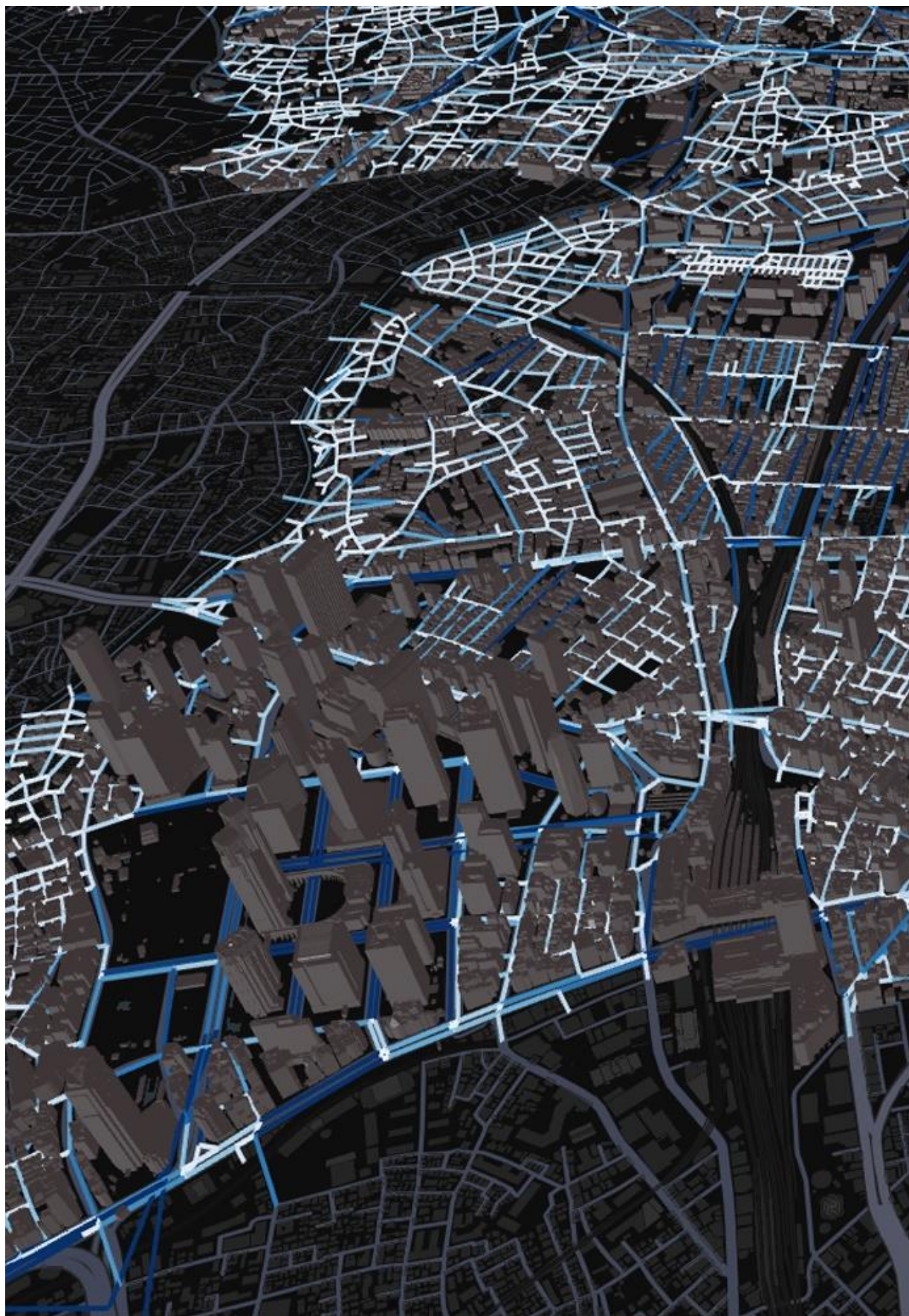




PLATEAU
by MLIT

PLATEAU Technical Report
3D都市モデル活用のための技術資料



WebGIS に関する技術調査レポート

series No. 59

Technical Report on Survey of WebGIS

目次

1. 調査の概要	- 1 -
1-1. 目的	- 1 -
1-2. 現状の課題	- 2 -
1-3. 調査のアプローチ・フロー	- 3 -
1-4. 調査のスケジュール	- 4 -
1-5. デモアプリケーションについて	- 4 -
2. 地図エンジンの調査	- 5 -
2-1. 目的	- 5 -
2-2. 調査方法	- 5 -
2-3. 調査結果のまとめ	- 6 -
2-4. 各地図エンジンの詳細	- 8 -
3. 3D 都市モデルの表示性能検証	- 38 -
3-1. 目的	- 38 -
3-2. 検証方法	- 38 -
3-3. 検証結果のまとめ	- 40 -
3-4. 各データ形式の詳細	- 41 -
3-5. 各地図エンジンの詳細	- 49 -
4. 表現方法の調査	- 87 -
4-1. 目的	- 87 -
4-2. 調査方法	- 87 -
4-3. 調査結果のまとめ	- 87 -
4-4. deck.gl の活用	- 89 -
4-5. MapLibre GL JS の活用	- 106 -
5. 成果と課題、今後の展望	- 110 -
5-1. 成果と課題	- 110 -
5-2. 今後の展望	- 111 -

1. 調査の概要

1-1. 目的

3D 都市モデルの可視化や解析等の活用を誰もが簡単に行えるようにするためには、ウェブ上で利用可能な地図を用いることが有用である。ウェブ上で利用可能な地理空間情報システムは一般に WebGIS と呼ばれる。

WebGIS は一般的には「地図エンジン」と呼ばれるライブラリを用いて構築を行う。以後、ウェブ上で利用可能な地理空間システムのことを「WebGIS」、WebGIS を構築するライブラリのことを「地図エンジン」と呼称するものとする。

現状、CesiumJS と呼ばれる地図エンジンを用いた 3D 都市モデルの可視化環境として PLATEAU VIEW が開発されている。これは、CityGML 形式で標準化されている 3D 都市モデルの形状及び属性情報をそのまま再現できるレンダリングフォーマットとして 3D Tiles 形式が採用され、その描画には地図エンジンとして Cesium エンジンが最適な選択肢とされていることによる。他方、3D モデルを使用可能な地図エンジンとしては CesiumJS 以外にも様々なライブラリやソフトウェアが存在しているが、3D 都市モデルとの親和性については明らかではない。

そこで、本調査では、3D 都市モデルを利用可能な地図エンジンを広く調査・比較し、可視化、解析、他の GIS データとの組み合わせなどの観点から 3D 都市モデルとの親和性を整理する。これらの整理を通じ、3D 都市モデルのデータとしての価値を最大限に引き出すことが可能な既存の地図エンジンを体系化することを本調査の目的とする。

1-2. 現状の課題

現行の PLATEAU VIEW で使用されている CesiumJS は、WebGL を使用した 3D 地図エンジンであり、3D 都市モデルや地形の描画に適しているが、現状では以下の課題が存在している。

- 高度な可視化表現機能の不足
 - 他の地図エンジンと比べ、CesiumJS は 3D 都市モデルや地形の精緻な表現に特化しており、様々なデータ可視化表現の機能（例としてヒートマップやクラスターといった表現）が不足している。
- パフォーマンスの悪さ
 - 前述した通り CesiumJS は精緻な表現を行うために処理の負荷が高く、アプリケーションの動作速度が低下し、ユーザーにとって利便性が損なわれることがある。
- カスタマイズ性の低さ
 - CesiumJS はレンダリングに関する処理を地図エンジン内部で行っており、アプリケーション開発者がレンダリングを直接カスタマイズすることが難しい。
 - CesiumJS 自体の改造により実現可能な事もあるが、今後の CesiumJS 自体のアップデート（例として次期バージョンの 3D Tiles への対応）に追従することが困難となる。
 - CesiumJS はオープンソースではあるが、Cesium GS, Inc.が開発を主導しており、変更や機能追加を提案することはできても、それが開発方針に沿っていなければ取り込まれない。

1-3. 調査のアプローチ・フロー

本調査では、CesiumJS 以外の様々な地図エンジンについて、「3D 都市モデルを適切に描画できるか」「パフォーマンス面など含め、実用上問題なく描画できるか」「より高度な表現が可能か」などの視点で調査を行う。なお、ここでいう「3D 都市モデル」とは、「3D 都市モデル標準製品仕様書」（国土交通省都市局）に基づき作成された CityGML 2.0 形式のデータをもとに、WebGIS での利用を想定したフォーマットに変換されたデータを指す。必ずしも 3D Tiles 形式とは限らない。

調査では、以下の三段階にフェーズを分けて進める。

- STEP1 地図エンジンの調査
 - まず、ユーザー数が一定以上あり、商用を含めた実サービスでの運用の実績がある地図エンジンの中から、3D 都市モデルを描画することが可能なものを調査し、その特色や扱うことが可能なデータ形式等を網羅的に整理する。
- STEP2 3D 都市モデルの表示性能検証
 - 次に、STEP1 で把握した地図エンジンを利用し、実際に 3D 都市モデルの可視化環境を構築することで、実運用上の 3D 都市モデルの表示パフォーマンスを確認する。この観点から各地図エンジンのメリット・デメリットの整理などを行う。
- STEP3 表現方法の調査
 - 最後に、STEP2 で実運用上利用可能と評価された地図エンジンを用い、CesiumJS では難しい高度な表現方法が可能かを調査する。このため、実際に実証用のアプリケーション環境やデータを準備する。この調査を通じて、各地図エンジンの特徴や可能な表現手法を整理し、CesiumJS との比較を行う。

なお、本調査では以下の実証環境を用意した。

- MacBook Pro, Apple M1 Pro
- メモリー16GB
- macOS Ventura 13.4
- Google Chrome v122 (arm64)

1-4. 調査のスケジュール

表 1-1 スケジュール

実施事項	令和 5 年度										令和 6 年度		
	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. STEP1 地図エンジンの調査		←→											
2. STEP2 3D 都市モデルの表示性能 検証			←→→→										
3. STEP3 表現方法の調査						←→→→→→→→							
4. 成果とりまとめ										←→			

1-5. デモアプリケーションについて

本調査で作成したデモアプリケーションは、以下の URL から閲覧することができる。

<https://project-plateau.github.io/mapengine-survey/>

2. 地図エンジンの調査

2-1. 目的

3D 都市モデルを表示する際、主に用いられている CesiumJS 以外に、どのような選択肢があるのか、それぞれにどのような特性があるかを把握することで、より具体的な調査・実証を行うべき対象を選定する。

2-2. 調査方法

ここでは「地図エンジンの調査」として、主に 3D 都市モデルを適切に可視化可能か、という視点で、各種の地図エンジンやライブラリを網羅的に調査し、主な機能や表示可能なデータ形式などを合わせて整理した。その上で、STEP2「3D 都市モデルの表示性能検証」の対象を選定する。

選定に際しては、

- 3D 都市モデルの描画機能
- ユーザー数や、開発の活発さ、ドキュメントの充実度
- 高度な可視化などの追加機能

などといった観点を含め、総合的に判断した。

2-3. 調査結果のまとめ

表 2-1 3D 都市モデルのための地図エンジンの比較

	オープン ソース	人気度 GitHub star	主なデータ形式	備考
CesiumJS	○	11.7k	3D Tiles	PLATEAU VIEW 3.0 で利用
MapLibre GL JS	○	5.6k	MVT	Mapbox GL JS のフォーク
Mapbox GL JS (v2)	X	10.7k	MVT	要アクセストークン
deck.gl	○	11.6k	3D Tiles, MVT	多種の可視化機能
ArcGIS Maps SDK	X	-	I3S, MVT	要ライセンス
iTowns	○	1.0k	3D Tiles, MVT	Three.js ベース
3DTilesRendererJS	○	1.4k	3D Tiles	Three.js ベース
maprayJS	○	0.1k	独自	要クラウドサービス利用
Tangram	○	2.2k	MVT	開発停止
maplibre-rs	○	1.2k	MVT	実験的実装
SuperMap iClient	○	0.8k	3D Tiles, MVT	各種地図エンジンのラッパー
TerriaJS	○	1.1k	3D Tiles	CesiumJS を利用
3D BAG	X	-	-	データセットがメイン

※GitHub star 数は 2024 年 3 月時点のもの

各種の地図エンジンを検討した結果、以下 6 つの地図エンジンを STEP2 での技術調査対象として選定した。

- MapLibre GL JS
 - オープンソースであり、開発が活発。MVT による軽快な描画や、ヒートマップやクラスター表示などの高度な可視化表現機能を備える
- Mapbox GL JS (v2)
 - MapLibre GL JS と同等の機能に加え、スカイレイヤーや多種の地図投影法といった独自の機能を備える
- deck.gl
 - オープンソースであり、開発が活発。3D Tiles をはじめとした各種データの描画が可能であり、他にはない可視化機能を多数備える
- ArcGIS Maps SDK for JavaScript
 - I3S による描画が可能であり、利用者が多くドキュメントが従実しており、また他にはない可視化機能を備える
- iTowns

WebGIS に関する技術調査レポート

- 3D Tiles や MVT の描画が可能で、他とは異なる方向性での開発が進められている
- 3DTilesRendererJS
 - Three.js での 3D Tiles 描画を可能とするものであり、Three.js をもとにした利用や開発の可能性が考えられる

他方で、以下の地図エンジンを技術選定の対象外とした。

- maprayJS
 - 独自のデータ形式を採用しており、利用にはベータ版のクラウドサービスが必須である
- Tangram
 - すでに開発が停止しており今後の発展やメンテナンスが望めず、そのデメリットを上回る独自の機能がない
- maplibre-rs
 - 実証実験的な実装であり実用に足る機能が不足している
- SuperMap iClient-JavaScript
 - 他の地図エンジンを内包したクラウドサービスのクライアントが実装されており、独自の描画機能がない
- TerriaJS
 - CesiumJS を地図エンジンとして使用しており、描画機能自体の差がない
- 3D BAG
 - データセットであり、利用されているビューアー実装は一般公開されていない

なお、Mapbox GL JS は、2023 年 8 月に v3 のパブリックプレビューが発表され、動的なライティングや、より高い LOD でのランドマークの 3D 表現といった新たな機能が導入された。当調査の実施時に v3 はプレビュー段階であり、またランドマークは Mapbox が提供するものであってユーザーが作成・追加することはできないため、今回の趣旨と異なる。そのため、STEP2 の対象から除外した。

- Mapbox Standard Style に関する Blog 記事, <https://www.mapbox.com/blog/standard-core-style>
- MapBox GL JS v3 移行ガイド, <https://docs.mapbox.com/mapbox-gl-js/guides/migrate-to-v3/>

2-4. 各地図エンジンの詳細

2-4-1. CesiumJS

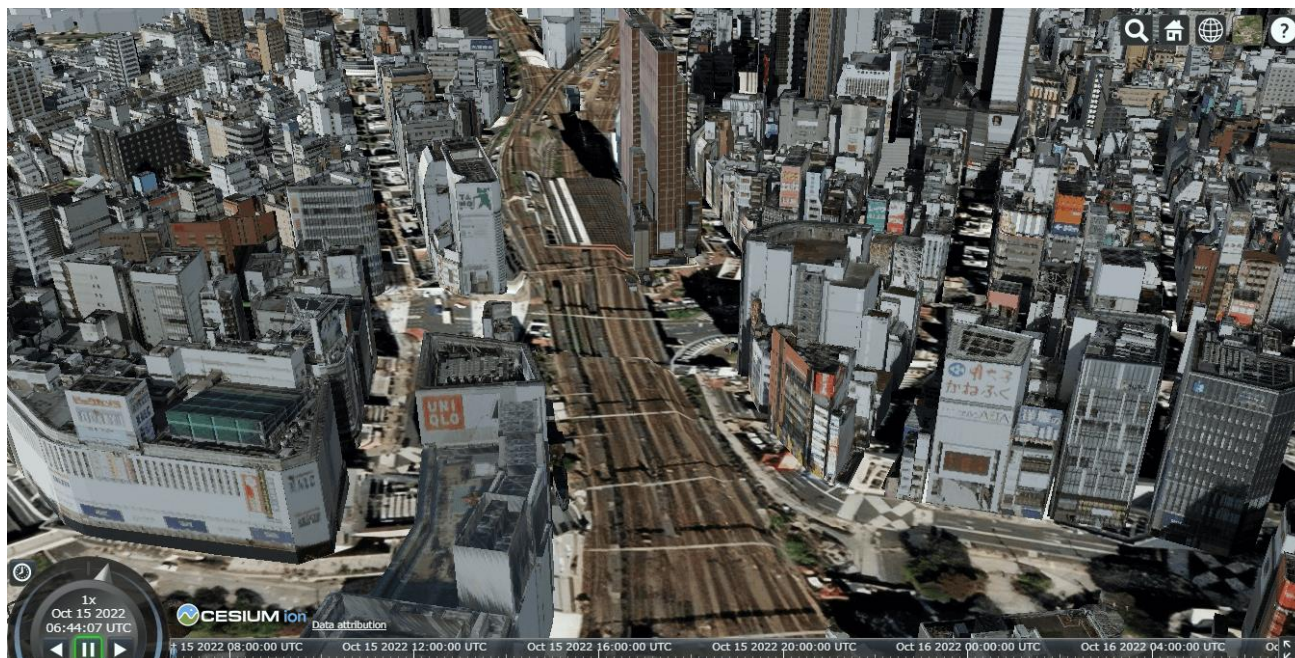


図 2-1 CesiumJS による 3D 都市モデル描画の例

<https://www.mlit.go.jp/plateau/learning/tpc06-1/>

2-4-1-a. 概要

3D 地球表面上を可視化するための、Cesium GS, Inc.が開発を主導するオープンソース JavaScript ライブラリ。WebGL を使用して高性能に地理空間情報をレンダリングすることが可能である。

2-4-1-b. 主な機能

- 3D 地形の描画
- 3D Tiles を用いた 3D 都市モデルの表示
- ラスタータイルの表示

2-4-1-c. 考察

3D Tiles の描画におけるデファクトスタンダードであり、3D 都市モデルや地形の描画に最適である。PLATEAU VIEW (v1.0、v2.0、v3.0) でも地図エンジンとして採用されている。

WebGIS に関する技術調査レポート

しかし、CesiumJS はレンダリングに関する部分の処理を内部で行っており、アプリケーション開発者がレンダリングを直接カスタマイズすることが難しい。具体的には、ヒートマップやクラスターなどの表現を CesiumJS で実現することは難しい。また、CesiumJS 自体の改造により実現可能な事もあるが、今後の CesiumJS 自体のアップデートに追従することが困難になるという大きなデメリットがある。

また、近年一般的な背景地図で用いられるベクタータイルを CesiumJS では取り扱うことができない。

なお、国土交通省都市局が開発した「PLATEAU VIEW 3.0」では、CesiumJS の内部実装を非公開 API を用いて上書きすることで、通常の CesiumJS では不可能な多彩な表現を可能とし、前述の一部課題の解決を図っている。

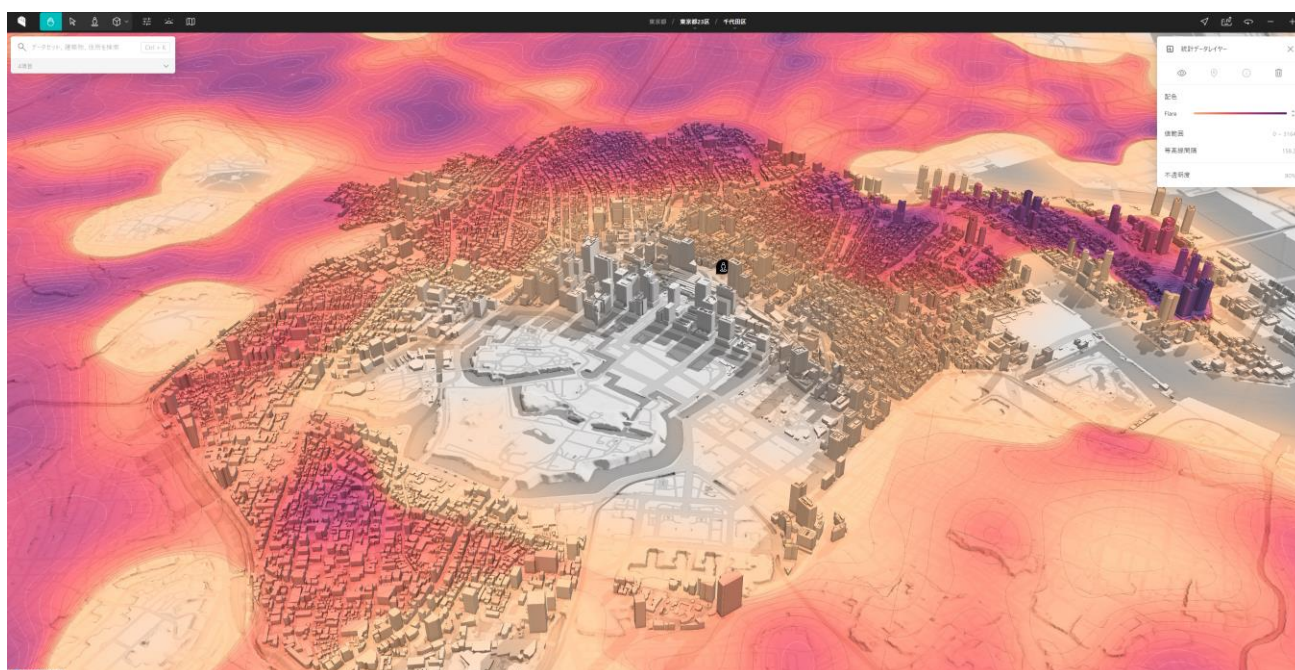


図 2-2 PLATEAU VIEW 3.0

2-4-1-d. 参考文献

表 2-2 CesiumJS 参考文献

タイトル	出典
CesiumJS 公式サイト	https://cesium.com/platform/cesiumjs/
PLATEAU VIEW 3.0	https://plateauview.mlit.go.jp/

2-4-2. MapLibre GL JS

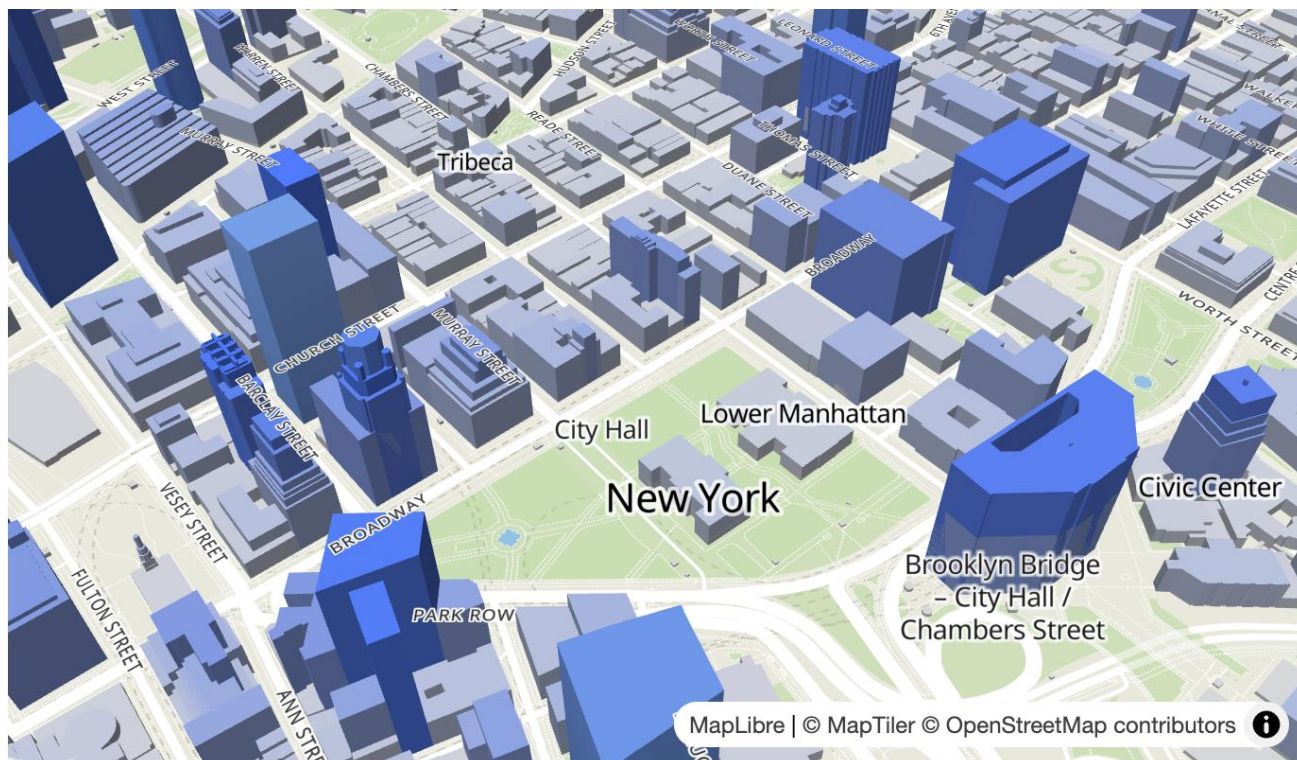


図 2-3 MapLibre GL JS による 3D 表現の例

<https://maplibre.org/maplibre-gl-js/docs/examples/3d-buildings/>

2-4-2-a. 概要

WebGL を活用した、Web 上で地図を表示するための JavaScript ライブラリ。後述する Mapbox 社が開発した Mapbox GL JS のライセンス変更を受けて 2020 年 12 月にフォークされたオープンソースソフトウェアである。

