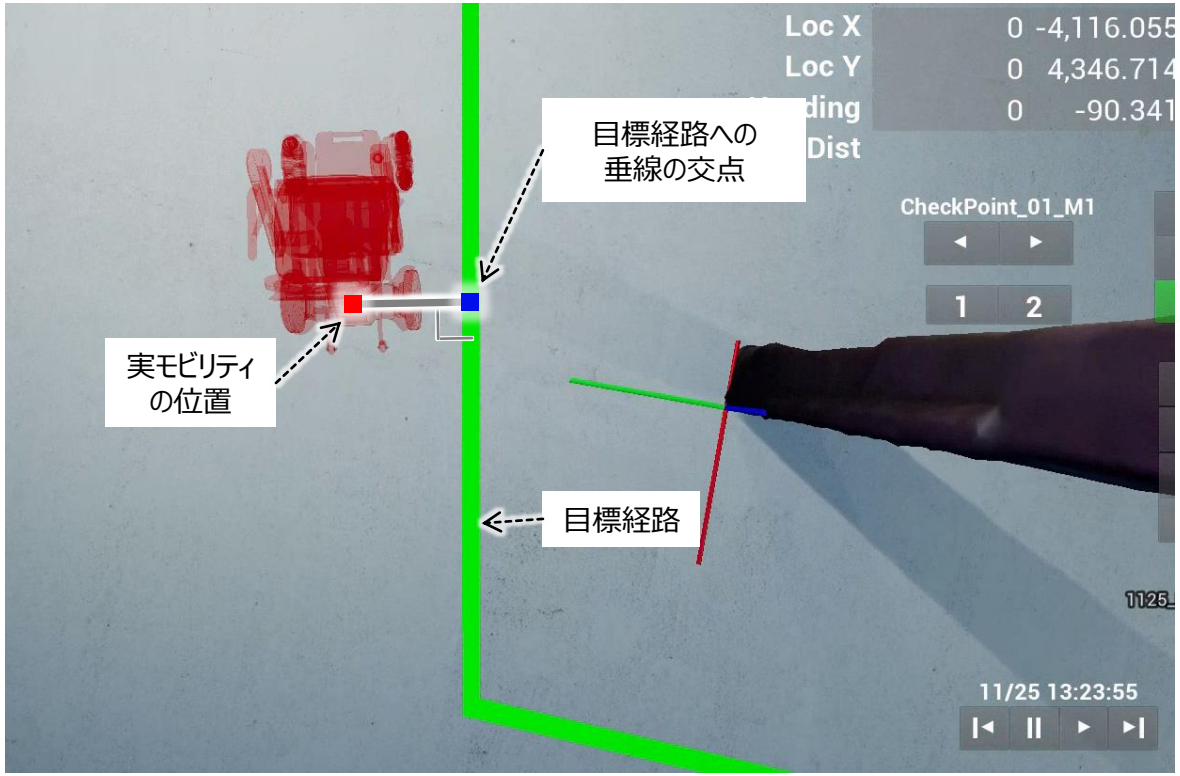
検証内容｜誤差の算出

計測・記録した実モビリティの位置をもとに、モビリティの位置誤差（目標経路に対する実モビリティ位置の横方向偏差）を求める

算出方法

- 計測・記録した実モビリティの位置をもとに、モビリティの位置誤差として垂線の長さを求める
  - 目標経路に対する実モビリティ位置の横方向偏差を算出する
- 算出方法は以下の3ステップ
  1. 制御アプリに記録された、実モビリティ位置および目標経路の座標を求める
  2. 実モビリティ位置から最も近い経路上に垂線を降ろす
  3. 垂線の長さを求める

算出イメージ



検証結果｜結果サマリ

いずれのモデル統合手法もパーソナルモビリティ運用には十分なモデル精度を有することが分かった

統合手法	検証結果	判定	示唆
統合手法1	最大51.2cm（復路④）	○	• 誤差の最大値は、いずれの統合手法においてもKPIで定めた60cm以内を達成した - 往路において地点③⑤の直前に急な旋回を行う必要があるが、旋回時にオドメトリによるモビリティ自己位置推定の誤差が大きくなることが原因と考えられる - カーブ通過後、直線区間をしばらく走行する際にモビリティ位置はLiDARで補正されるため、カーブで生じた誤差は蓄積されることはなく、目標精度を達成することができた • いずれのモデル統合手法も、本検証におけるモビリティ運用には十分なモデル精度を達成していたと考えられる - 誤差の平均値は、統合手法3を用いた場合に最も小さくなったが、平均誤差の差は0.8cm程度とわずかであった
統合手法2	最大53.4cm（復路③）	○	
統合手法3	最大51.8cm（往路⑤）	○	

検証結果｜各地点での誤差の大きさ

旋回位置となる③と⑤の位置で誤差が大きくなっているが、その前後では誤差の値は収まっており、モビリティ側の旋回動作の制限による影響があると考えられる

KPIの50%を超えた箇所

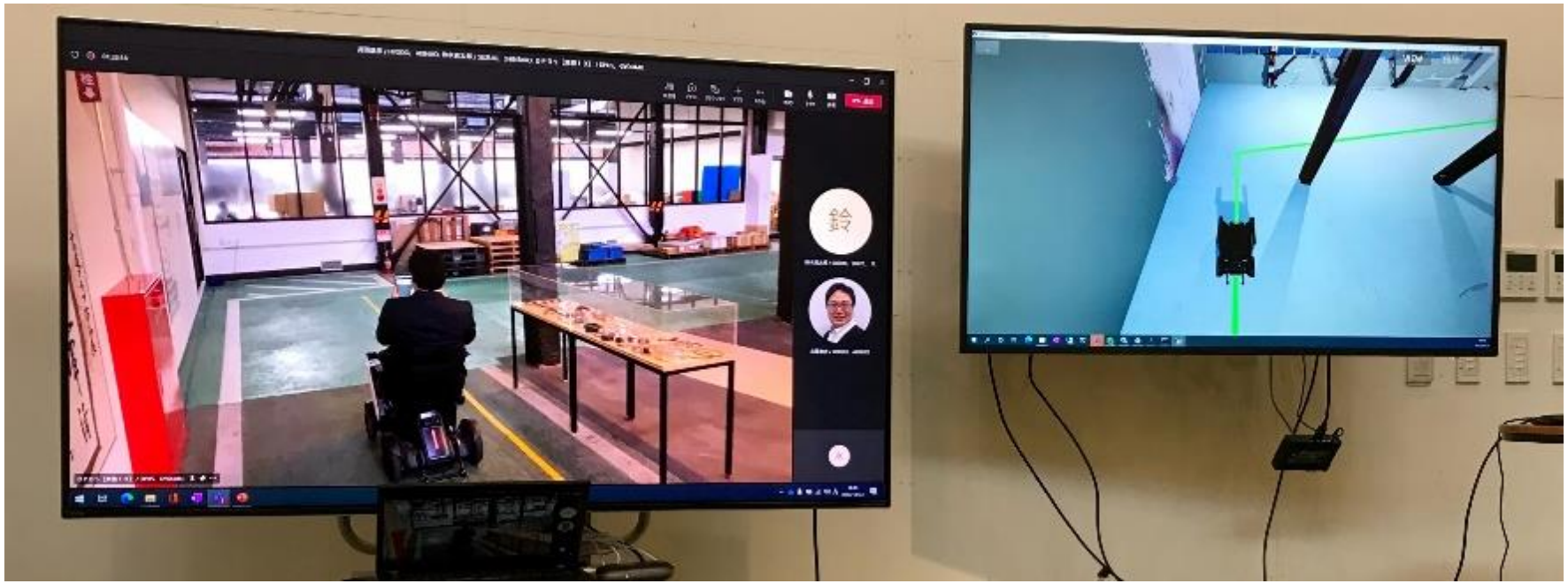
統合手法	項目	誤差（cm）													
		往路							復路						平均値
		①	②	③	④	⑤	⑥	⑦	⑥	⑤	④	③	②	①	
統合手法1	各3試行の平均値	6.2	11.1	35.3	17.3	33.1	15.4	19	18.4	10.9	26.9	22.1	11.3	16	18.7
	1回目	8.3	11.2	11.7	22.6	24.6	16	13.9	6.7	3.6	51.2	22.1	13.7	21.5	17.5
	2回目	6.2	11.9	49.2	25	36.3	18.6	25.9	21.6	12.7	9.3	18	1.6	10	18.9
	3回目	4	10.1	45	4.2	38.3	11.6	17.2	27	16.5	20.3	26.2	18.7	16.5	19.7
統合手法2	各3試行の平均値	8.2	16	25.5	18.2	39	18.5	6.6	13.9	6.3	14.5	32.2	18.2	22.7	18.4
	1回目	7.4	4.5	9.7	26.9	40.9	13	8.6	0.5	14.9	21.5	53.4	6.3	12.9	17.0
	2回目	2.5	41.8	28.1	20.4	49.3	15.3	10	12.1	0.2	10.5	23.9	13.6	32.6	20.0
	3回目	3.6	4	28.9	12.1	22.5	28.5	24.1	7.7	7.2	6.8	14.5	14.2	26.6	15.4
統合手法3	各3試行の平均値	2.8	18.1	36.1	12.7	39.8	27.9	23.5	5.7	15.5	6.2	17	9	18.1	17.9
	1回目	4	39.9	36.6	14.6	45.2	21.9	20.1	0.4	7	9.2	23.3	2.2	14.1	18.3
	2回目	0.8	10.3	42.9	11.5	51.8	33.2	26.3	9.1	32.4	2.5	13.1	10.6	13.5	19.8
	3回目	14.7	1.7	38.6	7.2	26.8	27.3	1.3	29.1	3.7	11.4	19.3	34.6	22.5	18.3