2-4-2-b. 主な機能

- ベクタータイル、ラスタータイルの表示
- 3D 地形の描画
- Extrusion（押出）による 3D 表現
- glTF の描画（Three.js を用いたカスタムレイヤーによる）
- ヒートマップ、クラスター表示

3D 地形が表示可能であるが、CesiumJS の様にジオイド高は考慮されていない。



図 2-4 MapLibre GL JS - 地形表現

<https://maplibre.org/maplibre-gl-js/docs/examples/3d-terrain/>

参考: Mapbox GL JS (v2) との差異

後述する Mapbox GL JS (v2) と比較して、MapLibre GL JS のみに存在する機能としては、タイルデータの動的加工がある。動的加工とは読み込んだタイルデータへリアルタイムに処理を施し変換するものである。動的加工を行うために、addProtocol という機能が存在する。addProtocol 機能は、データソースに対して独自の振る舞いを定義する機能である。例えば、国土地理院の標高タイルを TerrainRGB 形式に変換するといった事例がある。

2-4-2-c. 考察

MapLibre GL JS は、以下の理由から、STEP2 の調査対象として適切である。

- 3D 都市モデルの描画機能
 - MVT 押出やカスタムレイヤーによる glTF を用いた 3D 表現が可能である
- ユーザー数や、開発の活発さ、ドキュメントの充実度
 - オープンソースソフトウェアである
 - ユーザー数が多く、開発が持続的で活発である

- 高度な可視化などの追加機能
 - ヒートマップやクラスターなど、CesiumJS では容易ではない種類の可視化が存在する

2-4-2-d. 参考文献

表 2-3 MapLibre GL JS 参考文献

タイトル	出典
MapLibre GL JS ドキュメント	https://maplibre.org/maplibre-gl-js/docs/
MapLibre GL JS addProtocol ドキュメント	https://maplibre.org/maplibre-gl-js/docs/API/functions/addProtocol/
MapLibre GL JS と地理院標高タイルで 3D 地形を表示する事例	https://qiita.com/Kanahiro/items/1e9c1a4ad6be76b27f0f

表 2-4 MapLibre GL JS 参考文献 - 3D Tiles との併用事例

タイトル	出典
How to use MapLibre GL JS with 3D Tiles (pnts) - DEV Community	https://dev.to/hfu/how-to-use-maplibre-gl-js-with-3d-tiles-pnts-g4h

2-4-3. Mapbox GL JS (v2)



図 2-5 Mapbox GL JS による 3D 表現の例

<https://docs.mapbox.com/mapbox-gl-js/example/3d-buildings/>

2-4-3-a. 概要

WebGL を活用した、Web 上で地図を表示するための JavaScript ライブラリ。Mapbox 社が開発した Mapbox GL JS は、v1 ではオープンソースであったが、v2 からはライセンス変更により非オープンソース化した。なお先述した MapLibre GL JS は、この Mapbox GL JS v1 からフォークされたオープンソースソフトウェアのプロジェクトである。

2-4-3-b. 主な機能

- ベクタータイル、ラスタータイルの表示
- 3D 地形の描画
- Extrusion（押出）による 3D 表現

WebGIS に関する技術調査レポート

- glTF の描画（Three.js を用いたカスタムレイヤーによる）
- ヒートマップ、クラスター表示

3D 地形が表示可能であるが、CesiumJS の様にジオイド高は考慮されていない。

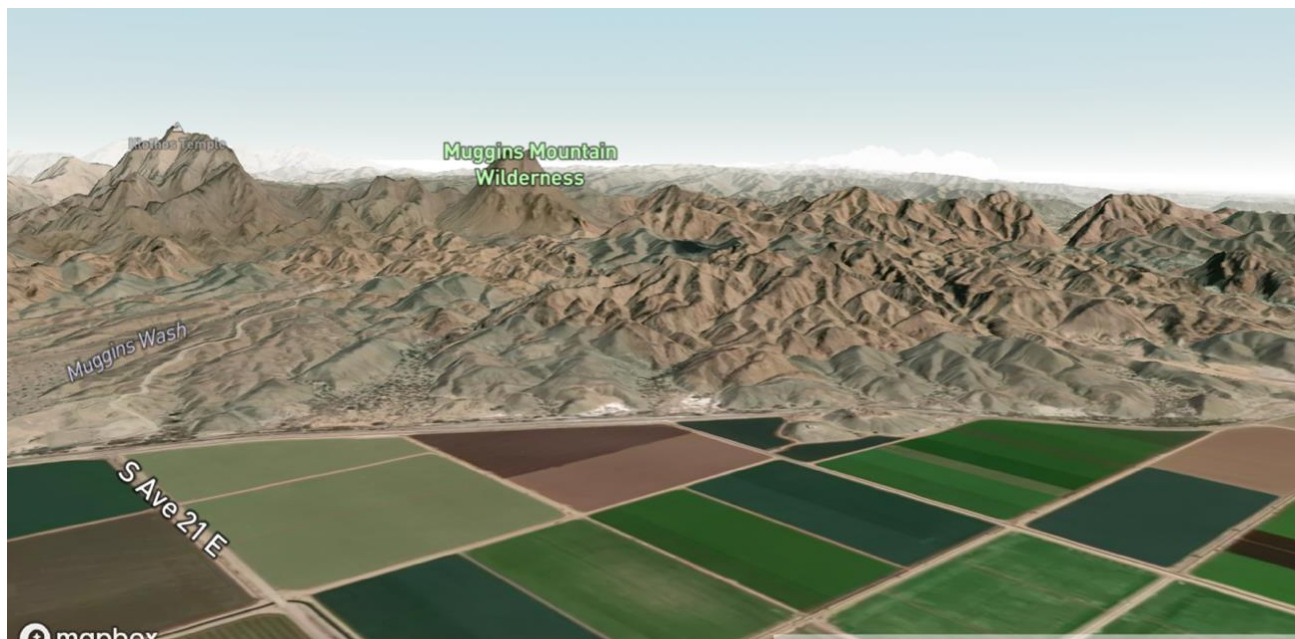


図 2-6 Mapbox GL JS (v2) - 地形表示

<https://docs.mapbox.com/mapbox-gl-js/example/add-terrain/>

参考: MapLibre GL JS との差異

MapLibre GL JS と比較して、Mapbox GL JS (v2) のみに存在する機能として、スカイレイヤーでの天気や霧の表現、メルカトル以外の地図投影法（地球儀ビューなど）がある。



図 2-7 Mapbox GL JS (v2) - スカイレイヤー霧の表現

<https://docs.mapbox.com/mapbox-gl-js/example/add-fog/>



図 2-8 Mapbox GL JS (v2) - メルカトル以外の地図投影

<https://docs.mapbox.com/mapbox-gl-js/example/globe/>

2-4-3-c. 考察

以下の理由から、Mapbox GL JS (v2) は、STEP2 の調査対象として適切である。

- 3D 都市モデルの描画機能
 - MVT 押出やカスタムレイヤーによる glTF を用いた 3D 表現が可能である
- ユーザー数や、開発の活発さ、ドキュメントの充実度
 - オープンソースソフトウェアではなく利用にトークンが必要だが、MVT を用いる際のデファクトスタンダードであり、ユーザー数が多く、開発が持続的で活発である
- 高度な可視化などの追加機能
 - ヒートマップやクラスターなど、CesiumJS では容易ではない種類の可視化が存在する
 - 加えて、MapLibre GL JS にはないスカイレイヤーや地球儀ビューといった機能を有する

2-4-3-d. 参考文献

表 2-5 Mapbox GL JS (v2) 参考文献

タイトル	出典
Mapbox GL JS	https://www.mapbox.com/mapbox-gljs

表 2-6 Mapbox GL JS (v2) 参考文献 - 3D Tiles との併用事例

タイトル	出典
Geodan/mapbox-3dtiles	https://github.com/Geodan/mapbox-3dtiles
2019 - 3D geo data in the Mapbox-gl viewer with 3D tiles (FOSS4G 2019)	https://www.youtube.com/watch?v=HXQJbyEnC9w
Deckgl, mapbox and 3D tiles	https://observablehq.com/@tomvantilburg/deckgl-mapbox-and-3d-tiles
How to use MapLibre GL JS with 3D Tiles (pnts) - DEV Community	https://dev.to/hfu/how-to-use-maplibre-gl-js-with-3d-tiles-pnts-g4h

2-4-4. deck.gl

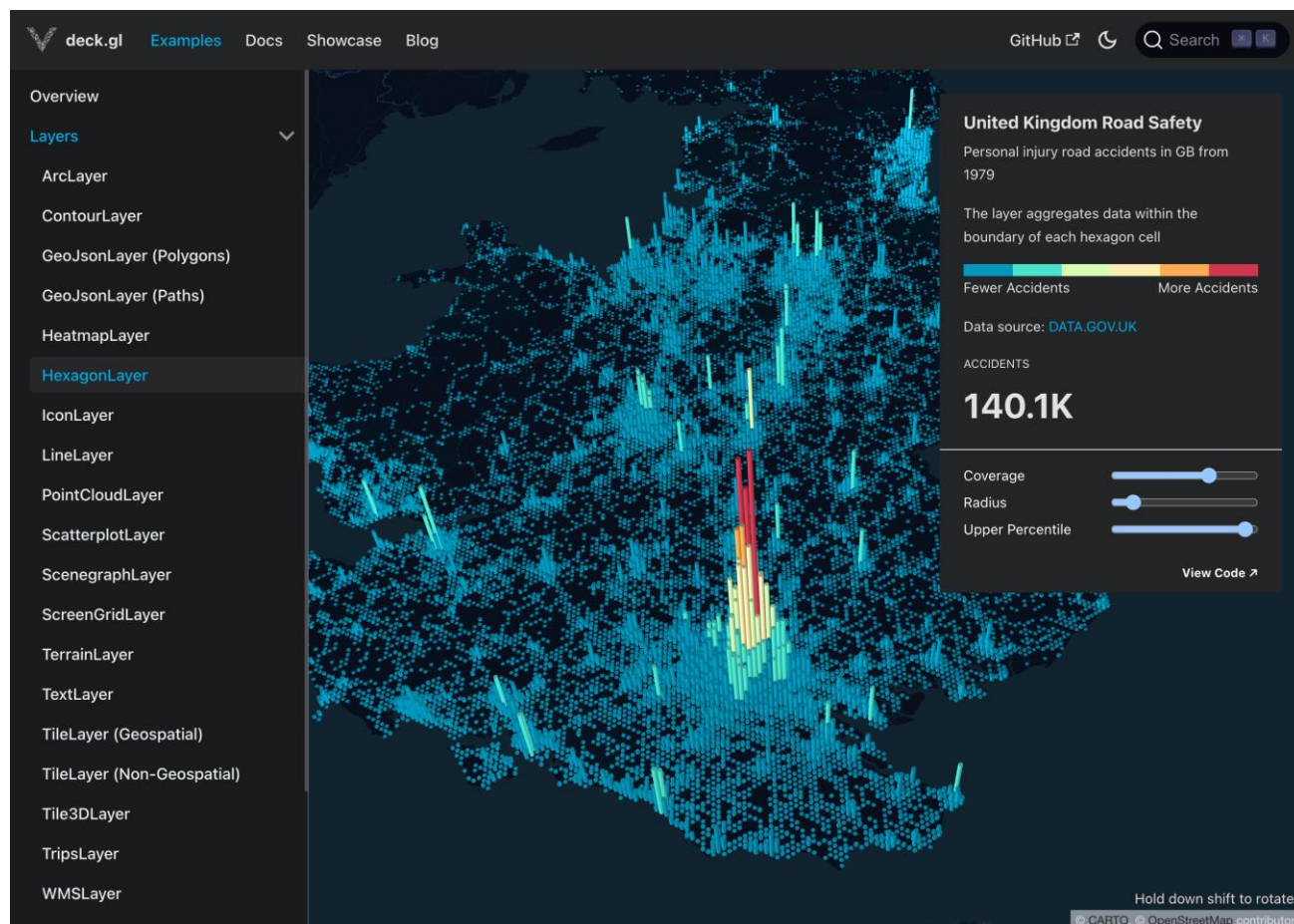


図 2-9 deck.gl によるデータ可視化の例 (HexagonLayer)

<https://deck.gl/examples/hexagon-layer>

2-4-4-a. 概要

WebGL を用いた高パフォーマンスの大規模データ可視化ライブラリ。Uber 社が開発を開始し、現在は OpenJS 財団下で、地理空間情報可視化ツール群「vis.gl」の中心として開発される。Unfolded 社（2021 年 5 月 Foursquare が買収）と CARTO 社が、リード開発者として主導している。

2-4-4-b. 主な機能

- ベクタータイル、ラスタータイルの表示
- 3D 地形の描画
- さまざまなデータ形式への対応

- loaders.gl というパーサー・エンコーダー群ライブラリを用いており、3D Tiles や I3S といった形式の描画が可能
- MapLibre GL JS、Mapbox GL JS (v2) と組み合わせて利用することが可能
- Web フレームワーク React との高い親和性
- 多様なデータ可視化表現を可能とするレイヤー



図 2-10 deck.gl – 公式サンプルのサムネイル

<https://deck.gl/examples>

2-4-4-c. 考察

Mapbox GL JS (v2) は、以下の理由から、STEP2 の調査対象として適切である。

- 3D 都市モデルの描画機能
 - 3D Tiles をはじめ多数のデータ形式を描画することが可能
- ユーザー数や、開発の活発さ、ドキュメントの充実度
 - オープンソースソフトウェアである
 - ユーザー数が多く、開発が持続的で活発である

- 高度な可視化などの追加機能
 - MapLibre GL JS などと組み合わせても使うことが可能である
 - 他にはない多様なデータ可視化表現を可能とする機能が存在する

2-4-4-d. 参考文献

表 2-7 deck.gl 参考文献

タイトル	出典
deck.gl 公式ページ	https://deck.gl/
loaders.gl 公式ページ	https://loaders.gl/
deck.gl の Tile3dLayer で 3D 都市モデルを表示している記事	https://qiita.com/Kanahiro/items/4ae702c575176501a157
3D モデルを 3DTiles に変換して deck.gl で表示している記事	https://qiita.com/waigania13/items/ce42ec615938386adcf5
deck.gl でカスタムシェーダーを使った記事	https://docs.google.com/presentation/u/2/d/e/2PACX-1vScyxLUbvTD4AQFMm6mbMSunxKjMRM7XzgAFjsHuzORvNMG-w8kQLfzcsxUXIkhk-XH1jUVU4rqld2j/pub?start=false&loop=false&delayms=3000
あなたの街の地価は？ 上昇・下落マップ 2023：日本経済新聞	https://vdata.nikkei.com/newsgraphics/land-value-map/

2-4-5. ArcGIS Maps SDK for JavaScript



図 2-11 Arakawa Digital Twin online - 荒川 3D 河川管内図 (下流域)

<https://www.esri.com/news/details/143571/>

2-4-5-a. 概要

ArcGIS を提供する Esri 社が開発する JavaScript ライブラリ。同社によるプロプライエタリなサービス群と組み合わせて使うためのものであり、単独で用いることはできない。利用にはライセンスが必要である。

2-4-5-b. 主な機能

- ベクタータイル、ラスタータイルの表示
- 3D 地形の描画
- I3S データ形式による 3D 都市モデルの描画
- 多種の可視化などを可能にするプラグイン

2-4-5-c. 考察

ライセンスが必要というハードルはあるが、ArcGIS Maps SDK for Javascript は、以下の理由から、STEP2 の調査対象として適切である。

- 3D 都市モデルの描画機能
 - I3S データ形式による描画が可能
- ユーザー数や、開発の活発さ、ドキュメントの充実度
 - オープンソースソフトウェアではないが、ユーザー数が多く、ドキュメントが豊富で、開発が持続的で活発である
- 高度な可視化などの追加機能
 - 高度なデータ可視化表現をはじめとする多数のプラグインが提供されている

2-4-5-d. 参考文献

表 2-8 ArcGIS 参考文献

タイトル	出典
ArcGIS Maps SDK for JavaScript 製品紹介ページ	https://www.esri.com/products/arcgis-maps-sdk-for-javascript/
国土交通省 3D 都市モデル「Project PLATEAU」の活用事例紹介	https://3d-city-model.esri.com/
3D 都市モデル (CityGML 形式) データ変換ツール for ArcGIS、GitHub ページ	https://github.com/EsriJapan/3D-CityModel-ConversionTools-for-ArcGIS
事例、久屋大通 3D ビューア	https://ej-nagoya.maps.arcgis.com/apps/instant/3dviewer/index.html?appid=9fc2541205764de39961dae299a505d6
事例、街並みデザイン	https://experience.arcgis.com/experience/72365dad06a54dad8441b1e2b77db773?data_id=dataSource_1-180649c87f1-layer-41-associated_data_source%3A92
事例、PLATEAU (v2) 四日市市	https://ej-smart-city.maps.arcgis.com/apps/instant/3dviewer/index.html?appid=5d8a7a0474904d0984fefe5532c7ef60
事例、太陽光パネル反射シミュレーション	https://experience.arcgis.com/experience/476fd06890174b74a0ad3a71d7ba0d7f
事例、浸水による建物影響を視覚化	https://storymaps.arcgis.com/stories/f581ea400ad64c0fba627d3e0795164a
事例、可視解析	https://storymaps.arcgis.com/stories/6a32b2adb40f4d3ba35aa08455ab45f4
事例、都市の風況シミュレーション (ArcGIS エクステンション「Airflow Analyst」)	https://storymaps.arcgis.com/stories/597bf478b0504fb6894648d5840a9b98

2-4-6. iTowns

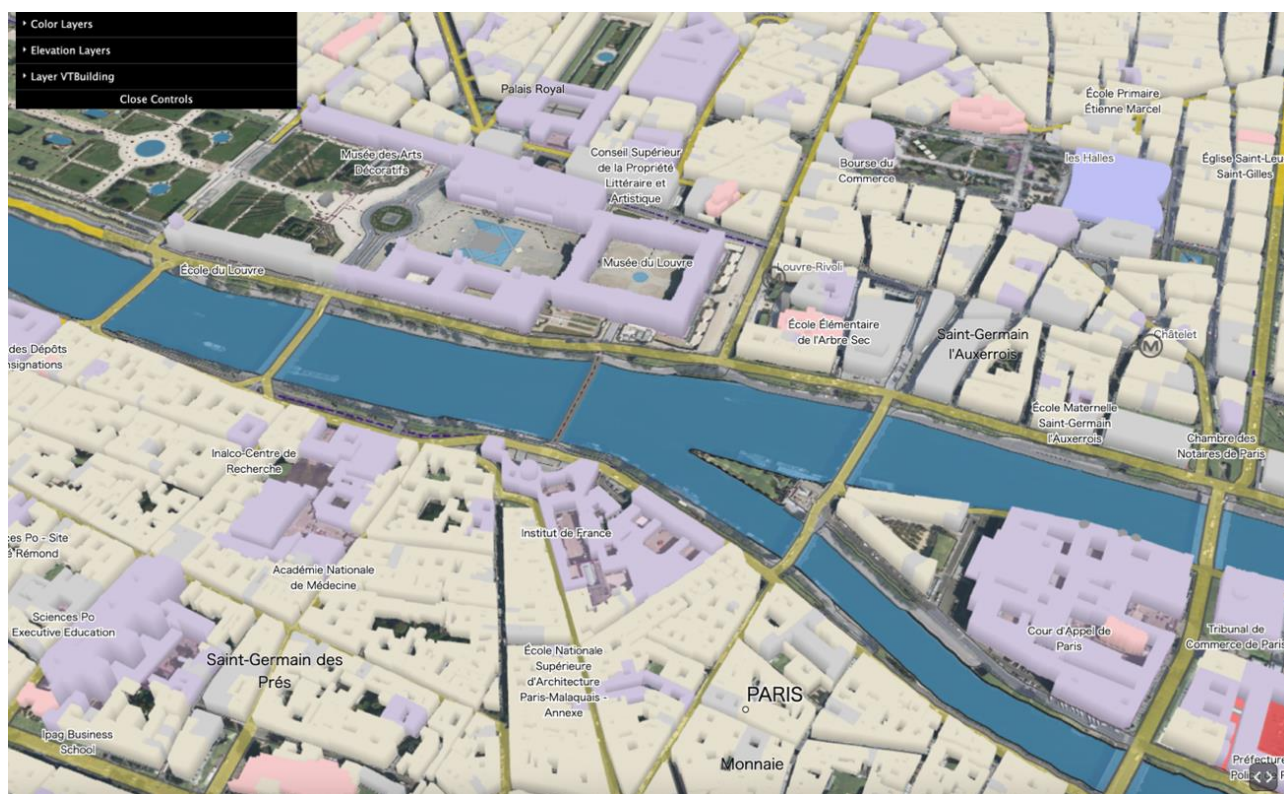


図 2-12 iTowns - 建築物モデル表示（フランス・パリ、ルーブル美術館付近に拡大した状態）

https://www.itowns-project.org/itowns/examples/#view_3d_map

2-4-6-a. 概要

Three.js をベースとしたオープンソースの地図エンジン。フランスの国立地理情報・森林情報庁（IGN, Institut national de l'information géographique et forestière）が開発を始め、現在は IGN（<http://www.ign.fr/>）と Ciril GROUP（<https://www.cirilgroup.com/en/>）によりメンテナンスされている。

2-4-6-b. 主な機能

- ベクタータイル、ラスタータイルの表示
- 3D 地形の描画
- 3D Tiles データ形式による 3D 都市モデルの描画

2-4-6-c. 考察

iTowns は、以下の理由から、STEP2 の調査対象として適切である。

- 3D 都市モデルの描画機能
 - 3D Tiles による 3D 表現が可能である
- ユーザー数や、開発の活発さ、ドキュメントの充実度
 - オープンソースソフトウェアであり、ユーザー数は多いとは言えないが、フランスの公的機関などが開発に参加している

2-4-6-d. 参考文献

表 2-9 iTowns 参考文献

タイトル	出典
iTowns 公式ページ	https://www.itowns-project.org/
iTowns GitHub ページ	https://github.com/iTowns/itowns
3次元及び4次元 WebGIS のための オープンツール紹介、情報通信研究 機構&GEO ソリューションズ	https://testbed.nict.go.jp/bunkakai/pdf/da-v_tf-01-02.pdf

2-4-7. 3DTilesRendererJS

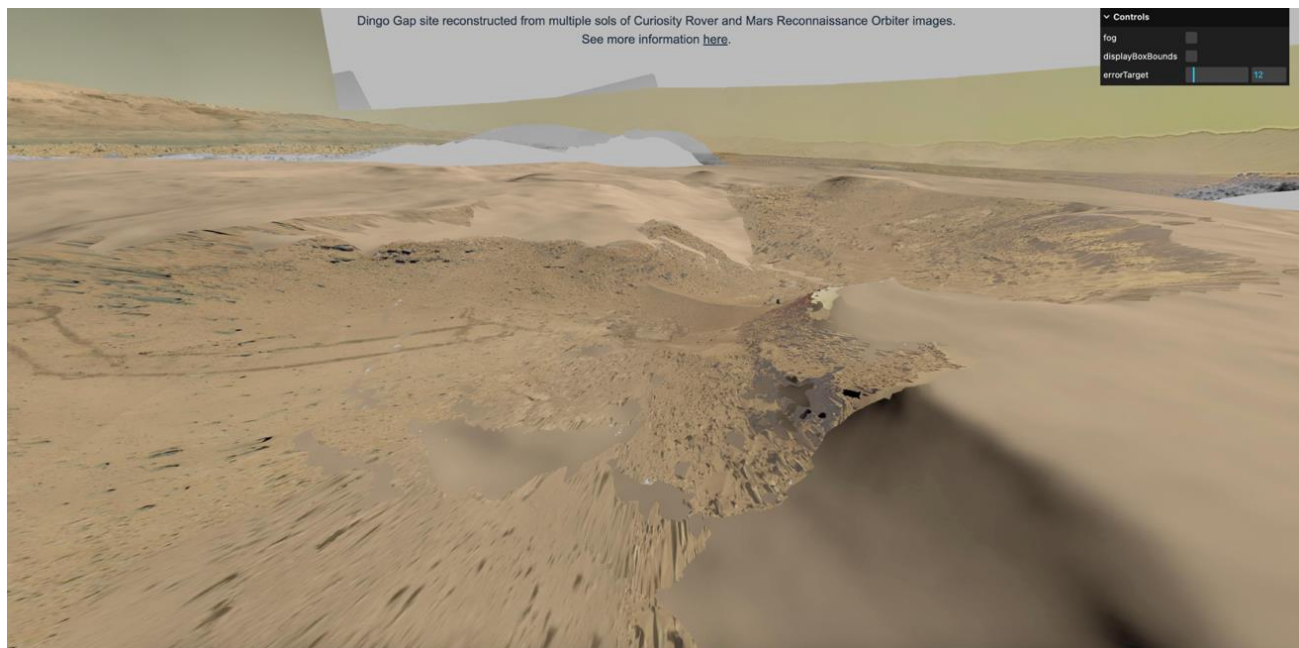


図 2-13 3DTilesRendererJS - 表示例

<https://nasa-ammos.github.io/3DTilesRendererJS/example/bundle/mars.html>

2-4-7-a. 概要

Three.js をもとにした 3D Tiles のレンダラー実装。NASA の機関（AMMOS, <https://ammos.nasa.gov/>）が開発したもので、オープンソースとして公開されている。