検証結果｜実証実験の様子

実証実験の様子は以下の通り

パーソナルモビリティの自律走行

移動の様子とデジタルツイン仮想空間上の移動イメージ



Ⅰ．実証概要

Ⅱ．実証技術の概要

Ⅲ．実証システム

Ⅳ．実証技術の検証

Ⅴ．成果と課題

Ⅴ. 成果と課題 ＞ 1. 今年度の実証で得られた成果
①3D都市モデルによる技術面での優位性

項目	想定される技術面での優位性
3D都市モデルを軸とした屋内外データ整備	• これまでに組み合わせて利用されていた3D都市モデルとBIMモデルに加えて、点群データも扱える統合手法を構築することで、屋内外をシームレスに繋ぐ経路探索のユースケース（ARナビゲーションおよびモビリティ自律運行システム）を実現可能な精度で、三次元統合空間を都市スケールで作成することが可能になった - 精度管理された公共測量成果であり広域に整備された3D都市モデルを活用することで、都市スケールで一貫した精度を保つデジタルツインを構築することが可能 - BIMモデルがない、設計BIMモデルしかなく現実と差異が大きいケースにおいて、独自に点群データをスキャンすることで屋内データを代替することが可能
モデル統合による段差や障害物など移動上の制約となる情報の活用	• 交通弱者や移動ロボットなどへ移動経路上の障害物の情報などを提供することで、より快適なルート情報の提供が可能となる。特に3D都市モデルの道路LOD3を利用することで車道と歩道の区別が可能なため、通行可能な幅・高さを状況に応じて柔軟に変更し、安全な経路を生成することができる - 乗り越え可能な段差の高さを指定することで、濡れているときは滑りやすいので高い段差を避けたり、バリアフリーを考慮した個人の状態毎のルート探索ができる

Ⅴ. 成果と課題 ＞ 1. 今年度の実証で得られた成果
②3D都市モデルによるビジネス面での優位性

項目	想定されるビジネス面での優位性
3D都市モデル、点群データ、BIMモデルを統合した広域デジタルツインの利活用	• 3D都市モデル、点群データ、BIMモデルを組み合わせた屋内外を仮想空間でシームレスに扱うことができるデジタルツインを都市スケールで構築する手法を開発したことで、自律走行ロボットが屋内だけではなく都市空間も活動エリアに含めることに寄与できる - 都市空間と屋内外を接続した移動や、ラストワンマイル物流等への活用も示唆される
統合データを使ったゲームエンジンでの経路探索	• 3D都市モデル、点群データ、BIMモデルを統合したデータによって、屋内外をシームレスにつないだ歩行者ナビゲーション、パーソナルモビリティ運用を実現できたことから、本統合手法による統合データのゲームエンジンでの活用可能性が見出された - Unity（ARナビ）とUnreal Engine（パーソナルモビリティ運用）の双方のゲームエンジンの経路探索機能を利用した経路情報の導出ができており、シミュレーションへを活用した目的別の移動方法や、サイバー・フィジカル横断のエンタメ等、多様なサービスの応用可能性がある