2-4-7-b. 主な機能

- 3D Tiles の読込、表示

2-4-7-c. 考察

3DTilesRendererJS は、以下の理由から、STEP2 の調査対象として適切である。

- 3D 都市モデルの描画機能
 - 3D Tiles データ形式による描画が可能
- ユーザー数や、開発の活発さ、ドキュメントの充実度
 - オープンソースソフトウェアである
 - 公的機関が開発を主導している

WebGIS に関する技術調査レポート

- 高度な可視化などの追加機能
 - デフォルトでは機能が用意されていないが、Three.js をもとにした利用や開発の可能性が考えられる

2-4-7-d. 参考文献

表 2-10 3DTilesRendererJS 参考文献

タイトル	出典
3DTilesRendererJS 公式ページ	https://nasa-ammos.github.io/3DTilesRendererJS/

2-4-8. maprayJS

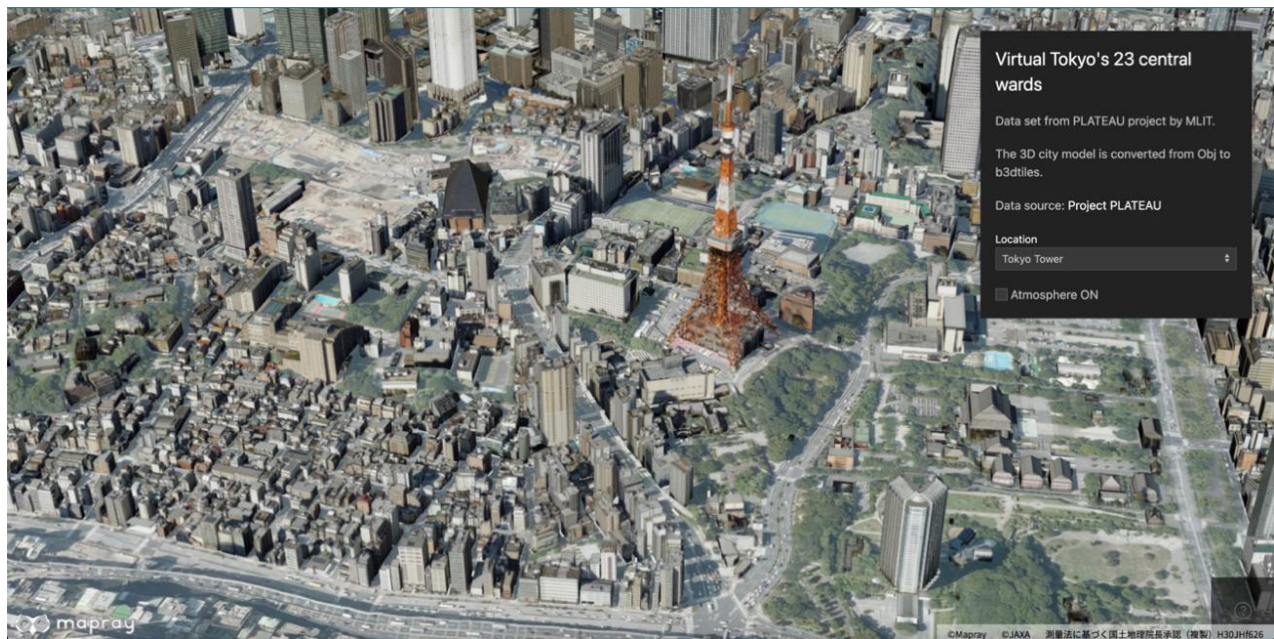


図 2-14 maprayJS - 表示例

<https://mapray.com/examples/b3dtiles/index.html>

2-4-8-a. 概要

SONY が開発するオープンソースの地図エンジン。JavaScript SDK の maprayJS と、それに対応するデータ生成・保管・配信のためのクラウドサービス Mapray Cloud がある。

2-4-8-b. 主な機能

- ベクタータイル、ラスタータイルの表示
- 3D 地形の描画
- 独自データ形式（b3dtiles）による 3D 都市モデルの描画
 - 最適化されており軽量に動作
 - b3dtiles へ変換するためのオープンな実装がなく、クラウドサービスの利用が必須
 - クラウドサービスは元データ形式として 2D データ（GeoJSON）、3D データ（glTF、OBJ）、点群データ（LAS、LAZ、ply、xyz）、建物データ（GML 及びテクスチャ）に対応

2-4-8-c. 考察

maprayJS は、以下の理由から、STEP2 の調査対象として適切ではない。

- ビューアーはオープンソースソフトウェアだが、データ利用には専用クラウドサービスが必須
- ユーザーベースが小規模

2-4-8-d. 参考文献

表 2-11maprayJS 参考文献

タイトル	出典
Mapray 公式ページ	https://mapray.com/
mapray-js GitHub ページ	https://github.com/sony/mapray-js
Mapray 事例紹介	https://mapray.com/documents/examples/index.html
Mapray 紹介 YouTube 動画	https://www.youtube.com/watch?v=03bPTci7bJI

2-4-9. Tangram

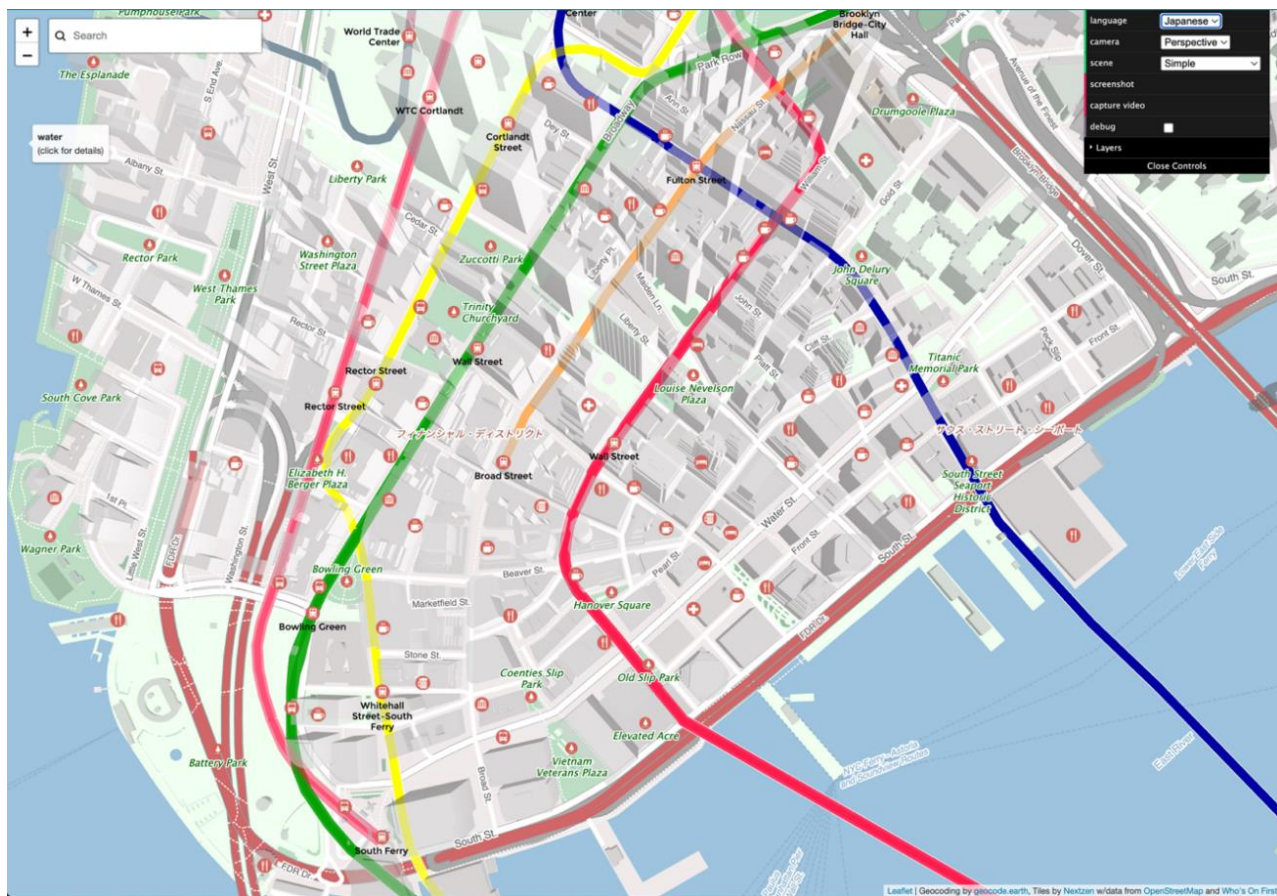


図 2-15 Tangram - 表示例

<https://demo.tangram.city/>

2-4-9-a. 概要

オープンソースの地図エンジン。Leaflet のプラグインとして利用可能であり、あわせてネイティブモバイルバージョンの Tangram ES も存在する。Mapzen 社により開発されていたが、同社の解散に伴い、現在は Linux 財団下に置かれている。2021 年 2 月以降、更新がされていない。

2-4-9-b. 主な機能

- 地図や建築物モデルの表示
- GLSL（シェーディング言語）を直接記述可能
- 対応データ
 - ベクター（MVT、GeoJSON、TopoJSON）
 - ラスター（JPEG、PNG など）

2-4-9-c. 考察

Tangram は、以下の理由から、STEP2 の調査対象として適切ではない。

- 2021 年から開発が停止しており、今後の発展やメンテナンスが期待できない
- 2D データの描画に特化しており、3D データに対する取り扱いが難しい
 - 3D 表現は出来るが、2D データの 3D 表現は自分でプログラミングを行う必要があり、プログラミングも低レイヤー言語である GLSL での記述が必要なため、描画処理の共通化もできない

2-4-9-d. 参考文献

表 2-12 Tangram 参考文献

タイトル	出典
tangram GitHub ページ	https://github.com/tangrams/tangram
tangram-es GitHub ページ	https://github.com/tangrams/tangram-es
tangram Docs Demos	https://tangrams.readthedocs.io/en/latest/Tutorials/Demos/
tangram で WEB GLSL 地図入門 Qiita 記事	https://qiita.com/northprint/items/00eee5ee48007f794de7
tangram で WEB GLSL 地図入門 その 2 Qiita 記事	https://qiita.com/northprint/items/cc8bb2cbdee6b74a6f3b
マッピング業界の繁栄と、あるスタートアップの死——「Mapzen」は、なぜ閉鎖に追い込まれたのか 記事	https://wired.jp/2018/01/16/mapzen-shuts-down/

2-4-10. maplibre-rs



図 2-16 maplibre-rs 表示例

<https://maplibre.org/news/2022-12-18-maplibre-rs-monthly/>

2-4-10-a. 概要

現行の MapLibre GL JS をベースとしていない次世代の地図エンジンプロジェクト。WebGL 後に制定されたウェブ標準である WebGPU をもとにした実験的な実装で、開発途上である。Web、モバイル及びデスクトップを対象としている。プログラミング言語 Rust が用いられている。

2-4-10-b. 主な機能

- ベクタータイルの表示

2-4-10-c. 考察

maplibre-rs は、以下の理由から、STEP2 の調査対象として適切ではない。

- 機能の不足
 - あくまでも概念実証の開発途上制作物であり、単純な描画程度しか実装されていない
 - テキストやラベルのレンダリング、ラスターデータの描画、3D 地形の描画といった重要な機能が欠けている

2-4-10-d. 参考文献

表 2-13 maplibre-rs 参考文献

タイトル	出典
maplibre-rs GitHub ページ	https://github.com/maplibre/maplibre-rs
maplibre-rs ポータブル地図レンダラーに向けて	https://isprs-archives.copernicus.org/articles/XLVIII-4-W1-2022/35/2022/

2-4-11. SuperMap iClient-JavaScript



図 2-17 SuperMap - 建築物モデル表示例

<https://iclient.supermap.io/>

2-4-11-a. 概要

SuperMap Software 社が提供するクラウドサービスを利用するための、各種地図エンジンを内包した JavaScript クライアント。クライアント実装自体はオープンソースである。

2-4-11-b. 主な機能

- CesiumJS、MapLibre GL JS、Mapbox GL JS、OpenLayers、Leaflet を内包
- SuperMap Software 社の各種クラウドサービスへ接続可能

2-4-11-c. 考察

SuperMap iClient-JavaScript は、以下の理由から、STEP2 の調査対象として適切ではない。

- 他の地図エンジンを内包したクライアントであり、描画能力に独自の特徴がない
- 中国企業が開発しており、ドキュメントが中国語であるなど取り扱いに難がある

2-4-11-d. 参考文献

表 2-14 SuperMap iClient-JavaScript 参考文献

タイトル	出典
SuperMap 公式ページ	https://www.supermap.com/
SuperMap GitHub ページ	https://github.com/SuperMap
SuperMap/iClient-JavaScript GitHub ページ	https://github.com/SuperMap/iClient-JavaScript

2-4-12. TerriaJS

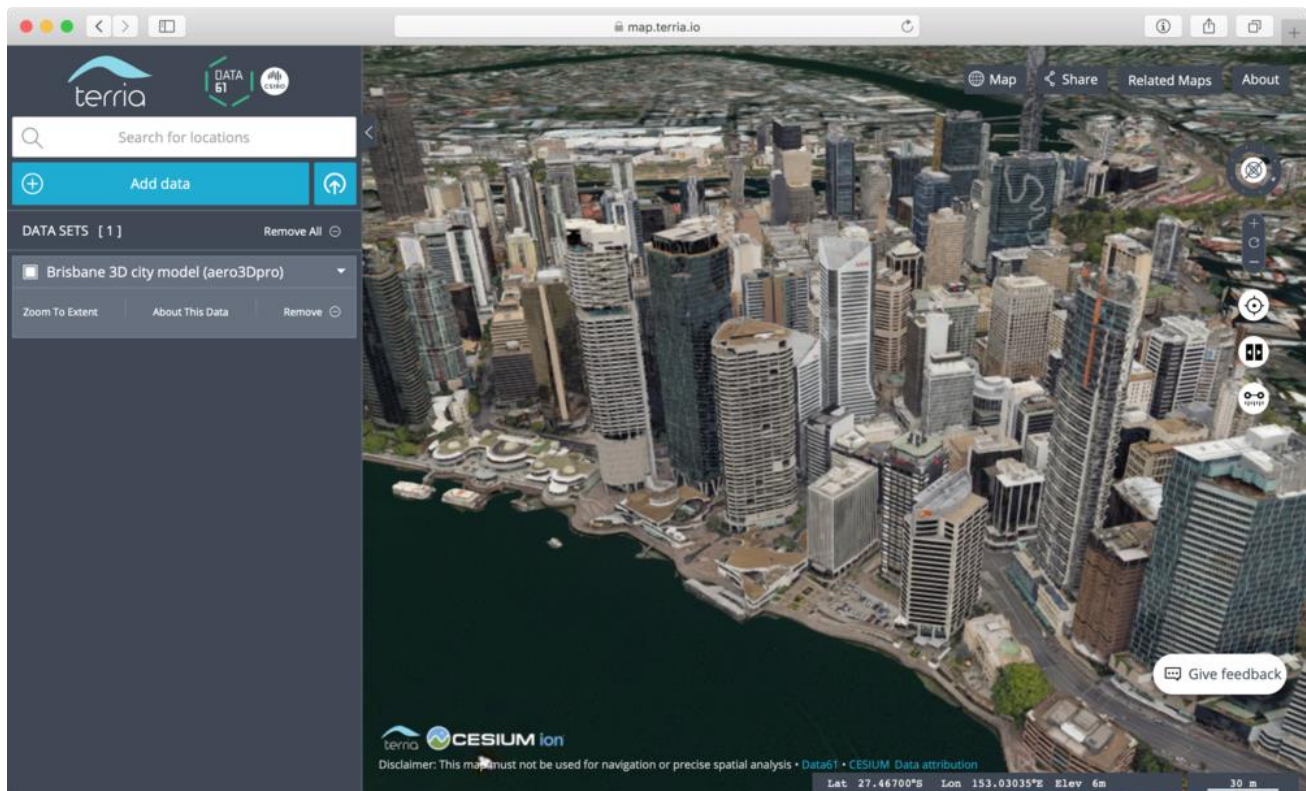


図 2-18 TerriaJS - 表示例

<https://terria.io/>

2-4-12-a. 概要

CesiumJS をベースに、データカタログや独自 UI などの機能が付与されたフレームワーク。スタイル等のカスタマイズが可能だが、機能拡張の幅が限られている。PLATEAU-VIEW-1.1 はこれを用いて構築された。

2-4-12-b. 主な機能

- Leaflet と CesiumJS による 2D、3D 表示
- データカタログ
- データ比較や、チャート表示、時系列データの取り扱い UI

2-4-12-c. 考察

TerriaJS は、以下の理由から、STEP2 の調査対象として適切ではない。

- CesiumJS をベースにしており、描画能力自体には差異がない

2-4-12-d. 参考文献

表 2-15 TerriaJS 参考文献

タイトル	出典
TerriaJS 公式ページ	https://terria.io/
PLATEAU-VIEW-1.1 GitHub ページ	https://github.com/Project-PLATEAU/PLATEAU-VIEW-1.1
TerriaJS で活用する[1/2] TerriaMap の開発環境を構築する	https://www.mlit.go.jp/plateau/learning/tpc07-1/

2-4-13. 3D BAG

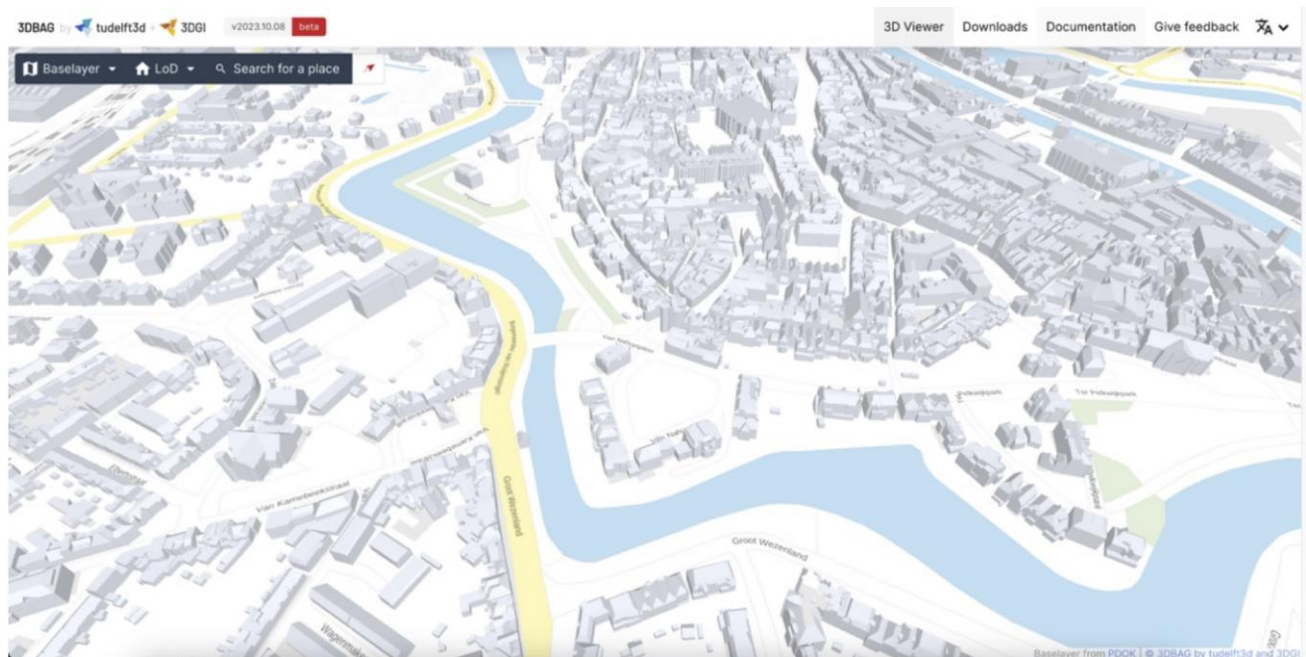


図 2-19 3D BAG 建築物モデル表示例

<https://3dbag.nl/en/viewer>

2-4-13-a. 概要

オランダ国土地情報局（Kadaster）によって開発及び管理されている地理情報プロジェクト。オランダ全土の建物情報を 3D モデルとして提供することを目標としている。デルフト工科大学（Delft University of Technology）の 3D 地理情報グループが開発主体となっている。データセットが主であり、そのビューアーが存在するが、それを第三者が利用できる形にはなっていない。

2-4-13-b. 主な機能

- 自動生成された 3D 建物データセット
 - LOD1.2、LOD1.3、LOD2.2（オランダ全土）
- 3D BAG Viewer
 - Three.js によるデータセットのビューアー

2-4-13-c. 考察

3D BAG は、以下の理由から、STEP2 の調査対象として適切ではない。

- ビューアー自体は 3D 都市モデルの描画が可能であるが、実装が公開されておらず一般利用できない

2-4-13-d. 参考文献

表 2-16 3D BAG 参考文献

タイトル	出典
3D BAG 概要	https://docs.3dbag.nl/en/
3D BAG Viewer	https://3dbag.nl/

3. 3D 都市モデルの表示性能検証

3-1. 目的

STEP1 では、「地図エンジンの調査」として、3D 都市モデルを表示可能な地図エンジンを整理し、その上で、実運用上の観点から STEP2 の対象を選定した。

STEP2 では、「3D 都市モデルの表示性能検証」を行う。選定した地図エンジンを利用し、3D 都市モデルを表示し、その性能を検証する。また、このプロセスで表示する際に使用したデータの形式や変換方法等も整理する。

それらの検証結果をもとに、CesiumJS とは異なる発展性のある地図エンジンを選定し、STEP3「表現方法の調査」の対象とする。

3-2. 検証方法

STEP1 で選定した地図エンジンと、それらに対する 4 種類のデータ形式（後述）を用いて、以下の観点で評価を行う。

- 背景地図の取り扱い
- 地形と高さ情報の取り扱い
- 3D 都市モデルの描画: LOD 切り替え
- 3D 都市モデルの描画: テクスチャ
- 3D 都市モデルの属性情報の取り扱い
- レンダリングパフォーマンス

3-2-1-a. 対象とする地図エンジン

STEP2 で対象となる地図エンジンは、STEP1 で選定した以下の 6 種類である。

- MapLibre GL JS
- Mapbox GL JS (v2)
- deck.gl
- iTowns
- 3DTilesRendererJS
- ArcGIS Maps SDK for JavaScript

3-2-1-b. 対象とするデータ形式

STEP2 では、4 種類のデータ形式を用いて 3D 都市モデルの描画を検証した。以下の表に、各地図エンジンと、対応するデータ形式を述べる。

表 3-1 各地図エンジンによるデータ形式の対応状況

	3D Tiles	MVT	glTF	I3S
MapLibre GL JS	X	○	△	X
Mapbox GL JS	X	○	△	X
deck.gl	○	○	○	○
iTowns	○	○	X	X
3DTilesRendererJS	○	X	X	X
ArcGIS Maps	X	○	X	○

3D 都市モデルは CityGML で作成・提供されている。CityGML 自体はデータ交換用の中間フォーマットとして定義されているため、これをウェブ上で可視化するためにはレンダリングに適した形式に変換する必要がある。そのため、各地図エンジンで取扱うことができる形式に変換する必要がある。

またこの時、GIS の観点からは、そのフォーマットが各建築物モデルなどに付随する「主題属性」を保持でき、これにアプリから容易にアクセスできるかも重要である。この観点では、glTF をはじめとする 3D コンピュータグラフィックス系のデータ形式は、アプリ上での属性情報へのアクセスに課題があることに注意が必要である。

3-3. 検証結果のまとめ

STEP1 で選定した 6 種類の地図エンジンを用いて、それぞれで利用可能な形式のデータを用意し、各種地図エンジンのデータ表示機能を調査した。

表 3-2 各地図エンジンによる評価基準の状況

	背景地図	地形・高さ情報	3D 都市モデルの描画：LOD 切り替え	3D 都市モデルの描画：テクスチャ	3D 都市モデルの属性情報	レンダリングパフォーマンス
MapLibre GL JS	○	○	△	△	○	○
Mapbox GL JS	○	○	△	△	○	○
deck.gl	○	○	○	○	△	○
iTowns	○	○	○	○	X	△
3DTilesRendererJS	X	X	○	○	X	○
ArcGIS Maps	○	○	○	○	△	○

MapLibre GL JS と Mapbox GL JS (v2) は、主に MVT 押出による建築物モデルの簡易な表現が可能であり、これは軽量に動作し、属性情報の取得や、簡易なテクスチャ付与も可能である。両者には機能の差異が存在するが、基本的な描画能力は共通しており、後者は利用にクラウドサービスのアクセストークンが必要である。

deck.gl は 3D Tiles を表示することができ、加えて MVT の表示や MapLibre GL JS との併用も可能である。また、これらと様々な可視化レイヤーが柔軟に組み合わせられる仕組みになっている。

iTowns も 3D Tiles や MVT の表示が可能だが、他に比べて特徴的な機能は見られず、また描画速度が劣る。また、ドキュメントが充実していないというデメリットがある。

3DTilesRendererJS は、3D Tiles の読込が可能であり軽量に動作したが、他の地理空間情報との組み合わせが容易ではなく、また高度な可視化表現は Three.js を用いて追加実装することで可能ではあるが多くの実装が必要であり、現実的な用途には適していない。

ArcGIS Maps SDK for JavaScript は I3S による 3D 都市モデルの表現が可能であり、また他に見られない機能も提供されているが、あくまで ArcGIS のサービスと共に利用する必要がある。

これらの結果や、STEP1 で確認した各種機能や開発状況、ライセンスの必要性を踏まえて、以下の 2 種類による発展的な可視化表現を検証する。

- deck.gl (3D Tiles による緻密な表現)
- MapLibre GL JS (MVT 押出による簡易な表現)

3-4. 各データ形式の詳細

3-4-1. 3D Tiles

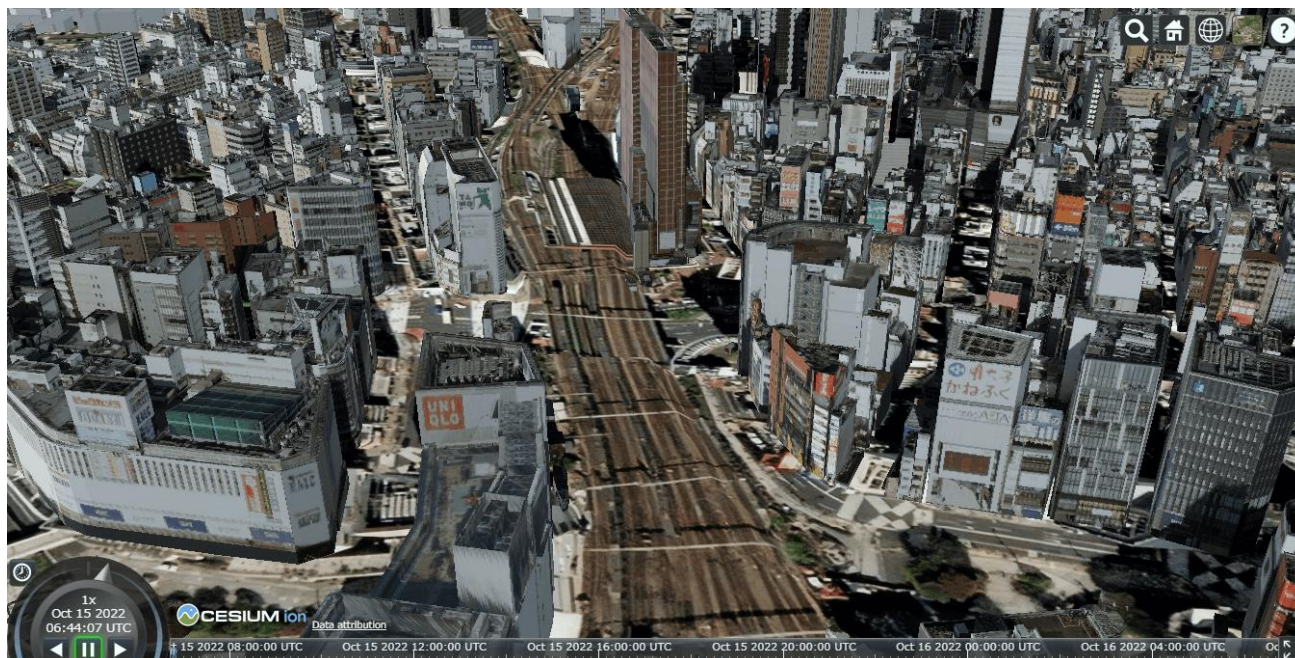


図 3-1 3D Tiles と CesiumJS による 3D 都市モデルの表示例

<https://www.mlit.go.jp/plateau/learning/tpc06-1/>

大規模な多種の 3D 地理空間データをストリーミングするためのオープンな標準仕様。CesiumJS を開発する Cesium 社が策定を始めた仕様であり、Open Geospatial Consortium (OGC) スタンダードに認定されている。

全ての建築物モデルを一つの大きなファイルで表すわけではなく、範囲ごとのタイル形式で配信されるため、広い領域を効率的に扱うことができる。

使用する地形とデータの組み合わせにより、高さ方向（ジオイド高の含有）の調整が必要なことに注意が必要である。

CityGML から 3D Tiles へは、Safe Software 社のソフトウェア FME を用いることで変換が可能である。また、3D 都市モデルが提供されている G 空間情報センターの 3D 都市モデルポータルサイトでは 3D Tiles へ変換済みのものも公開されている。

今回の検証では、G 空間情報センターで公開されている変換済みの 3D Tiles データを用いた。

3-4-1-a. 参考文献

表 3-3 3D Tiles 参考文献

タイトル	出典
3D Tiles – Cesium	https://cesium.com/why-cesium/3d-tiles/
3D Tiles - Open Geospatial Consortium	https://www.ogc.org/standard/3DTiles/
3D 都市モデルの入手方法とデータ形式	https://www.mlit.go.jp/plateau/learning/tpc03-1/
FME Form の機能と基本的な使い方	https://www.mlit.go.jp/plateau/learning/tpc04-1/
3D 都市モデル（Project PLATEAU） 東京 23 区（2022 年度）	https://www.geospatial.jp/ckan/dataset/plateau-tokyo23ku-2022

3-4-2. MVT (Mapbox Vector Tiles)



図 3-2 MVT と MapLibre GL JS による建物の押出表現例

<https://maplibre.org/maplibre-gl-js/docs/examples/3d-extrusion-floorplan/>

Mapbox 社が制定した、地図データを効率的に表現・配信するためのオープンな標準仕様。ベクターデータをタイル形式で表す。拡大・縮小によっても高い解像度でクリアな画質を保つことができる。サイズは小さく、迅速に伝送することが可能である。

Mapbox 社が開発し、現在は Felt 社が引き継いで発展させているソフトウェア tippecanoe により、GeoJSON や FlatGeobuf といったデータ形式から変換することができる。

このとき、MVT はあくまで 2 次元での範囲をポリゴンとして表すことに注意が必要である。そのため、CityGML に含まれる FootPrint や RoofEdge といった属性を用い、高さ情報を属性として付与することで、後述の押出表現が可能となる。しかし、複雑な 3D 形状のものはその性質上、表現できない。

3-4-2-a. 参考文献

表 3-4 MVT (Mapbox Vector Tiles)参考文献

タイトル	出典
Mapbox Vector tiles standards	https://docs.mapbox.com/data/tilesets/guides/vector-tiles-standards/
Tippecanoe GitHub ページ	https://github.com/felt/tippecanoe
3D 都市モデル (Project PLATEAU) 東京都 23 区 (Building-MVT 2020 年 度)	https://www.geospatial.jp/ckan/dataset/plateau-tokyo23ku-building-mvt-2020
PLATEAU の CityGML Building LOD2 を MVT にしてみた Qiita 記 事	https://qiita.com/frogcat/items/84b413c7e08c00b78c6a
plateau-lod2-mvt GitHub ページ	https://github.com/indigo-lab/plateau-lod2-mvt
PLATEAU の CityGML LOD2 を MVT に変換するツールを作ってみた Qiita 記事	https://qiita.com/t-mat/items/51d605911b78a21a48a5
PLATEAU の CityGML から MVT を 作成する作業の記録 Qiita 記事	https://qiita.com/t-mat/items/a57f44c46d9e7f0aac56

3-4-3. glTF (GL Transmission Format)



図 3-3 glTF のサンプルアセットモデルをレンダリングした例

<https://github.com/KhronosGroup/glTF-Sample-Assets/blob/main/Models/ABeautifulGame/README.md>

Khronos Group により開発とメンテナンスが行われているオープンな標準仕様。JSON によって 3D モデルやシーンを表現する。特に Web 環境での 3D アセットの効率的な転送と表示を目的として設計されている。「3D における JPEG」とも呼ばれ、多くのアプリケーションや地図エンジンでサポートされている。

バイナリ形式として GLB フォーマットも存在し、テクスチャなどを外部ファイルの参照ではなく同梱とすることができる。

3D Tiles と異なり、タイル形式ではなく個別ファイルであるため、広領域を対象とすることには不向きである。

CityGML からの変換の一例としては、Safe Software 社のソフトウェア FME を用いることで FBX 形式（Autodesk 社が所有するプロプライエタリなフォーマット）へ変換し、そこから glTF へとするという方式がある。また、一部の 3D 都市モデルは FBX 形式でも公開されている。

今回の検証では、G 空間情報センターで公開されている FBX をダウンロードし、それをオープンソースの DCG 制作ソフトウェア Blender により glTF へと変換した。

Blender で glTF へ変換する際に、水平位置を平面直角座標系の原点から対象メッシュ中心分を並行移動させた。また、そのままの状態では建築物モデルが地表面と設置せず浮上した状態になるため、Blender のアドオン drop_it を用いて建築物モデルを着地させ、データを作成した。

3-4-3-a. 参考文献

表 3-5 glTF 参考文献

タイトル	出典
glTF GitHub ページ	https://github.com/KhronosGroup/glTF
3D 都市モデル（Project PLATEAU） 東京都 23 区	https://www.geospatial.jp/ckan/dataset/plateau-tokyo23ku
Three.js を使った 3D 都市モデルの読み込みと表示	https://www.mlit.go.jp/plateau/learning/tpc12-1/
Blender Addon Drop It	https://andreausaust.gumroad.com/l/drop_it

3-4-4. I3S (Indexed 3D Scene Layer)



図 3-4 I3S と ArcGIS Online による 3D 都市モデルの描画例

<https://github.com/Esri/i3s-spec>

ArcGIS を提供する Esri 社による、3D 地理空間情報のオープンな標準仕様。3D Tiles と同様に、OGC スタandard に認定されている。

Scene Layers のストリーミングモデルが I3S、永続化モデルが SLPK (Scene Layer Package) で、どちらも JSON 及びバイナリとしてエンコードされる。

基本的に ArcGIS での利用が想定されるが、deck.gl でも、loaders.gl の I3S Loader を用いてこのデータ形式を扱うことが可能である。

CityGML から、ArcGIS で利用可能な Esri 社の独自ファイル形式「ファイル ジオデータベース」へは「3D-CityModel-ConversionTools-for-ArcGIS-v2」を用いることで変換可能である。このツールを利用するためには、バージョン 3.1 以上の ArcGIS Pro と Data Interoperability エクステンションが必要であり、またその利用にはライセンスの有効化が必要である。ファイル ジオデータベースから SLPK へは、ArcGIS Pro の拡張機能「ArcGIS Data Interoperability エクステンション」を利用することで可能となる。また、ファイル ジオデータベースへ変換済みの 3D 都市モデルも公開されている。

3D Tiles から SLPK へは、loaders.gl の tile-converter ツールを用いることで変換が可能である。

変換した SLPK やファイル ジオデータベースは、ArcGIS Online へアップロードすることで利用可能となる。

3-4-4-a. 参考文献

表 3-6 I3S 参考文献

タイトル	出典
i3s-spec GitHub ページ	https://github.com/Esri/i3s-spec
I3S 規格	https://www.ogc.org/standard/i3s/
I3S をサポートしているフレームワーク紹介 YouTube 動画	https://www.youtube.com/watch?v=0C2fvQXqODQ
I3SLoader	https://loaders.gl/docs/modules/i3s/api-reference/i3s-loader
deck.gl で I3S (シーンサービス) を使用してみる	https://freedom-tech.hatenablog.com/entry/deckgl/I3S
3D-CityModel-ConversionTools-for-ArcGIS-v2 GitHub ページ	https://github.com/EsriJapan/3D-CityModel-ConversionTools-for-ArcGIS-v2
ArcGIS Pro ドキュメント (エクステンションでサポートしているフォーマット)	https://pro.arcgis.com/ja/pro-app/latest/help/data/data-interoperability/supported-formats-with-the-data-interoperability-extension.htm
国土交通省 3D 都市モデル「Project PLATEAU」の活用事例集	https://3d-city-model.esri.com/search?collection=Document
tile-converter,	https://www.npmjs.com/package/@loaders.gl/tile-converter
tile-converter-bundle	https://loaders.gl/docs/modules/tile-converter/api-reference/tile-converter-bundle
3D 都市モデル(3D Tiles)をシーンレイヤー(I3S)に変換	https://community.esri.com/t5/arcgis-開発者コミュニティ-documents/plateau-の 3d-都市モデル-3d-tiles-をシーン-レイヤー-i3s/ta-p/1105433

3-5. 各地図エンジンの詳細

以下の節では、各地図エンジンによる各種 3D 都市モデルの描画結果について述べ、また加えてデータ描画以外の各種機能についても紹介する。

3-5-1. MapLibre GL JS



図 3-5 MapLibre GL JS - MVT を用いた建築物モデル表現

デモアプリケーション STEP2 - MapLibre GL JS

<https://project-plateau.github.io/mapengine-survey/step2/maplibre/index.html>

3-5-1-a. 背景地図

ラスタータイルおよび、CesiumJS が対応していないベクタータイルによる背景地図が描画可能である。

3-5-1-b. 地形・高さ情報

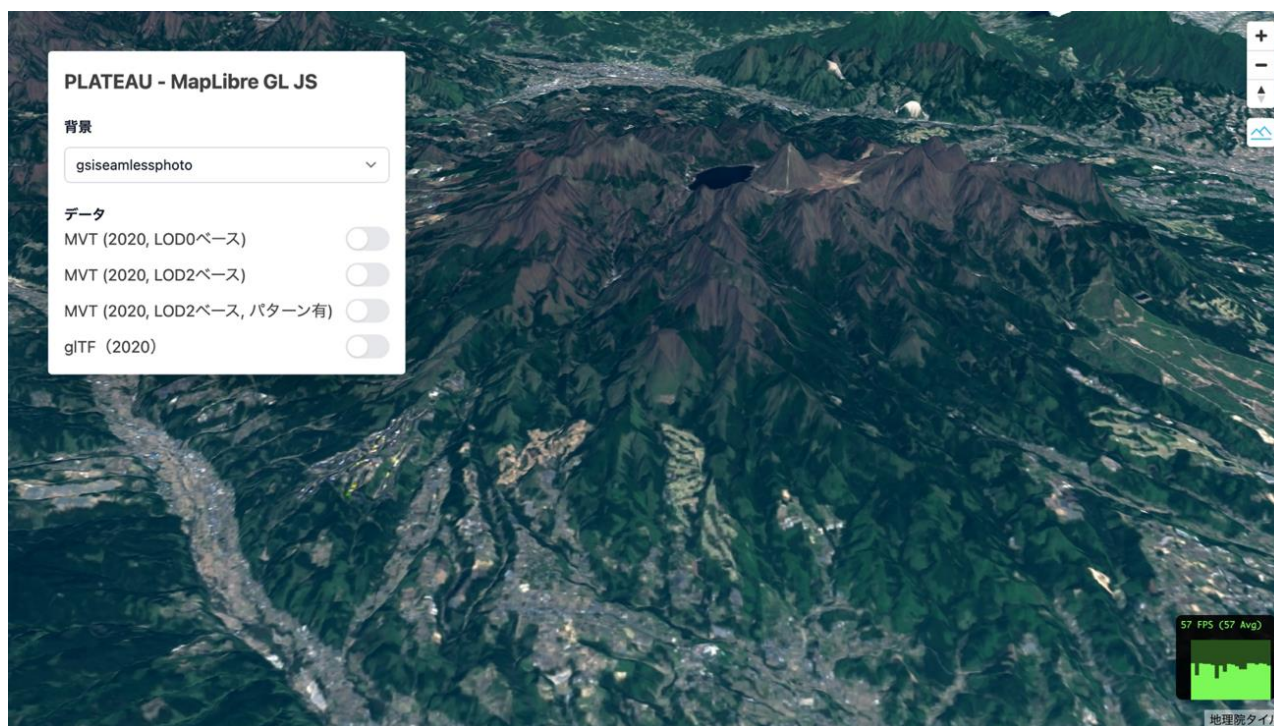


図 3-6 MapLibre GL JS - 地形表示

デモアプリケーション STEP2 - MapLibre GL JS

<https://project-plateau.github.io/mapengine-survey/step2/maplibre/index.html>

標高タイルにより、3D 地形表示が可能である。

例では、地理院標高タイルをもとに地形を表示している。地理院標高タイルは、一般的に用いられる Terrain-RGB ではなく独自形式であるため、そのままの形では MapLibre GL JS で利用することができない。地理院標高タイルおよび Terrain-RGB、どちらもピクセルの R・G・B 値を元に 256 進数として変換しているが、その計算式が異なる。このとき、これら元のピクセル値から Terrain-RGB 相当の標高値へと変換することで利用可能となる（地理院標高タイルの標準仕様、および Terrain-RGB との比較は参考資料を参照のこと）。このリアルタイム変換には、MapLibre GL JS の addProtocol 機能を利用しているため、Mapbox GL JS (v2) で同様のことはできない。

MapLibre GL JS のデフォルトで用意されている、地形表示ボタン（画面右上にある山アイコン）で、地形表示のオンオフができる。

3-5-1-c. 3D 都市モデルの描画: LOD 切り替え

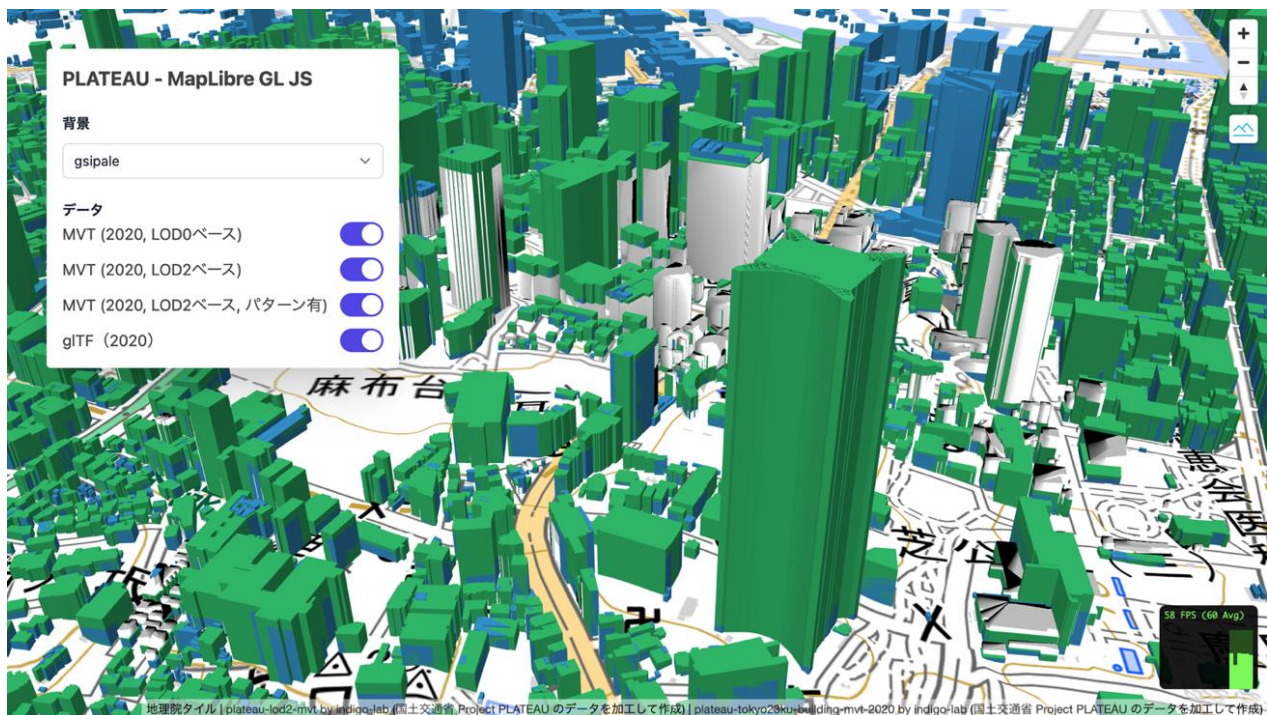


図 3-7 MapLibre GL JS - 建築物モデル表示

デモアプリケーション STEP2 - MapLibre GL JS (2020, LOD0 ベース)

<https://project-plateau.github.io/mapengine-survey/step2/maplibre/index.html>

前述した 3D 地表面を表示し、その上に、MVT や glTF による 3D 都市モデルを地図上に表示することが可能である。

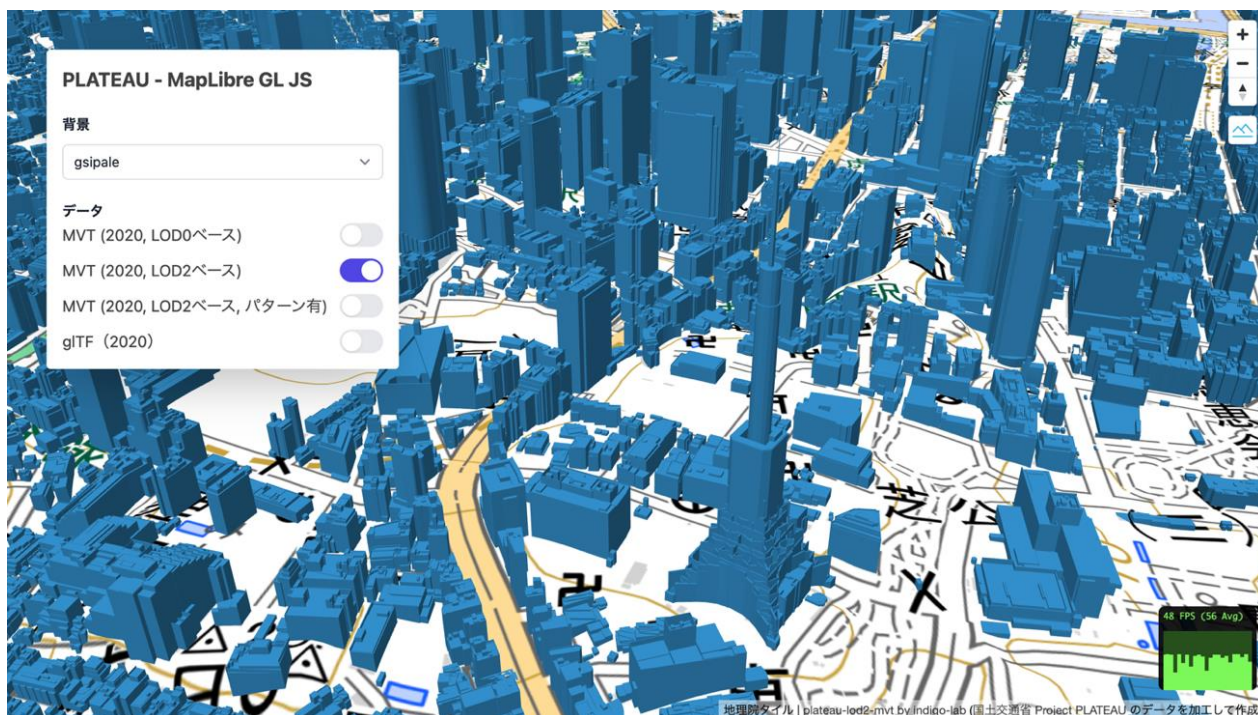


図 3-8 MapLibre GL JS - MVT を用いた建築物モデル表現 (LOD2 ベース)

デモアプリケーション STEP2 - MapLibre GL JS、MVT (2020,LOD2 ベース)

<https://project-plateau.github.io/mapengine-survey/step2/maplibre/index.html>

MVT では 2D の範囲のみを指定できる。この範囲（ポリゴン）へ対して、高さ情報を属性として持たせることで、その高さ分だけ押出（extrusion）表現を行い、3D で建築物モデルを表現する。

当検証では、LOD0 ベースのものと、LOD2 ベースの MVT を用いた。LOD2 ベースの方が、細かく 2D 範囲とその高さが指定されているため、より現実に近い形状表現が可能であるが、押出という表現の性質上、ドーム状の曲線などは表現できない。