活用にあたっての課題

項目	活用にあたっての課題
統合作業工数	• 3Dデータの統合手法はガイドライン化し、今後のデータ統合に役立てたいが、手作業による基準点設定など自動化はできない • 複雑な3D形状であるので、作業工数の低減化を検討することは今後の3Dデータの有効活用にもつながると考える
データ著作権	• 安全上、収益上他様々な要件で全ての3D空間情報を簡単に公開はできない • 商業施設などの来訪者など、公開範囲、利用目的を整理した上で、何処まで3Dデータを公開するかを様々な関係者とすり合わせていく必要がある
マネタイズ	• 3Dデータの統合や作成された3Dデータの活用とさらなる利用アプリケーションへの展開が都市の活性化目的だけでなく、収益を生み出すものとするべく、マネタイズへの取り組みを考えていく必要がある
自己位置推定の誤差	• 混雑時は誤差が大きくなる。アプリ側でカメラに写る情報から周囲の混雑度合を判断し、その混雑度に応じて適した自己位置推定エリアへの誘導を行うことで、障害物等の影響を回避することが可能と考えられる - 例えば、混雑時には地面部分の点群を見るようにUI上で指示され、ユーザ側で自己位置推定できるエリアを探すことなく即座に自己位置推定が可能となる • エレベータで地下から地上へ移動後に自己位置推定した際、想定以上の時間がかかる。人の移動のようなランダムに変化する対象は自己位置推定の難度が上がるため、固定されていて外部影響を受けづらい場所、モノの点群を取得する - 一意で変化しない部分の点群を取得することが望ましいと考えられる
移動物の取り扱い	• 運用時に経路上仮設物、駐車車両等一時的な障害物が発生する可能性があり、アプリケーション側で回避機能を有する必要がある
データ作成時点の差異	• BIMモデルが計画段階で構築された場合、実際と差異が発生する可能性があり、3Dデータと実空間との時間的整合性を確認する仕組みが必要である

用語集（1/4）

用語		内容
ア行	ROS	• Robot Operating Systemの略 • ロボット開発において重要な役割を果たすオープンソースのロボット制御ソフトウェア、およびそれを包括するロボット開発プラットフォーム全体
	Unreal Engine	• Epic Games社により開発・提供される、ゲームエンジンの一種
	WHILL	• WHILL社が開発した電動車椅子 • パーソナルモビリティの一種
	Web Socket	• ブラウザとウェブサーバーとの間で双方向通信を行うための通信規格
	ARナビ	• ARナビゲーションの略 • 実際に見えているスマホなどの画像上にナビを重ね合わせたナビゲーション
	FBX	• Autodesk社が開発した3Dモデリング、アニメーション、レンダリングなどのソフトウェア間でのデータ交換を可能にするファイルフォーマット • 3Dモデル、アニメーション、テクスチャ、ライティング、カメラなどのデータを保存可能 • デファクトスタンダードとなっており、様々なソフトウェアで本形式を扱うことができる
	MQTT	• Message Queueing Telemetry Transportの略 • メッセージ指向ミドルウェアのアプリケーション層で使用される、TCP/IPによるPub/Sub型データ配信モデルの軽量なデータ配信プロトコル
	OBJ	• 3次元コンピュータグラフィックス（3DCG）で用いる物体の形状データを記録するファイル形式の一つ • 三次元空間における物体の形状を表すデータで、頂点の座標、物体表面を構成する面の情報、曲線や曲面を表すパラメータなどをテキスト形式で記述する

用語集（2/4）

用語		内容
ア行	オドメトリ	• 自己位置姿勢推定法の一つ • 単位時あたりの車輪の回転角の積分によって、移動ロボットの現在位置や姿勢を推定する方法
カ行	仮想空間	• コンピューターの中に作られた仮想的な世界
	ゲームエンジン	• ゲーム開発を支援するソフトウェア • ゲームのグラフィックス、サウンド、AI、物理演算、インタラクションなどを実装するためのツールやライブラリを利用できる
サ行	サーフェス化	• 点群データを「点」から「面」の状態に変換すること
	CGPF	• コモングラウンドプラットフォーム • Society5.0の実現に向けた汎用的なインフラとなりえるプラットフォーム • 建築や都市の3Dデータをインデックスに、フィジカル空間とサイバー空間をリアルタイムにシームレスにつなぎ、人とロボットが共通認識を持ち得る社会の実現に貢献する
	ジオメトリ	• 英語で「幾何学」を意味する • 形状や、形状を定義づける頂点の座標や線分等のデータ
	自己位置推定	• ある場所にいる自分自身の位置を推定する技術 • GPSを使う手法やカメラ映像を使う手法、LiDARの超音波センサーを使用する手法などが存在する
	ジョイスティック	• スティック（レバー）を傾けることで方向入力が行える入力機器の総称
	ストリーミング	• 音声や動画などのマルチメディアファイルを転送・再生するダウンロード方式の一種 • すべてのダウンロードが完了してから再生するのではなく、ファイルのダウンロードを行いながら同時に再生を行うことで、ユーザーの待ち時間を大幅に短縮することが可能