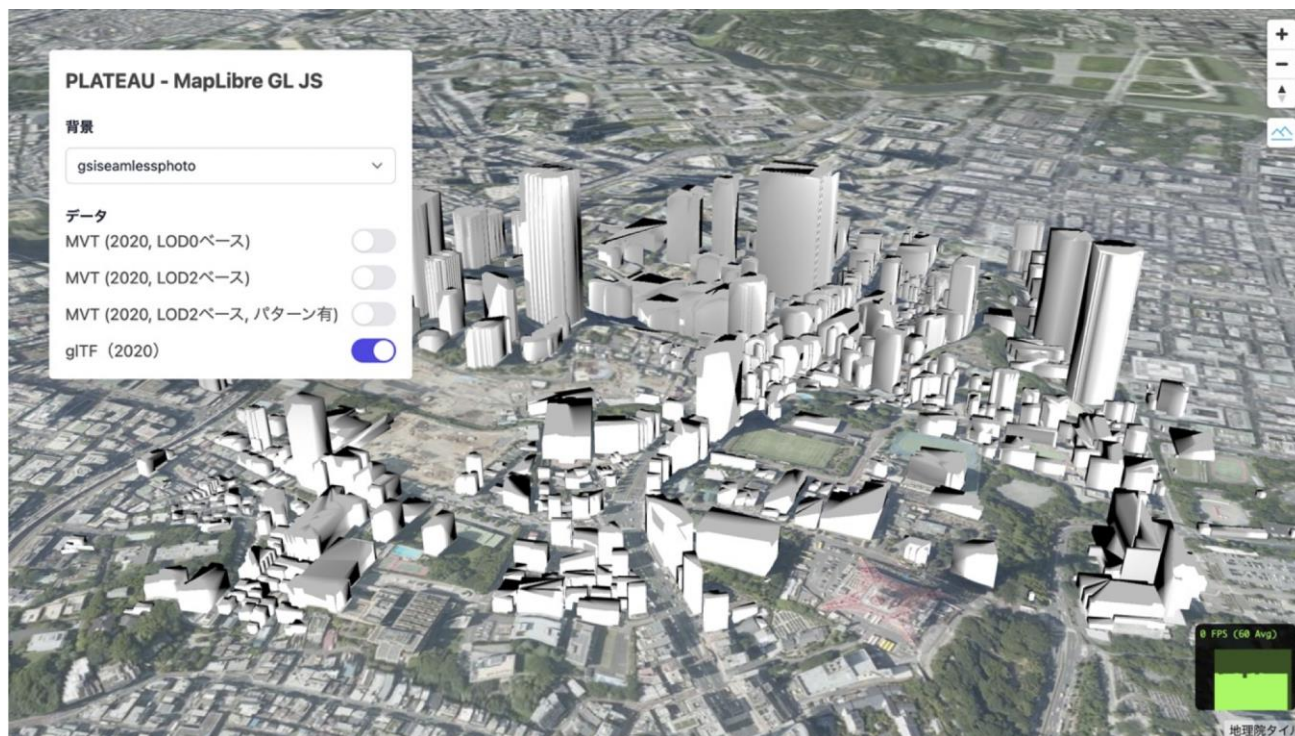


図 3-9 MapLibre GL JS - glTF を用いた建築物モデル表現
デモアプリケーション STEP2 - MapLibre GL JS、glTF(2020)

<https://project-plateau.github.io/mapengine-survey/step2/maplibre/index.html>

glTF は、MapLibre GL JS のデフォルト機能ではなく、3D コンピュータグラフィックスライブラリ Three.js を用いたカスタムレイヤーを作成することで、表示を実現することが可能である。ライティングなどの設定は自ら調整する必要がある。

3D Tiles は、MapLibre GL JS 単体ではそのまま描画することができない。ただし、後述する deck.gl と組み合わせることでの利用は可能である。

先述した glTF の例と同様に、Three.js を用いたカスタムレイヤーによって、理論上は 3D Tiles の描画が可能である（参考資料に実験的実装の例を掲載）。ただし、これはあくまで描画を可能とするものであり、パフォーマンス最適化や属性の取り扱いなど、不足する部分が多く現実的な利用は困難である。

3-5-1-d. 3D 都市モデルの描画: テクスチャ

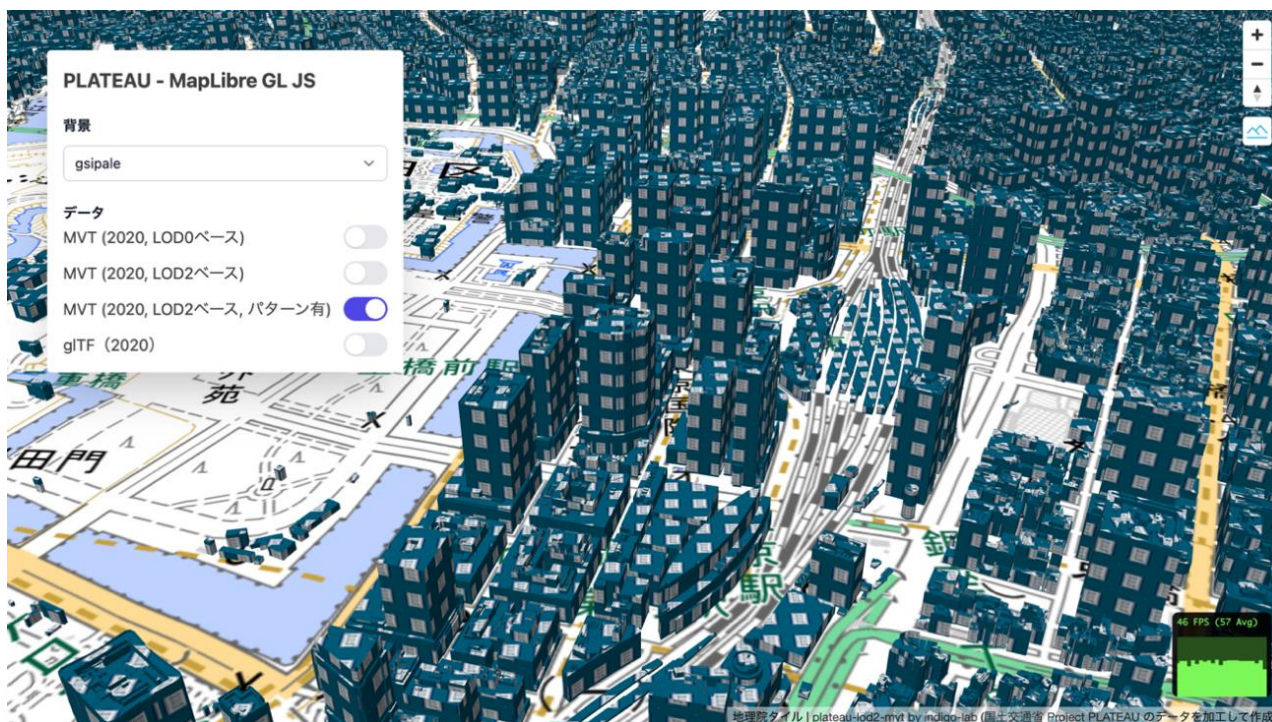


図 3-10 MapLibre GL JS - 建築物モデルへテクスチャを貼り付けた例

デモアプリケーション STEP2 - MapLibre GL JS、MTG (2020,LOG2 ベース,パターン有)

<https://project-plateau.github.io/mapengine-survey/step2/maplibre/index.html>

建築物モデルへは、パターン（テクスチャ画像）を貼り付けることが可能である。テクスチャは一様のパターンとして建築物モデル表面へ適用される。属性の条件により、建築物モデルごとに異なるテクスチャを適用することは可能であるが、全建築物モデルへそれぞれ異なる適切なテクスチャを付与することは現実的には困難である。

3-5-1-e. 3D 都市モデルの属性情報

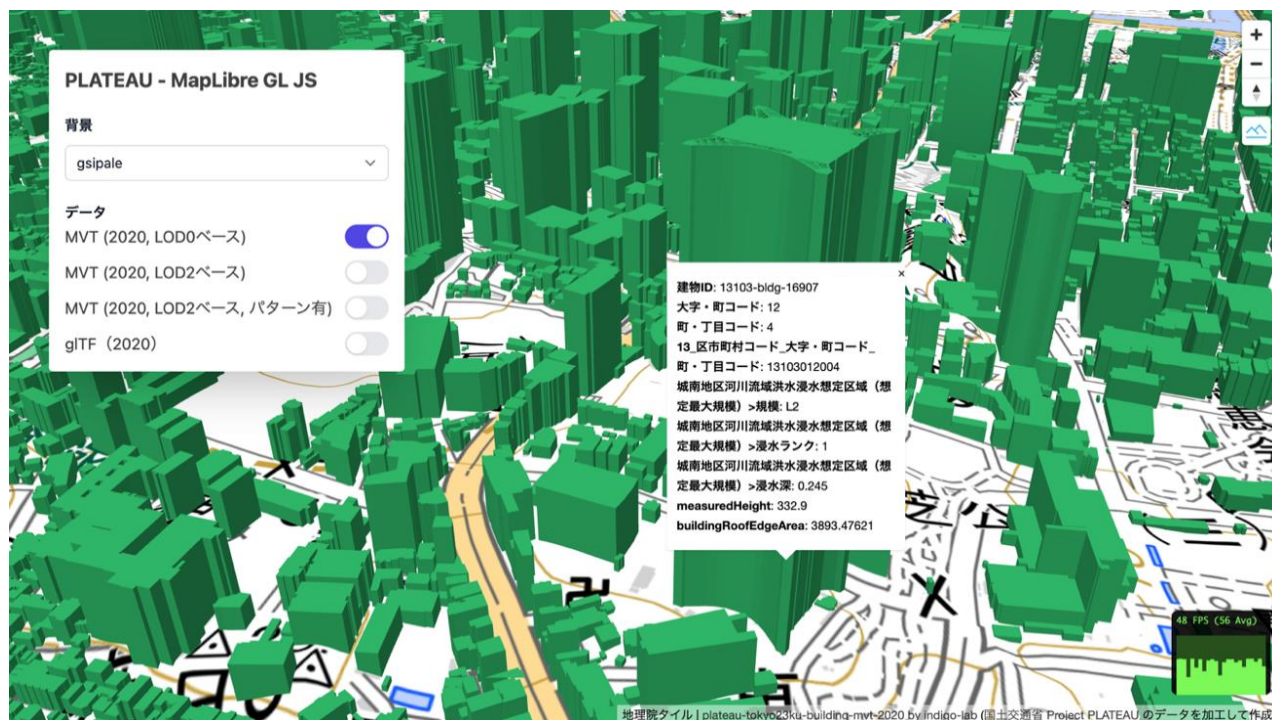


図 3-11 MapLibre GL JS - MVT を用いた建築物モデル表現と属性表示

デモアプリケーション STEP2 - MapLibre GL JS 、MVT (2020,LOD0 ベース)

<https://project-plateau.github.io/mapengine-survey/step2/maplibre/index.html>

MVT の建築物モデルには属性情報を保持させることが可能であり、この情報は、建築物モデルをクリックすることで取得し、例えばポップアップによる表示が可能である。

3-5-1-f. レンダリングパフォーマンス

MVT という軽量なタイルデータ形式を用い、また押出という簡易な 3D 表現のため、CesiumJS などと比べて非常に高速に動作する。一方で、CesiumJS での 3D Tiles のように、緻密な建築物モデル表現を行うことはできない。

3-5-1-g. 考察

MVT 内の高さ情報を用いて押し出すという簡素な表現にはなるが、3D の建物を表現可能である。また MVT はシンプル・軽量で、取扱いが容易で、高速に動作する。この表現力で十分であれば、用途によっては適していると考えられる。また、Three.js を用いた glTF の表示も可能ではあるが、広域で一定以上の建築物モデルを対象とする GIS ではタイル形式の方がパフォーマンスが最適化されている点から現実的な利用を想定すると他の選択肢に比べて劣る。

3-5-1-h. 参考資料

表 3-7 MapLibre GL JS 参考文献

タイトル	出典
標高タイルの詳細仕様	https://maps.gsi.go.jp/development/demtile.html
MapLibre GL JS と地理院標高タイルで 3D 地形を表示する	https://qiita.com/Kanahiro/items/1e9c1a4ad6be76b27f0f
maplibre-gl-gsi-terrain GitHub ページ	https://github.com/mug-jp/maplibre-gl-gsi-terrain
MapLibre GL JS TerrainControl	https://maplibre.org/maplibre-gl-js/docs/API/classes/TerrainControl/
MapLibre GL JS 3D 屋内マッピングのためのポリゴンの押し出し事例	https://maplibre.org/maplibre-gl-js/docs/examples/3d-extrusion-floorplan/
MapLibre GL JS ポリゴンにパターンを追加する事例	https://maplibre.org/maplibre-gl-js/docs/examples/fill-pattern/
MapLibre GL JS three.js で 3D モデルを追加する事例	https://maplibre.org/maplibre-gl-js/docs/examples/add-3d-model/
mapbox-3dtiles GitHub ページ	https://github.com/Geodan/mapbox-3dtiles
mapbox_3dtiles_samples GitHub ページ	https://github.com/bertt/mapbox_3dtiles_samples

3-5-2. Mapbox GL JS (v2)

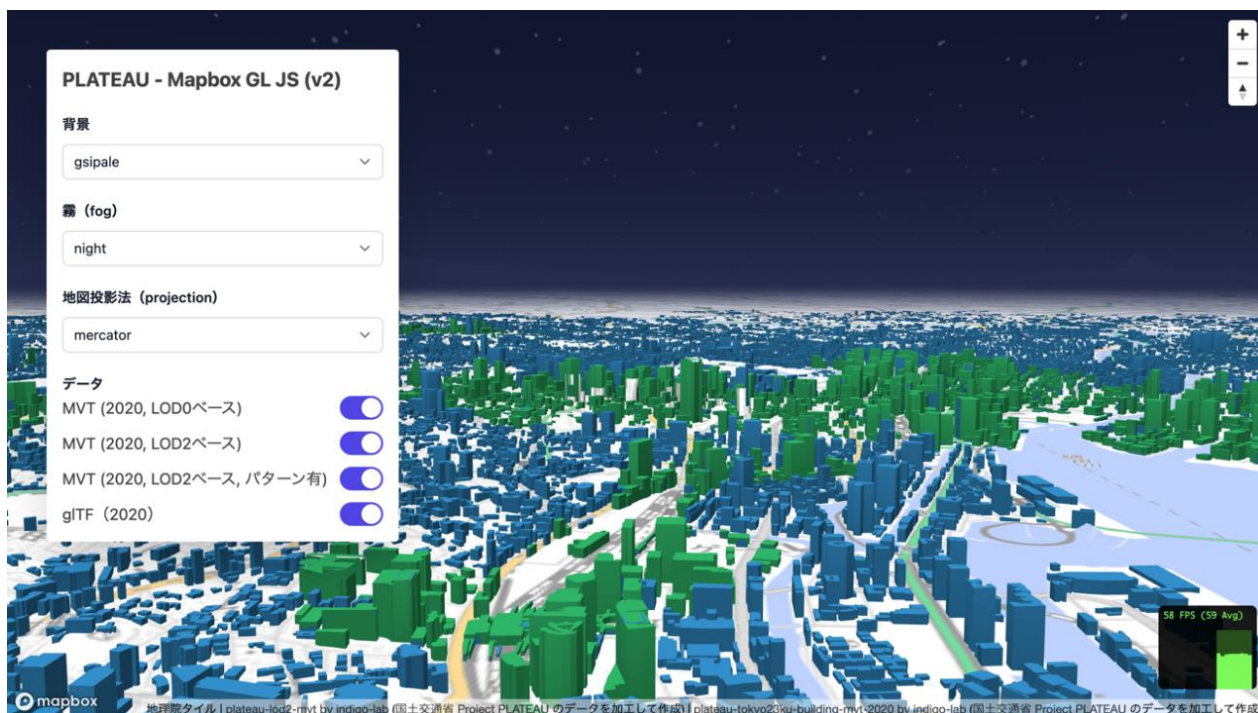


図 3-12 Mapbox GL JS (v2) - 建築物モデル表現
デモアプリケーション STEP2 - Mapbox GL JS(v2)

3-5-2-a. 背景地図

MapLibre GL JS と同様に、ラスタータイルおよび、CesiumJS が対応していないベクタータイルによる背景地図が描画可能である。

3-5-2-b. 地形・高さ情報

MapLibre GL JS と同様に、標高タイルを用いて、3D 地形表示が可能である。先述した MapLibre GL JS と同じく、一般的な Terrain-RGB 形式に対応している。しかし MapLibre GL JS の節で解説したタイルの動的加工（addProtocol）機能がないため、Terrain-RGB と異なる独自形式を採用する地理院標高タイルをそのまま用いることはできないが、外部のエッジコンピューティングサービスを用いて同様の処理を達成することは可能である。

3-5-2-c. 3D 都市モデルの描画: LOD 切り替え

MapLibre GL JS と同様に、前述した 3D 地表面を表示し、その上に、MVT や glTF による 3D 都市モデルを地図上に表示することが可能である。

MVT では 2D の範囲のみを指定できる。この範囲（ポリゴン）へ対して、高さ情報を属性として持たせることで、その高さ分だけ押出（extrusion）表現を行い、建築物モデルを表現する。

MapLibre GL JS と同様に、Three.js を用いたカスタムレイヤーを用いることで glTF の描画が可能である。

3D Tiles も、MapLibre GL JS と同様に、Mapbox GL JS 単体ではそのまま描画することができない。ただし、後述する deck.gl と組み合わせることでの利用は可能である。

3-5-2-d. 3D 都市モデルの描画: テクスチャ

MapLibre GL JS と同様に、建築物モデルへは、パターン（テクスチャ画像）を貼り付けることが可能である。テクスチャは一様のパターンとして建築物モデル表面へ適用される。属性の条件により、建築物モデルごとに異なるテクスチャを適用することは可能であるが、全建築物モデルへそれぞれ異なる適切なテクスチャを付与することは現実的には困難である。

3-5-2-e. 3D 都市モデルの属性情報

MVT の建築物モデルには属性情報を保持させることが可能であり、この情報は、建築物モデルをクリックすることで取得し、例えばポップアップによる表示が可能である。

3-5-2-f. レンダリングパフォーマンス

MVT という軽量なタイリング技術を用い、また押出という簡易な 3D 表現のため、CesiumJS などと比べても非常に高速に動作する。一方で、CesiumJS での 3D Tiles のように、緻密な建築物モデル表現を行うことはできない。

3-5-2-g. 考察

基本的な描画性能は、先述の MapLibre GL JS と同等である。以下に、MapLibre GL JS と比べた際の主な差異について述べる。

スカイレイヤー

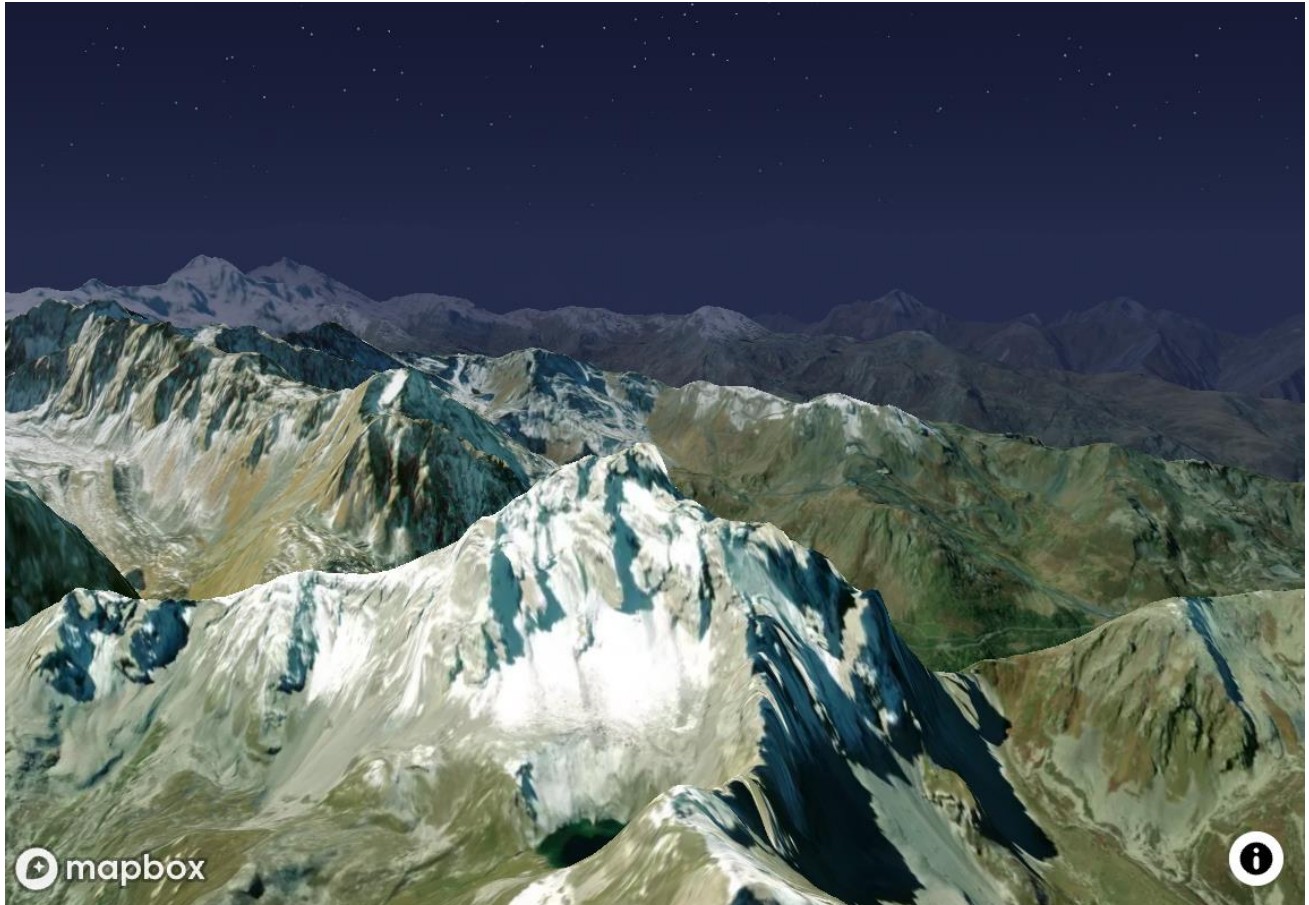


図 3-13 Mapbox GL JS でのスカイレイヤーによる夜空表現の例

<https://docs.mapbox.com/mapbox-gl-js/example/add-fog/>



図 3-14 Mapbox GL JS でのスカイレイヤーによる大気グラデーションの例

<https://docs.mapbox.com/style-spec/reference/layers/#sky>

Mapbox GL JS (v2) では、大気のグラデーションや星、霧などを表現することが可能である。他方、MapLibre GL JS でも同様の機能が検討されているが、また開発途上である。

地图投影法



図 3-15 Mapbox GL JS での Equal Earth 投影法の例

<https://docs.mapbox.com/mapbox-gl-js/guides/projections/>

Mapbox GL JS (v2) では、WebGIS で標準となっているメルカトル図法に加えて、天気図や航空図などに適し国土地理院「50 万分の 1 地方図」などでも採用されるランベルト正角円錐図法や、国土地理院「全国都道府県市町村別面積調」でも用いられているアルベルス正積円錐図法など、計 7 種類の地図投影法を利用可能である。また、ズームアウトした時に全体を地球儀ビューとすることもできる。

MapLibre GL JS では、現在はメルカトル図法以外は利用できない。地球儀ビューについては、検討されているがまだ開発途上である。

Mapbox GL JS (v1) は、機能の大部分は MapLibre GL JS と同様である。ただし先述したように、大気グラデーション機能や地球儀ビューの表示といった、後者にはない機能を一部有する。そのような追加機能を求める場合は、Mapbox GL JS を用いる必要がある。

また Mapbox GL JS (v1) は、コードは GitHub で公開されているが、ライセンスが OSS ライセンスではなく、オープンソースではない。利用には Mapbox 社が提供するクラウドサービスのアクセストークンが必要で、マップインスタンスが一定回数以上初期化された場合に課金対象となる。そのため、Mapbox 社のタイルデータを利用して地図描画する場合には、Mapbox GL JS の利用が現実的に考えられるが、他のクラウドサービスやデータを利用する際には MapLibre GL JS の利用がより適している。

3-5-2-h. 参考文献

表 3-8 MapLibre GL JS (v2) 参考文献

タイトル	出典
Mapbox GL JS ドキュメント フォグ (霧)	https://docs.mapbox.com/jp/mapbox-gl-js/style-spec/fog/
MapLibre Style Spec	https://maplibre.org/maplibre-style-spec/sky/
Mapbox GL JS Projections	https://docs.mapbox.com/mapbox-gl-js/guides/projections/
MapLibre Roadmap - Globe View	https://maplibre.org/roadmap/globe-view/
MapLibre GL JS API addProtocol	https://maplibre.org/maplibre-gl-js/docs/API/functions/addProtocol/
MapLibre GL JS と地理院標高タイルで 3D 地形を表示する Qiita 記事	https://qiita.com/Kanahiro/items/1e9c1a4ad6be76b27f0f
地理院標高タイルを Cloudflare Workers で Terrain RGB に変換して利用する Qiita 記事,	https://qiita.com/ciscorn/items/43daa33dca31e6233928
MapLibre Style Spec Deprecations	https://maplibre.org/maplibre-style-spec/deprecations/
Mapbox Docs Create an access token	https://docs.mapbox.com/help/getting-started/access-tokens/

3-5-3. deck.gl

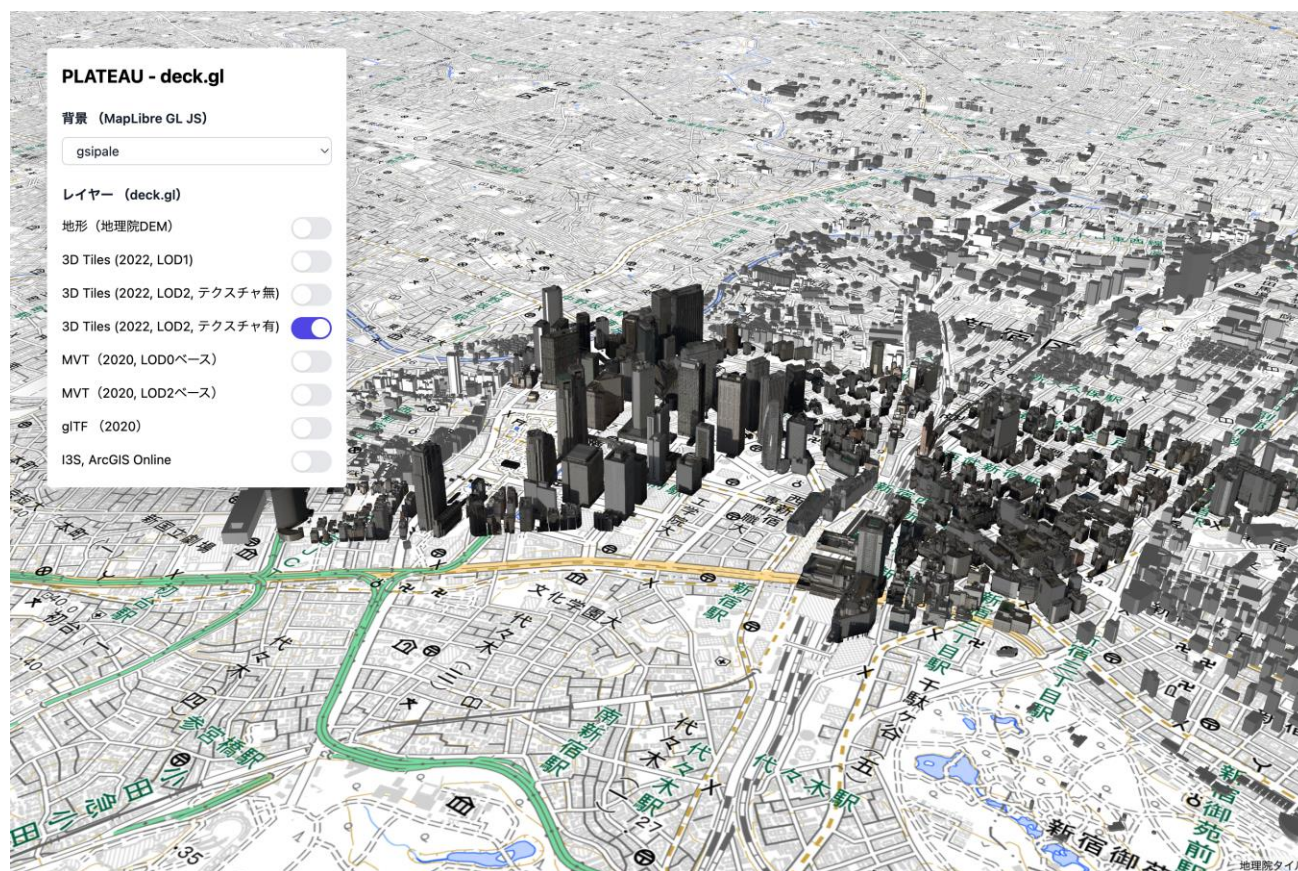


図 3-16 deck.gl - 建築物モデルの表示
デモアプリケーション STEP2 - deck.gl

<https://project-plateau.github.io/mapengine-survey/step2/deckgl/index.html>

3-5-3-a. 背景地図

標準の `TileLayer` を用いることで、ラスタータイルを表示することが可能である。また、標準の `MVTLayer` を用いることで、CesiumJS が対応していないベクタータイルによる背景地図を表示可能である。他にも、`MVTLayer` を用いず、`deck.gl` と `MapLibre GL JS` もしくは `Mapbox GL JS` を併用し重ねて利用することができる。それら地図エンジンの Web フレームワーク `React` でのラッパーライブラリが提供されており、これを使うことで容易に導入することができる。

3-5-3-b. 地形・高さ情報

標準の TerrainLayer を用いることで、標高タイルから、3D 地形を表現することが可能である。

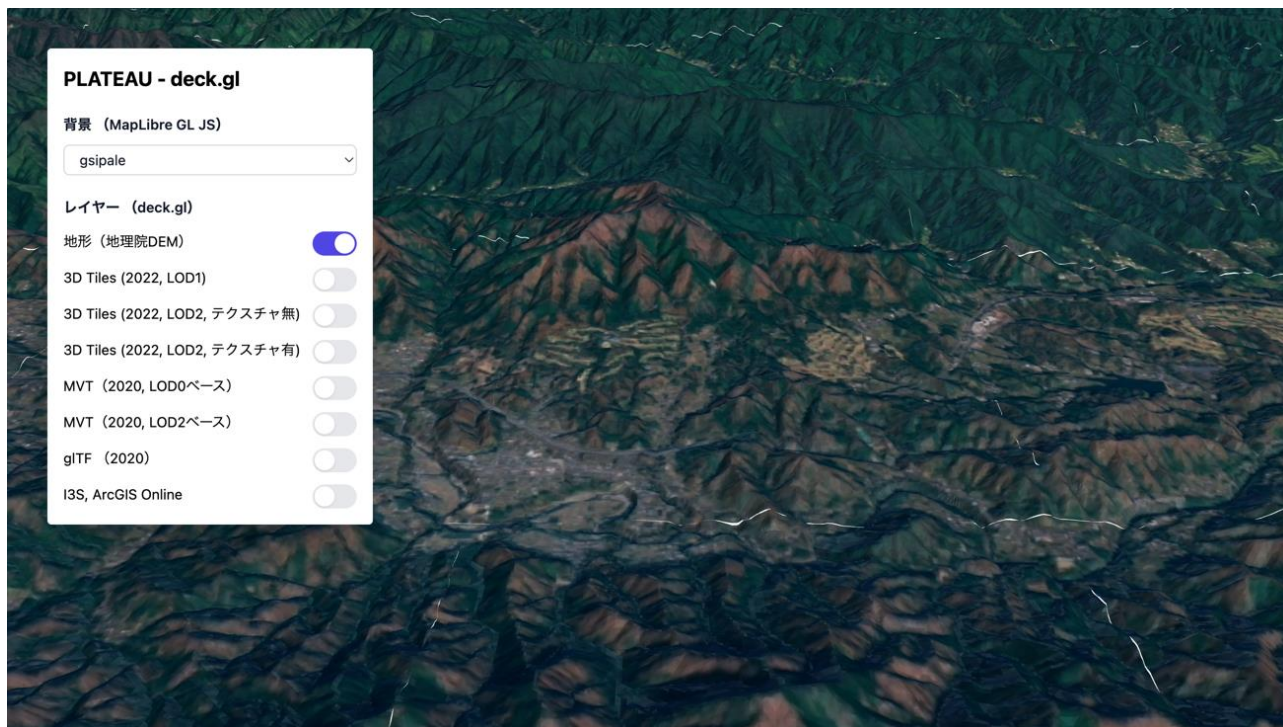


図 3-17 deck.gl - 地形表現

デモアプリケーション STEP2 – deck.gl、地形（地理院 DEM）

<https://project-plateau.github.io/mapengine-survey/step2/deckgl/index.html>

例では地理院標高タイルを利用し、3D 地形を表示した。MapLibre GL JS での例と同様に、このタイルは独自形式のためそのまま利用ができない。そのため、これを変換して利用するためのプラグイン（レイヤー）を追加した。

MapLibre GL JS での例では地形（標高）自体を設定するが、deck.gl では個別のレイヤーとして、標高情報と背景地図タイル（航空写真）をあわせて表示するという形になる。

提供されている地理院標高タイルは、最大ズームレベルが 14 のため、maxZoom 設定もそのように設定する必要がある。このとき、背景地図タイル（航空写真）もあわせて、最大ズームレベルが 14 と設定されるため、14 を超えてズームインした際に、画質が荒くなってしまう。また、このようにタイルを用いた例では、「図 3-8 deck.gl - 地形表現」内に白い線が入っている様に、タイルの切れ目が見えてしまうケースがある。

deck.gl で、後述する Tile3DLayer を用いて 3D Tiles を表示するとき、そのままでは建築物モデルが地表面と接地せず浮いてしまうことがある。これはジオイド高のためであり、これを考慮して Z 方向の調整が必要となる。

3D 都市モデルで考慮されている地形（DEM）は解像度が高く、狭い範囲でもそれなりに起伏が表現されており、建築物モデルはその起伏にしたがって配置される。

他方、国土地理院標高タイルをはじめとする、配信されており入手容易な DEM データは、解像度が 10m 程度と PLATEAU の地形よりも荒い。そのため、それらを組み合わせたときには、厳密な「接地」は難しい。

3-5-3-c. 3D 都市モデルの描画: LOD 切り替え

deck.gl では標準の Tile3DLayer により、3D Tiles の描画が可能である。

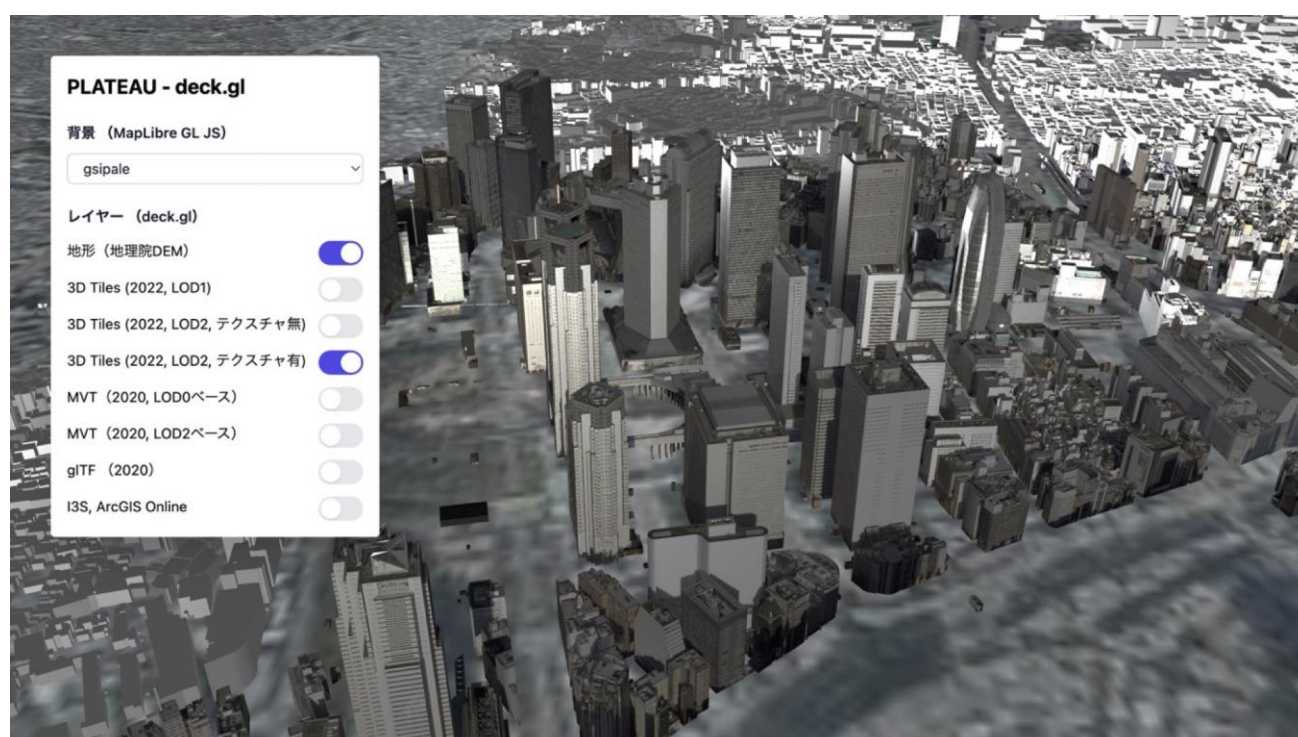


図 3-18 deck.gl - 建築物モデルと地形の表現

デモアプリケーション STEP2 – deck.gl、地形（地理院 DEM） + 3D Tiles(2022,LOD2,テクスチャ有)

<https://project-plateau.github.io/mapengine-survey/step2/deckgl/index.html>

WebGIS に関する技術調査レポート

また、MVTLayer、もしくは MapLibre GL JS および Mapbox GL JS との併用により、MVT の描画が可能であり、このとき高さ情報をもとに押出表現を行うことができる。

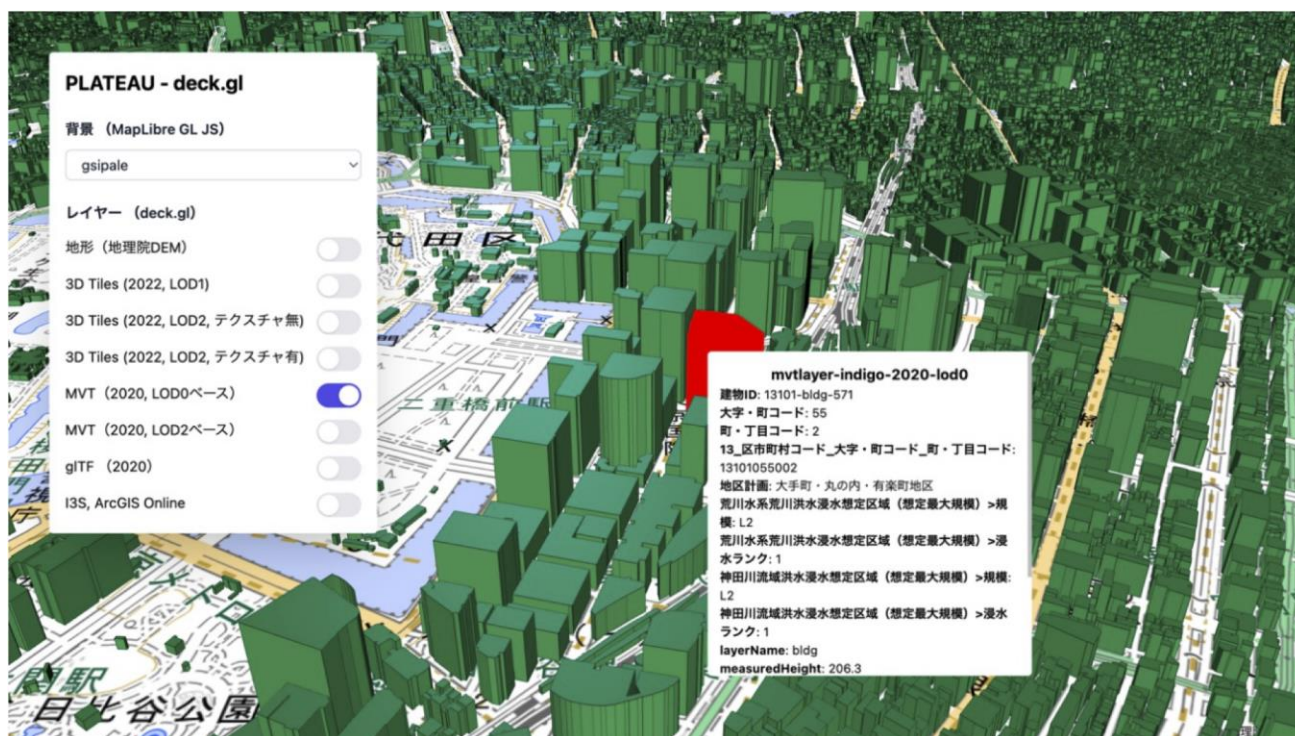


図 3-19 deck.gl - 建築物モデル表現 (MVT)

デモアプリケーション STEP2 - deck.gl、3D Tiles (2020, LOD0 ベース)

<https://project-plateau.github.io/mapengine-survey/step2/deckgl/index.html>

deck.gl に用意されている ScenegraphLayer により、glTF を表示可能である。

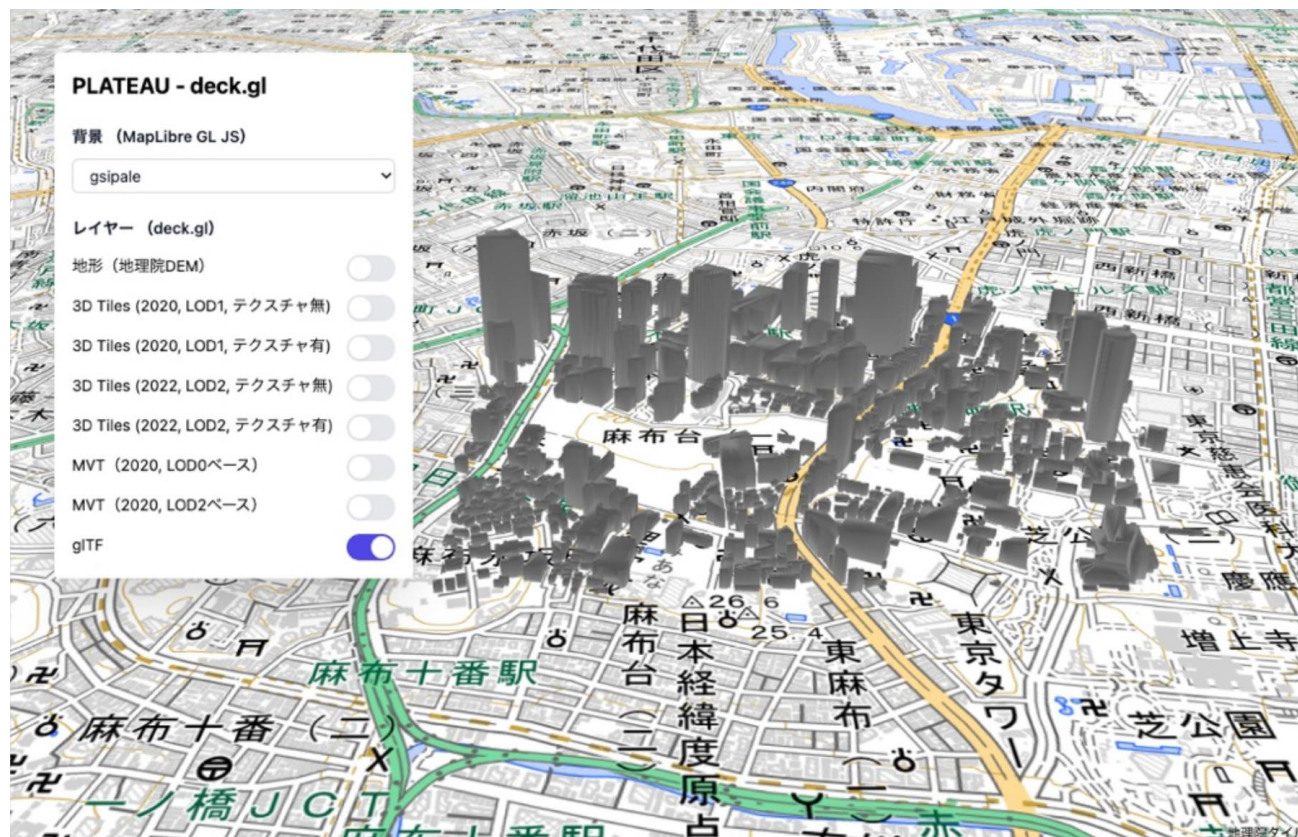


図 3-20 deck.gl - 建築物モデル表現 (glTF)

デモアプリケーション STEP2 - deck.gl、glTF

<https://project-plateau.github.io/mapengine-survey/step2/deckgl/index.html>

ArcGIS で主に利用されるデータ形式の I3S も、3DTiles と同様に Tile3DLayer によって表示可能である。



図 3-21 deck.gl - 建築物モデル表現 (I3S)

デモアプリケーション STEP2 – deck.gl、I3S ArcGIS Online

<https://project-plateau.github.io/mapengine-survey/step2/deckgl/index.html>

3-5-3-d. 3D 都市モデルの描画: テクスチャ

テクスチャが付与された 3D Tiles を描画した場合、問題なくそれらが建築物モデル上に描画される。

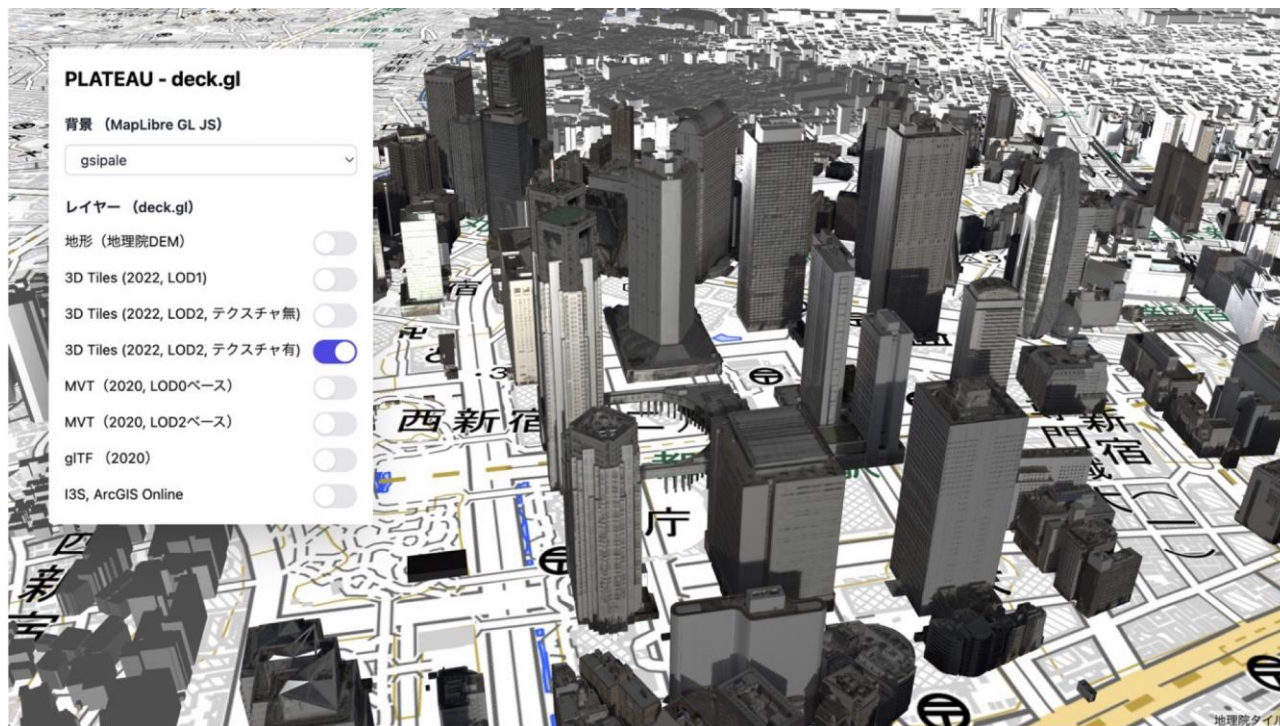


図 3-22 建築物モデル表現 (3D Tiles)

デモアプリケーション STEP2 – deck.gl、3D Tiles(2022,LOD2,テクスチャ有)

<https://project-plateau.github.io/mapengine-survey/step2/deckgl/index.html>

3-5-3-e. 3D 都市モデルの属性情報

3D Tiles の場合、クリックした建築物モデル単位での属性の取得はできず、タイル単位での情報となってしまう。

今回利用したデータでは、3D Tiles に含まれるコンテンツは Batched 3D Model (b3dm) という形式である (FME が出力した 3D Tiles 1.0)。

例えば PLATEAU VIEW で使用されている CesiumJS では、b3dm に埋め込まれている建築物モデルの属性データ (featureTable、batchTable) を読み、tileset.json 自体に埋め込まれている属性表示用のテンプレートを元に、属性情報を表示している。このような独自実装により、CesiumJS では属性情報を表示することができる。

deck.gl の場合は、CesiumJS と同等の処理が実装されていないために、個別の建築物モデルは特定できるが、それに対応する個別のデータを特定することができず、結果、それぞれの属性情報をクリックして表示するといったことができない。

次世代の仕様である 3D Tiles Next (3D Tiles 1.1) では、b3dm や i3dm ではなく、直接 glTF (バイナリ表現である GLB も含む) を参照することができるようになると述べられている。この形式であれば、glTF 自体に属性を埋め込むことができる。現在の CesiumJS はこの形式に対応しているため、独自実装なしに属性を読み込むことが可能である。他方で、deck.gl は 3D Tiles 1.1 に対応していないため、属性情報を読み込むことができない。