用語集（3/4）

用語		内容
タ行	テクスチャ	• 3Dモデルの表面に貼り付ける画像 • テクスチャを使うことで、3Dモデルに質感や模様を与えることができる
	デジタルツイン	• センサで収集した現実世界の情報を用いて、デジタルな現実世界の複製を作成する概念
ナ行	NavMesh	• Unityにおける、自動移動するオブジェクトが経路を探索できる範囲
ハ行	パーソナルモビリティ	• 1人乗りの移動支援機器 • 次世代型電動車いすとも言う
	BIM	• Building Information Modellingの略 • 形状情報と仕様情報を建物の属性情報として併用して扱う設計手法・ソフトウェア • 必ずしも3D形状だけではなく2D形状も形状属性として扱う
	VPS	• Visual Positioning Systemの略 • カメラ等から得られた周囲の映像から、特徴点を抽出し、データベース上のデータと照合することで、現在位置を測定するための技術
	ポリゴンモデル	• 3つ以上の頂点を結んでできる多角形の組み合わせで、任意の3Dモデルを表現するもの
マ行	メッシュサーフェス	• 点群データを元に生成した面の集合体のこと
	モーションセンサー	• 物の加速度・傾き・方向等を検知して信号として出力する装置
	モビリティ運用・モビリティ運行	• パーソナルモビリティなどを自動にて安全に運行すること

用語集（4/4）

用語		内容
ヤ行	UE4	• Unreal Engine4の略称 • Epic Gamesによって開発・提供されている3Dゲームエンジン
	USDデータ	• Universal Scene Descriptionの略 • 任意の3Dシーンを、堅牢かつスケーラブルに交換および拡張できるようにするために Pixarによって開発されたデータ形式
	Unity	• Unity Technology社が開発・提供するゲームエンジンおよび開発環境 • スマートフォン向けのアプリや家庭用ゲーム機・ウェブなどの様々なプラットフォームで実行可能なアプリケーションを開発することができる
ラ行	LiDAR	• Light Detection And Rangingの略 • レーザー光を照射して、その反射光の情報をもとに、対象物までの距離や対象物の形などを計測する技術
	LAS	• LiDARによって取得された点群の標準フォーマット • 米国写真測量学会が制定している
	リダクション	• 点群を構成する点やメッシュデータを構成する面を間引くこと • そのままではデータが大きすぎて使えない場合に、データの軽量化を目的として行う
	リダクション処理	• 音声や映像などといった信号に含まれるノイズを抑圧・軽減するための信号処理の一種
	レイトレーシング	• 光源から出ている光の量や方向を把握し、水面や物体の表面などで起こる光の屈折や反射など現実世界で発生しうる様々な影響をコンピューターで計算し、より現実に近い映像を作り出す技術
	レンダリング	• 3Dモデルに表面仕上げの素材・光源の設定・動きを設定して、動画や静止画を計算させて生成すること • モデルやデータが多くなると計算時間は飛躍的に増大する
	ロギング	• コンピュータの動作記録やアクセス履歴などを定期的に記録すること

都市空間の統合デジタルツインの構築
技術検証レポート

令和5年3月　発行
委託者：国土交通省 都市局 都市政策課
受託者：株式会社 竹中工務店・株式会社 日立製作所・株式会社 gluon

本報告書は、株式会社 竹中工務店・株式会社 日立製作所・株式会社 gluonが国土交通省との間で締結した業務委託契約書に基づき作成したものです。受託者の作業は、本報告書に記載された特定の手続や分析に限定されており、令和5年3月までに入手した情報にのみ基づいて実施しております。従って、令和5年4月以降に環境や状況の変化があったとしても、本報告書に記載されている内容には反映されておりません。