MVT による押出表現の場合は、MapLibre GL JS の節と同じく、建築物モデルごとの属性情報の取得が可能である。

3-5-3-f. レンダリングパフォーマンス

deck.gl は大量のデータを扱うデータ可視化ライブラリを標榜しており、全体的な動作は軽快である。しかし、3D Tiles レイヤーを追加すると、CesiumJS ほどのレンダリングパフォーマンスは出ず、実感できるほど遅い挙動となってしまうことが確認された。これは特にテクスチャ付きの 3D Tiles データであるときに顕著である。これは、CesiumJS が 3D Tiles に特化しており、描画最適化などの処理を行っており、他方で deck.gl の Tile3DLayer は多くある標準レイヤーの一部であり、開発が活発とは言えないといったことが理由として想定される。

3-5-3-g. 考察

deck.gl は、3D Tiles や MVT などを行うことができ、それらにより建築物モデルや地形の描画が可能である。また、MapLibre GL JS や Mapbox GL JS とともに 3D Tiles データを用いたいときも、単独では現実的な利用はできないが、この deck.gl と併用することで容易に可能となる。

加えて deck.gl には、他のライブラリにはない独自の様々なデータ可視化レイヤーが標準で存在する。そのため、一定以上の可視化表現を行いたい際には有力な選択肢となる。

他方で、3D Tiles の描画パフォーマンスや属性情報の扱いは CesiumJS に比べて劣るため、シンプルに 3D Tiles を描画したい場合には後者が有力な選択肢となる。

3-5-3-h. 参考文献

表 3-9 deck.gl 参考文献

タイトル	出典
React Map GL	https://visgl.github.io/react-map-gl/
deckgl-gsi-terrain-layer GitHub ページ	https://github.com/Kanahiro/deckgl-gsi-terrain-layer
地理院タイル一覧	https://maps.gsi.go.jp/development/ichiran.html
Introducing 3D Tiles Next, Streaming Geospatial to the Metaverse	https://cesium.com/blog/2021/11/10/introducing-3d-tiles-next/#integrating-with-the-gltf-ecosystem
MVTLayer	https://deck.gl/docs/api-reference/geo-layers/mvt-layer
ScenegraphLayer	https://deck.gl/docs/api-reference/mesh-layers/scenegraph-layer
Tile3DLayer	https://deck.gl/docs/api-reference/geo-layers/tile-3d-layer
I3SLoader	https://loaders.gl/docs/modules/i3s/api-reference/i3s-loader

3-5-4. iTowns

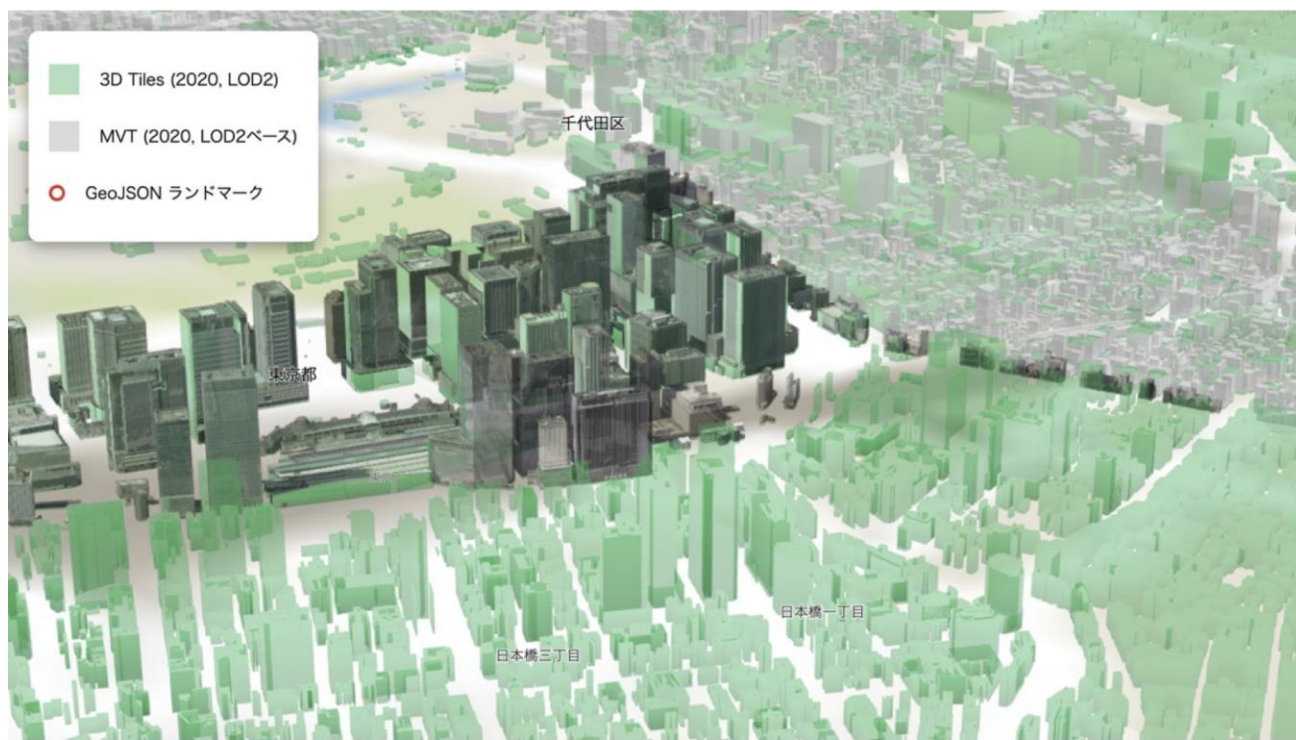


図 3-23 iTowns - 建築物モデル表現 (3D Tiles、MVT)

デモアプリケーション STEP2 - iTowns

<https://project-plateau.github.io/mapengine-survey/step2/itowns/index.html>

3-5-4-a. 背景地図

ラスタータイルおよび、CesiumJS では対応していないベクタータイルによる背景地図の表示が可能である。



図 3-24 iTowns - 背景地図表現

デモアプリケーション STEP2 – iTowns

<https://project-plateau.github.io/mapengine-survey/step2/itowns/index.html>

3-5-4-b. 地形・高さ情報

WMTS 形式のデータを用いて、3D 地形の表示が可能である。

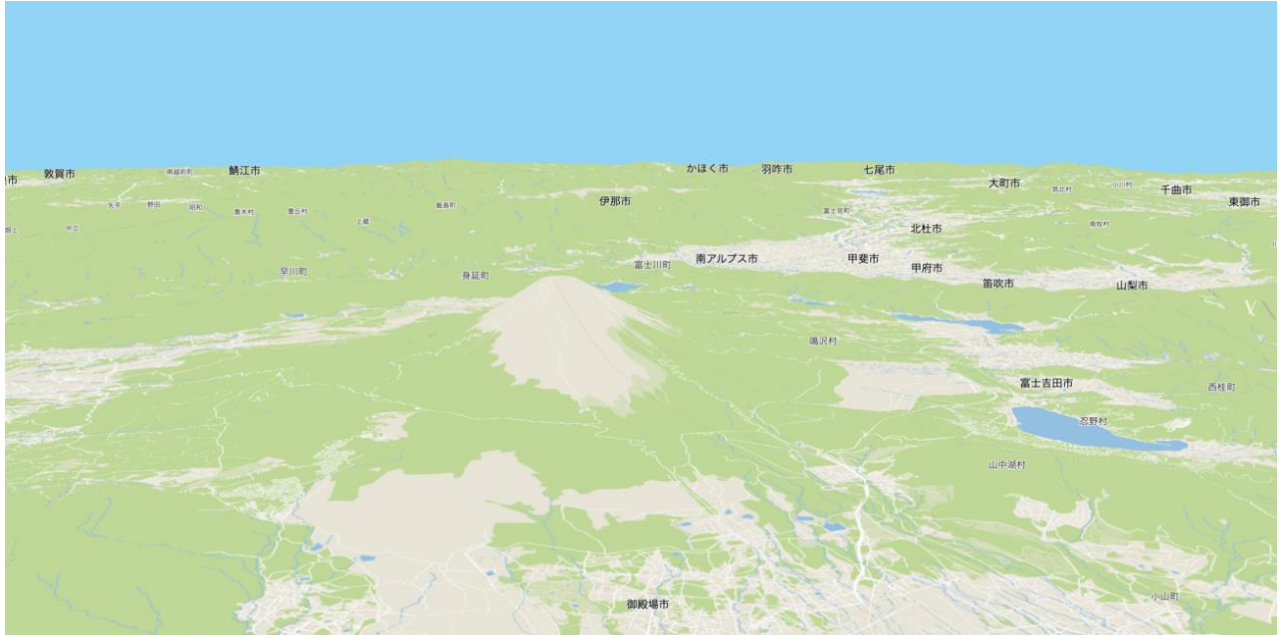


図 3-25 iTowns - 地形表現

デモアプリケーション STEP2 – iTowns

<https://project-plateau.github.io/mapengine-survey/step2/itowns/index.html>

3-5-4-c. 3D 都市モデルの描画: LOD 切り替え

3D Tiles の描画が可能である。また、MVT による押出表現も可能である。押出の高さは、MapLibre GL JS などと同様に属性情報を元にする事が可能である。

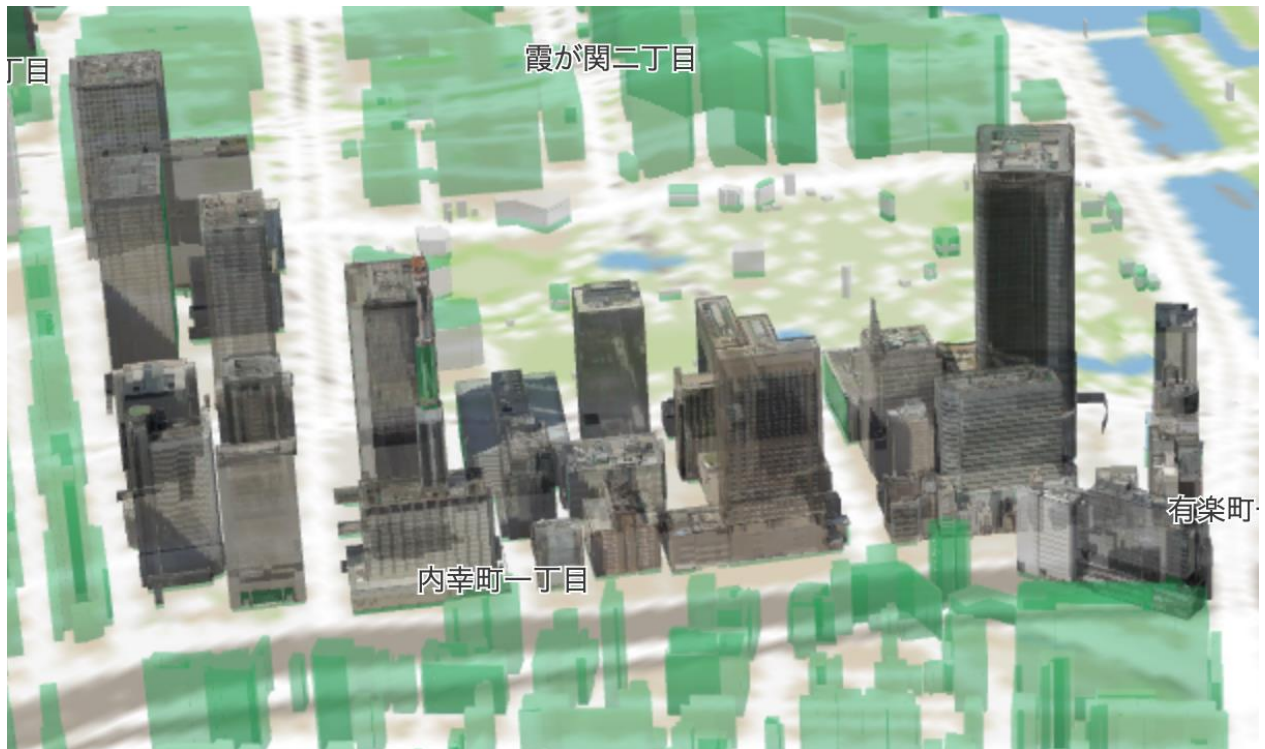


図 3-26 iTowns - 建築物モデル表現 (3D Tiles テクスチャあり)

デモアプリケーション STEP2 - iTowns

<https://project-plateau.github.io/mapengine-survey/step2/itowns/index.html>

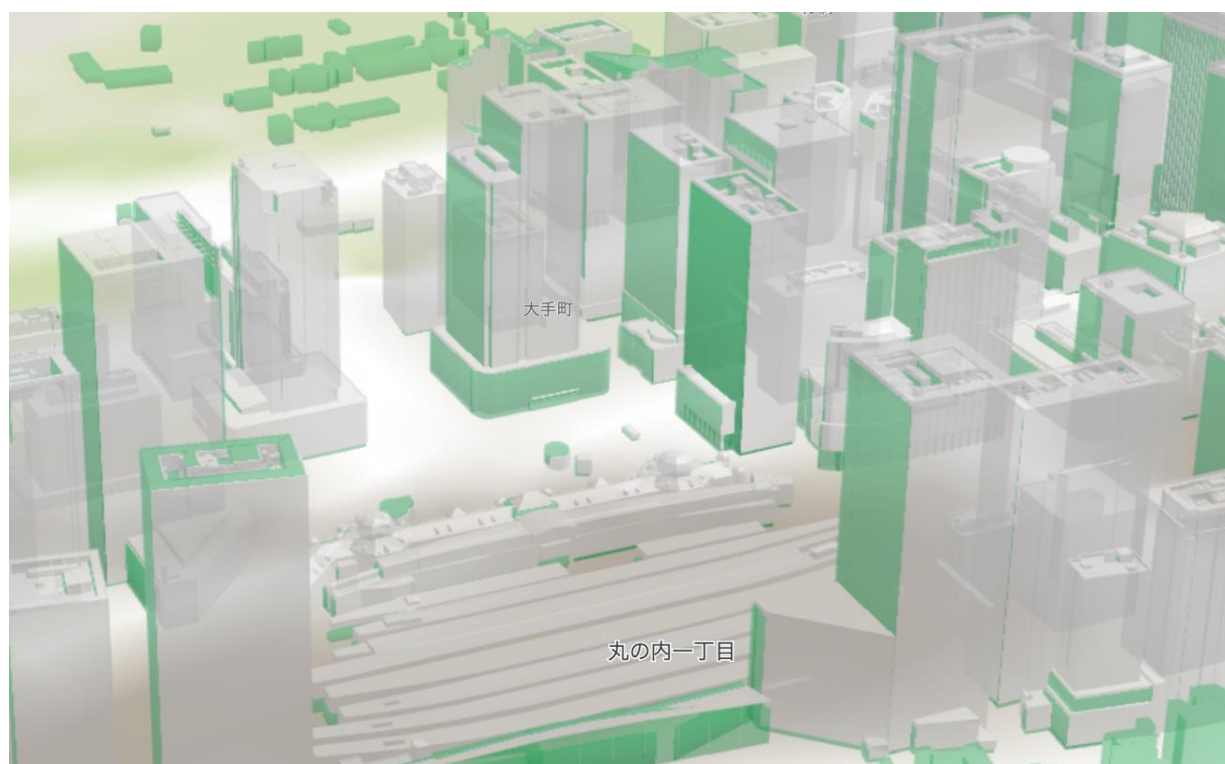


図 3-27 iTowns - 建築物モデル表現 (MVT)

デモアプリケーション STEP2 - iTowns

<https://project-plateau.github.io/mapengine-survey/step2/itowns/index.html>

3-5-4-d. 3D 都市モデルの描画: テクスチャ

テクスチャが付与された 3D Tiles を描画した場合、問題なくそれらが建築物モデル上に描画される。

3-5-4-e. 3D 都市モデルの属性情報

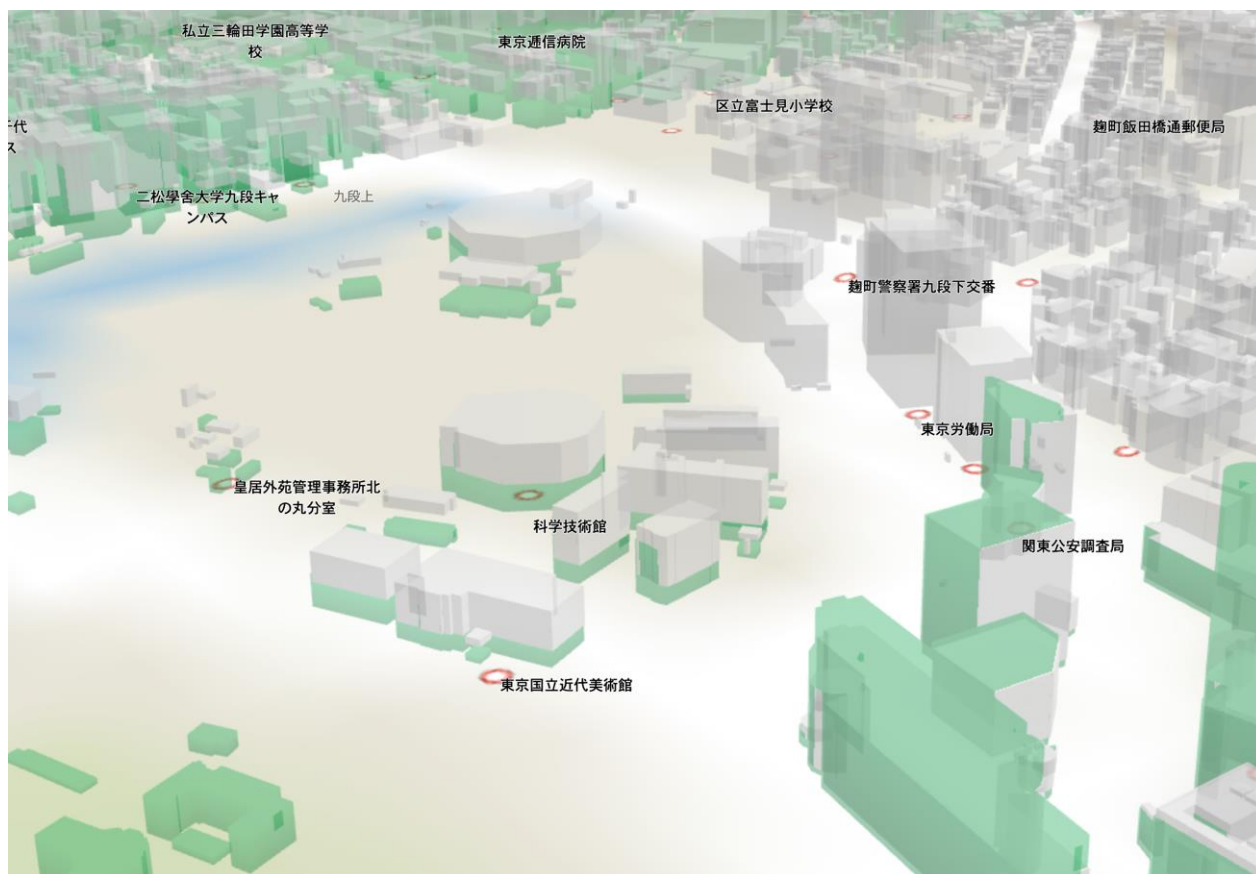


図 3-28 iTowns - 属性情報の表示 (GeoJSON)

デモアプリケーション STEP2 - iTowns

<https://project-plateau.github.io/mapengine-survey/step2/itowns/index.html>

クリックイベントなどにより建築物モデルを選択した属性取得は、調査の限りでは実現方法が存在しなかった。

補足として、iTownns では GeoJSON を表示することが可能であり、その際に属性情報をテキストラベルとして表示することが可能である。

3-5-4-f. レンダリングパフォーマンス

iTowns のレンダリングパフォーマンスは、CesiumJS や、MapLibre GL JS、deck.gl と比べて著しく悪い。これは、建築物モデルだけでなく、背景地図の描画に関しても同様である。他のライブラリに比べて開発が活発ではなく、また Three.js というライブラリをベースとしており内部的なパフォーマンスチューニングが不十分であるが原因として考えられる。

3-5-4-g. 考察

iTowns では、3D Tiles や MVT による建築物モデルの表示が可能である。

しかし、読込速度が明らかに遅いなどパフォーマンスの問題が顕著であり、利便性を大きく損なう。また、属性情報をうまく扱うことができない。

また、ユーザー数が少なく事例や情報が不足しており、特に日本語の情報がほとんど存在しないため、開発の難易度が高い。

これらの状況に加えて、MapLibre GL JS や deck.gl と比較して、iTowns で可能な独自の表現や機能が基本的に存在しないため、あえてこの地図エンジンを選択する強い理由が存在しないと判断した。

3-5-5. 3DTilesRendererJS

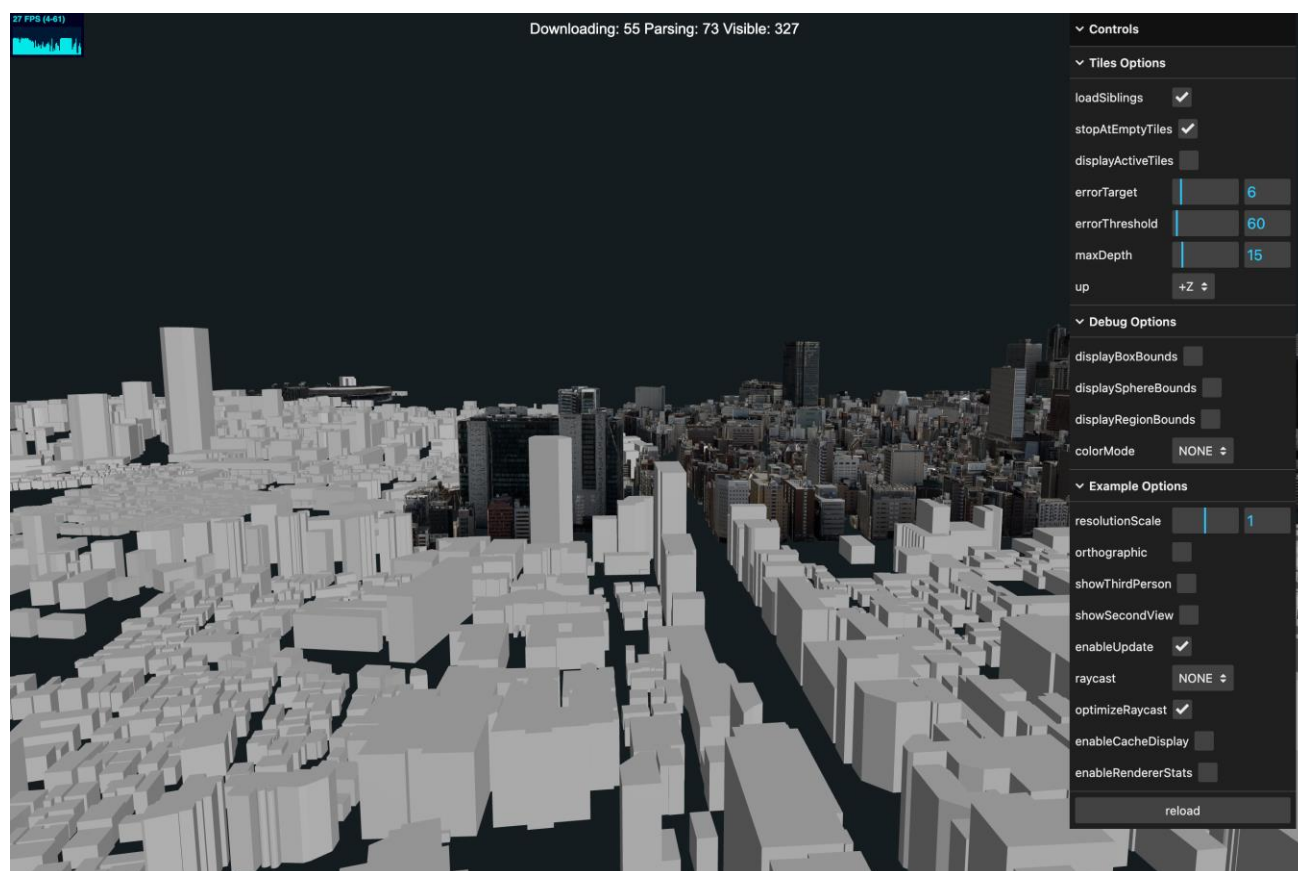


図 3-29 デモアプリケーション STEP2 – NASA-AMMOS/3DTilesRendererJS

デモアプリケーション STEP2 – NASA-AMMOS/3DTilesRendererJS

<https://nasa-ammos.github.io/3DTilesRendererJS/example/bundle/index.html> -

https://assets.cms.plateau.reearth.io/assets/57/e338bc-4e38-447c-a82e-0f4a7b075ed0/13100_tokyo23-ku_2022_3dtiles_1_1_op_bldg_13104_shinjuku-ku_lod2/tilesset.json

3-5-5-a. 背景地図

3D Tiles を描画する機能に特化しているため、他の地図エンジンと異なり、ラスタータイルやベクタータイルなどによる背景地図を描画する方法は用意されていない。

3-5-5-b. 地形・高さ情報

他の地図エンジンと異なり、地理空間情報を扱う機能が用意されておらず、表示データと他の地理空間情報を結びつけて容易に利用することができない。

3-5-5-c. 3D 都市モデルの描画: LOD 切り替え

以下のように、3D Tiles を描画することが可能である。

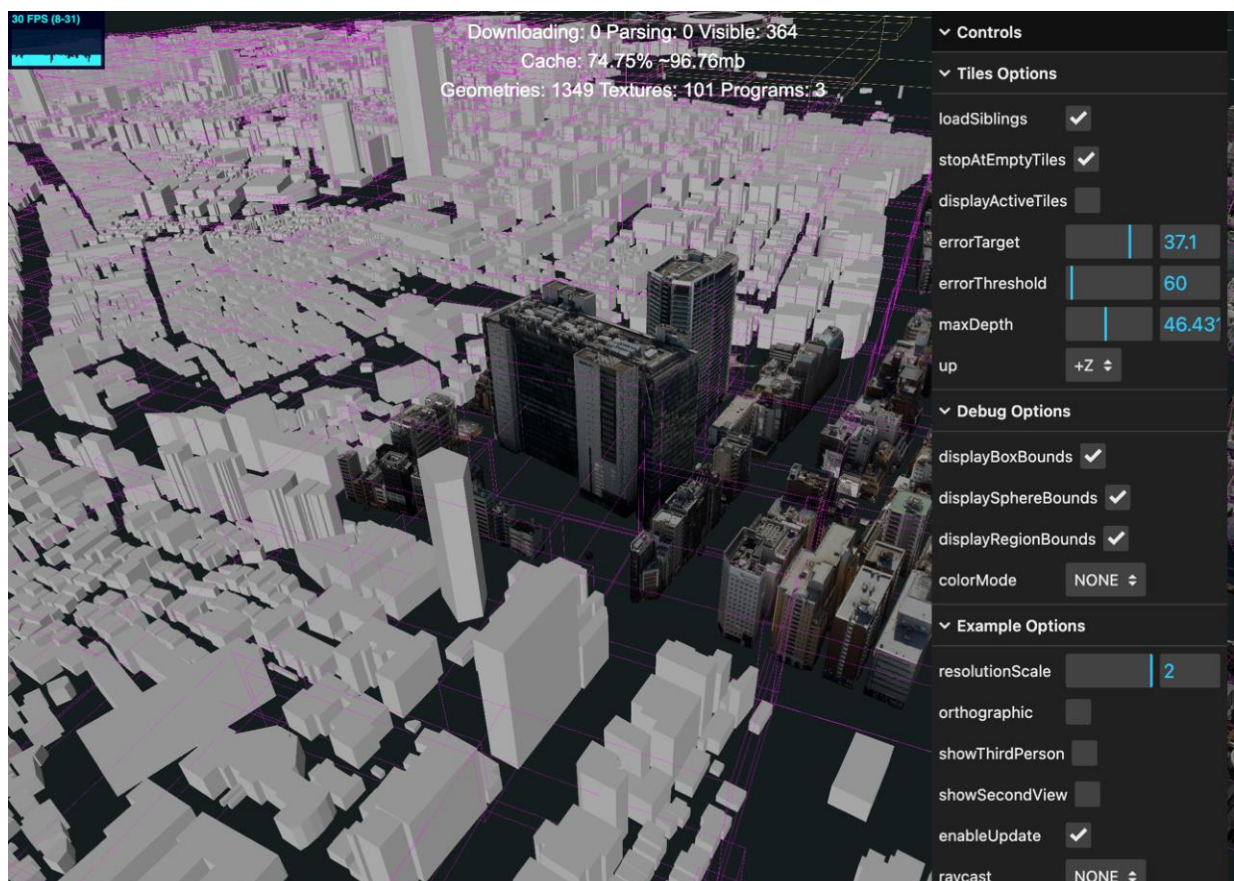


図 3-30 3DTilesRendererJS - 建築物モデル表示

デモアプリケーション STEP2 - NASA-AMMOS/3DTilesRendererJS

<https://nasa-ammos.github.io/3DTilesRendererJS/example/bundle/index.html> -

https://assets.cms.plateau.reearth.io/assets/57/e338bc-4e38-447c-a82e-0f4a7b075ed0/13100_tokyo23-ku_2022_3dtiles_1_1_op_bldg_13104_shinjuku-ku_lod2/tileset.json

3-5-5-d. 3D 都市モデルの描画: テクスチャ

テクスチャが付与された 3D Tiles を描画した場合、それらが問題なく建築物モデル上に描画される。

3-5-5-e. 3D 都市モデルの属性情報

建築物モデルをクリック選択するなどして属性情報を取得する方法は用意されていない。

3-5-5-f. レンダリングパフォーマンス

他の地図エンジンと比べて機能が限定的なこともあり、3D Tiles の描画に関してある程度軽快に動作する。

3-5-5-g. 考察

3D Tiles の描画自体は問題なく動作するが、背景地図を合わせて描画したり、既存の地理空間情報と容易に組み合わせたりする機能が標準で用意されておらず、そのままでは GIS の高度な利用を行うことができない。

Three.js を用いた高度な可視化表現を自ら実装する際、そこへ 3D Tiles を組み合わせる際にこの地図エンジンを併用するといったユースケースを想定することはできるが、一般的なユーザーの用途には適していない。

3-5-6. ArcGIS Maps SDK for JavaScript

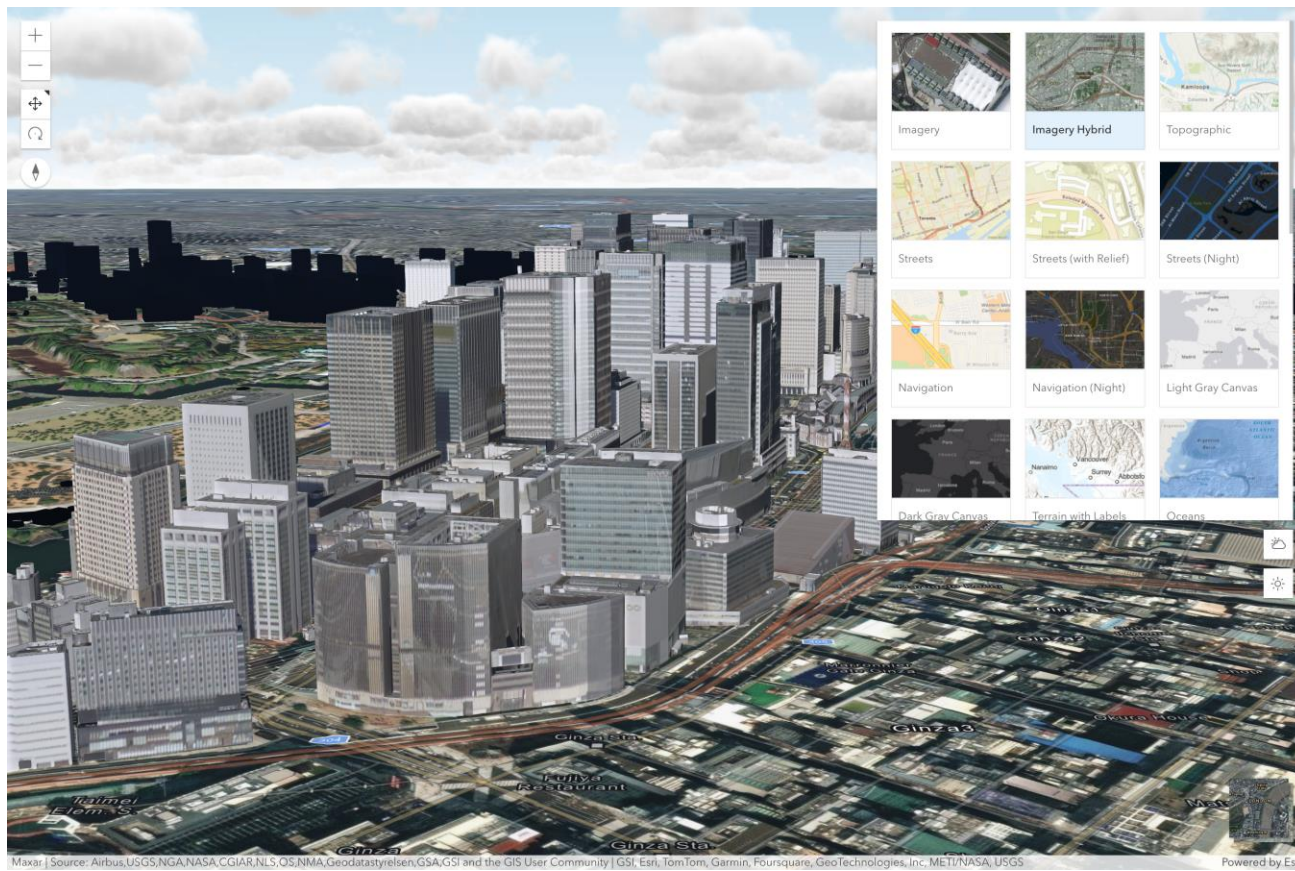


図 3-31 ArcGIS Maps SDK for JavaScript - 建築物モデル表示
デモアプリケーション STEP2 -ArcGIS Maps SDK for JavaScript

3-5-6-a. 背景地図

Esri 社からさまざまな背景地図が提供されている。

2つの背景地図を切り替えるウィジェット（BaseMapToggle、例では画面右下のサムネイル）や、一覧から背景地図を選択できるウィジェット（BasemapGallery、例では画面右上）も提供されている。

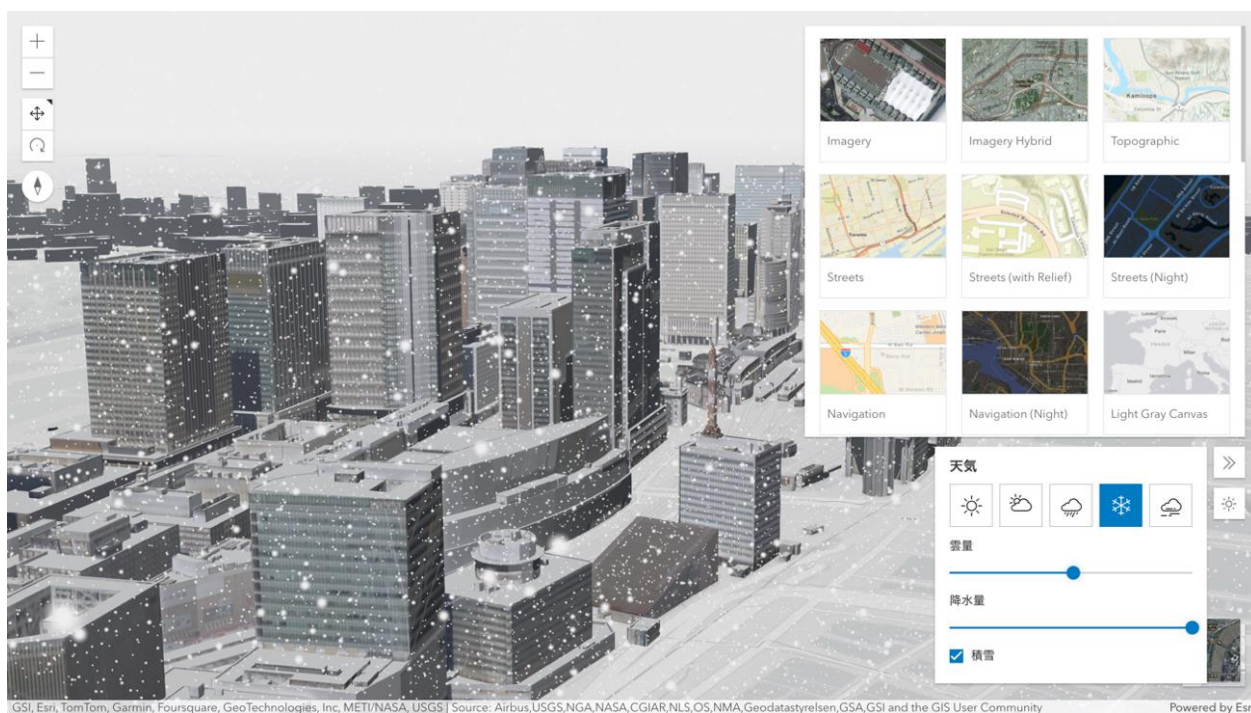


図 3-32 ArcGIS Maps SDK for JavaScript - 天気の可視化
デモアプリケーション STEP2 –ArcGIS Maps SDK for JavaScript

晴れや曇り、雨天や降雪、霧といった表現を画面上に表示することができる。雲量や降水量・降雪量、霧の密度を調整することもできる。建築物モデル上へ積雪を表示することも可能である。またそれらを調整するためのウィジェットも用意されている。

3-5-6-b. 地形・高さ情報

標準で用意されている ElevationLayer を用いることで 3D 地形の表現が可能である。このレイヤー自体は可視ではないが、この上に背景地図を載せることで、3D 地形を確認することができる。またその上に位置させるレイヤーは、絶対的な高度か相対的な高度かを選択することができる。

3-5-6-c. 3D 都市モデルの描画: LOD 切り替え

標準で用意されている SceneLayer を用いて、I3S フォーマットによる 3D データの表示ができる。3D オブジェクトやポイントの他にも、建物（BIM）やメッシュ、点群やボクセルといったレイヤーも存在する。

- A guide to SceneLayers ArcGIS Maps SDK for JavaScript 4.29,
<https://developers.arcgis.com/javascript/latest/working-with-scene-layers/>
- 3D オブジェクト、ポイント（SceneLayer）：
<https://developers.arcgis.com/javascript/latest/api-reference/esri-layers-SceneLayer.html>
- 建物、BIM（BuildingSceneLayer）：<https://developers.arcgis.com/javascript/latest/api-reference/esri-layers-BuildingSceneLayer.html>
- メッシュ（IntegratedMeshLayer）：<https://developers.arcgis.com/javascript/latest/api-reference/esri-layers-IntegratedMeshLayer.html>
- 点群（PointCloudLayer）：<https://developers.arcgis.com/javascript/latest/api-reference/esri-layers-PointCloudLayer.html>
- ボクセル（VoxelLayer）：<https://developers.arcgis.com/javascript/latest/api-reference/esri-layers-VoxelLayer.html>

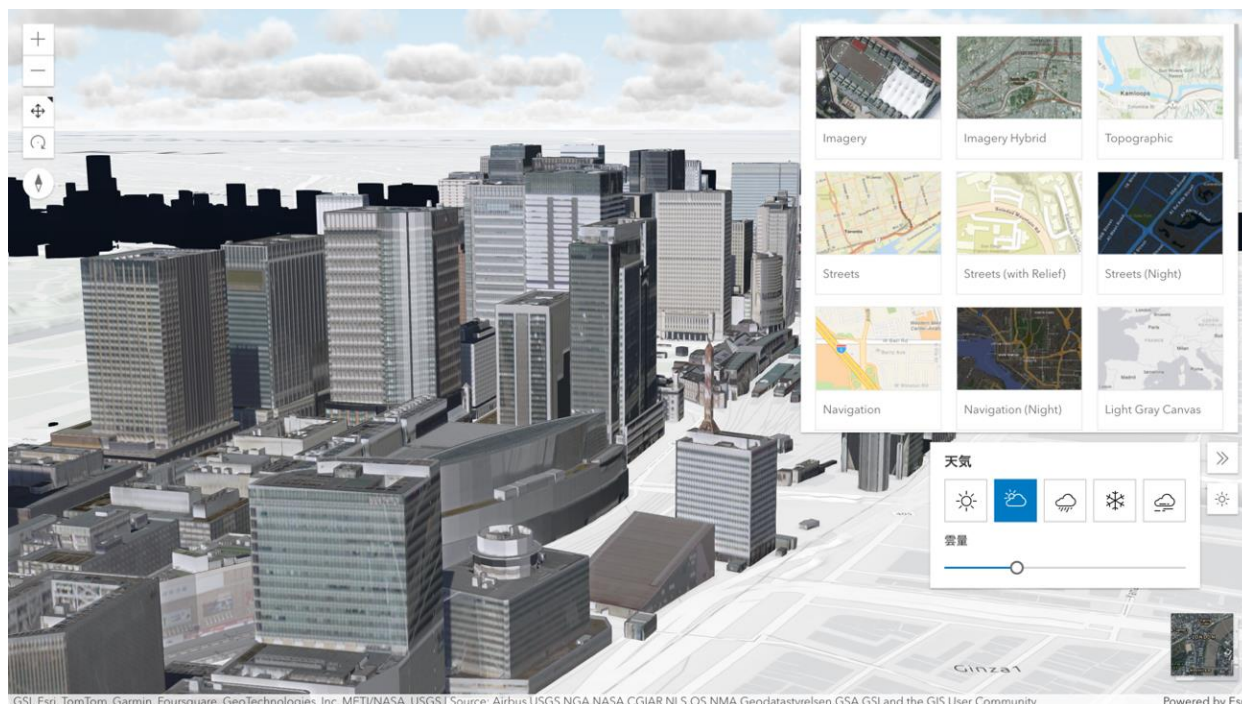


図 3-33 ArcGIS Maps SDK for JavaScript - 建築物モデル表示
デモアプリケーション STEP2 –ArcGIS Maps SDK for JavaScript

3-5-6-d. 3D 都市モデルの描画: テクスチャ

I3S データ形式を用いる際、テクスチャも描画することが可能である。

3-5-6-e. 3D 都市モデルの属性情報

クリックによる建築物モデル選択と属性情報の取得は可能であるが、今回検証したデータと実装の範囲では、場所によって複数の建築物モデルがまとめて選択されてしまい、想定する形での建築物モデルの属性情報取得は可能とならなかった。

3-5-6-f. レンダリングパフォーマンス

I3S は、3D Tiles と同様にタイル化されたデータであり、広範囲でも軽快に動作することが確認された。また deck.gl での例とは異なり、テクスチャが付与されたデータでも顕著なパフォーマンス劣化は見られなかった。

3-5-6-g. その他の機能

他の地図エンジンには見られない様々な機能が用意されており、ドキュメントでも利用例が紹介されている。以下に主なものをいくつか列挙する。

- 照準線: Line of sight widget Sample Code ArcGIS Maps SDK for JavaScript 4.29, <https://developers.arcgis.com/javascript/latest/sample-code/widgets-line-of-sight/>
- インタラクティブなジオメトリ作成: SceneLayerView - query statistics by geometry Sample Code ArcGIS Maps SDK for JavaScript 4.29, <https://developers.arcgis.com/javascript/latest/sample-code/layers-scenelayerview-query-stats/>
- 標高断面: ElevationProfile Widget Sample Code ArcGIS Maps SDK for JavaScript 4.29, <https://developers.arcgis.com/javascript/latest/sample-code/widgets-elevation-profile/>
 - ✧ 3D 編集: Edit features in 3D with the Editor widget Sample Code ArcGIS Maps SDK for JavaScript 4.29, <https://developers.arcgis.com/javascript/latest/sample-code/widgets-editor-3d/>

3-5-6-h. 考察

Esri 社製品群の一部であり、ドキュメント、事例、サポート体制が充実している。

また、他の地図エンジンではできない、天気や照準線、標高断面といった機能やウィジェットが用意されている。

しかし SDK の利用には、ArcGIS へのサブスクリプション（API キー）が必要である。ArcGIS のオンラインサービスと密に連携しているため、MapLibre GL JS や deck.gl などのように地図エンジンとして単体で利用することには適さない。

また、3D データを表示するための SceneLayer を利用するには、ArcGIS のサービスを利用する必要がある。具体的には、SceneLayer へ指定する URL パラメータには、GIS サーバーである ArcGIS Enterprise のためのプロダクト（ArcGIS Server、Portal for ArcGIS）もしくは、クラウドプラットフォームである ArcGIS Online のリソースを指定する必要がある。今回の調査では、ローカル環境で変換した SLPK ファイルを、ArcGIS Online へアップロードし、ホスト シーン レイヤーを作成し、それを JavaScript SDK で表示した。

元々 Esri 社の製品を利用しておりライセンスを保持しており、ArcGIS 製品群に慣れ親しんでいたりと、データがそのプラットフォーム上へすでに存在したりする場合は、この地図エンジンの利用が適している。

しかし、オープンソースで無償利用可能な他の地図エンジンがあり、それらも一定以上の機能を保持しているため、ArcGIS ライセンスを持たない一般的なユーザーの利用用途にとっては、これが第一の選択肢候補とはなりづらいと判断した。

3-5-6-i. 参考文献

表 3-10 ArcGIS 参考文献

タイトル	出典
Display a map ArcGIS Maps SDK for JavaScript 4.29	https://developers.arcgis.com/javascript/latest/tutorials/display-a-map/
Display a scene ArcGIS Maps SDK for JavaScript 4.29	https://developers.arcgis.com/javascript/latest/tutorials/display-a-scene/
BasemapToggle API Reference ArcGIS Maps SDK for JavaScript 4.29	https://developers.arcgis.com/javascript/latest/api-reference/esri-widgets-BasemapToggle.html
BasemapGallery API Reference ArcGIS Maps SDK for JavaScript 4.29	https://developers.arcgis.com/javascript/latest/api-reference/esri-widgets-BasemapGallery.html
Weather visualization Sample Code ArcGIS Maps SDK for JavaScript 4.29	https://developers.arcgis.com/javascript/latest/sample-code/scene-weather/
SceneLayer API Reference ArcGIS API for JavaScript 4.4,	https://totalapis.github.io/api-reference/esri-layers-SceneLayer.html#url

4. 表現方法の調査

4-1. 目的

STEP3 では、「表現方法の調査」として、選定した地図エンジンを用い、3D 都市モデルと他の GIS データを組み合わせた動的な表現や美しい表現を調査する。これらの高度な表現は一般に CesiumJS が苦手とされている。

STEP2 で行われた検証の結果、評価項目である 3D 都市モデルの描画やレンダリングパフォーマンスといった観点から、deck.gl と MapLibre GL JS が 3D 都市モデルの利用環境として有望であるといえる。また、これら 2 つの地図エンジンは、ヒートマップやクラスターといった CesiumJS で行うことが容易ではないデータ可視化表現ができる可能性がある。

そこで STEP3 では、この 2 つの地図エンジンを用いたアプリケーション環境を構築し、実際に 3D 都市モデルと組み合わせて新たなデータ表現を試み、その有用性を調査する。

4-2. 調査方法

STEP3 では deck.gl と MapLibre GL JS を対象に、公式サンプル等を参考にしてデータ可視化のデモアプリケーションを制作しながら、表現のバリエーションや描画パフォーマンス面での違いを検証する。deck.gl では、3D Tiles 形式の 3D 都市モデルと組み合わせて、deck.gl の様々な可視化機能を用いたデモアプリケーションを作成する。

MapLibre GL JS では、3D 都市モデルを MVT による押出で表現し、それと組み合わせて MapLibre GL JS が提供する可視化機能を用いたデモアプリケーションを作成する。

4-3. 調査結果のまとめ

deck.gl では、3D Tiles により 3D 都市モデルを表示し、あわせて deck.gl が提供するデータ可視化機能を用いて、以下の 5 事例を作成した。

- 条件に応じた建物色の変化（TerrainExtension 機能）
- 経路表示（LineLayer）
- 移動体の走行軌跡（TripsLayer）
- 拠点間の移動状況（ArcLayer）
- ヒートマップ（HeatmapLayer）

利用する地理空間情報データは、GeoJSON などの定型的な形式ではなく、プログラムに合わせて用意する必要があった。

ただし、deck.gl による 3D Tiles の描画は、広範囲を表示すると描画パフォーマンスが CesiumJS に比べ顕著に遅かった。

MapLibre GL JS では、MVT による押出表現とともに、MapLibre GL JS が提供するデータ可視化機能を用いて、以下の事例を作成した。

- ヒートマップ
- クラスター

MapLibre GL JS では、一般的な形式である GeoJSON データがそのまま利用可能であった。

MVT 押出と組み合わせた表現は、CesiumJS や deck.gl での 3D Tiles を用いた例と比べ高速に動作した。

これら deck.gl と MapLibre GL JS を用いた高度なビジュアル表現は、CesiumJS では容易に実現できないものであり、3D 都市モデルと地理空間情報を組み合わせた新たな活用を可能とする。

ただし deck.gl は、CesiumJS と同様に 3D Tiles の利用が可能であるが、パフォーマンスでは劣るため、単に 3D 都市モデルを表示したい場合には、CesiumJS の方が用途に適する。

一方で、3D Tiles による精密な建築物モデル表示が主体ではなく、地図の一部として簡易に建築物を表示したいのであれば、MapLibre GL JS による MVT 押出が、軽量に動作することもあり、用途に適している。

4-4. deck.gl の活用

4-4-1. deck.gl - 条件に応じた建物色変化

4-4-1-a. 概要

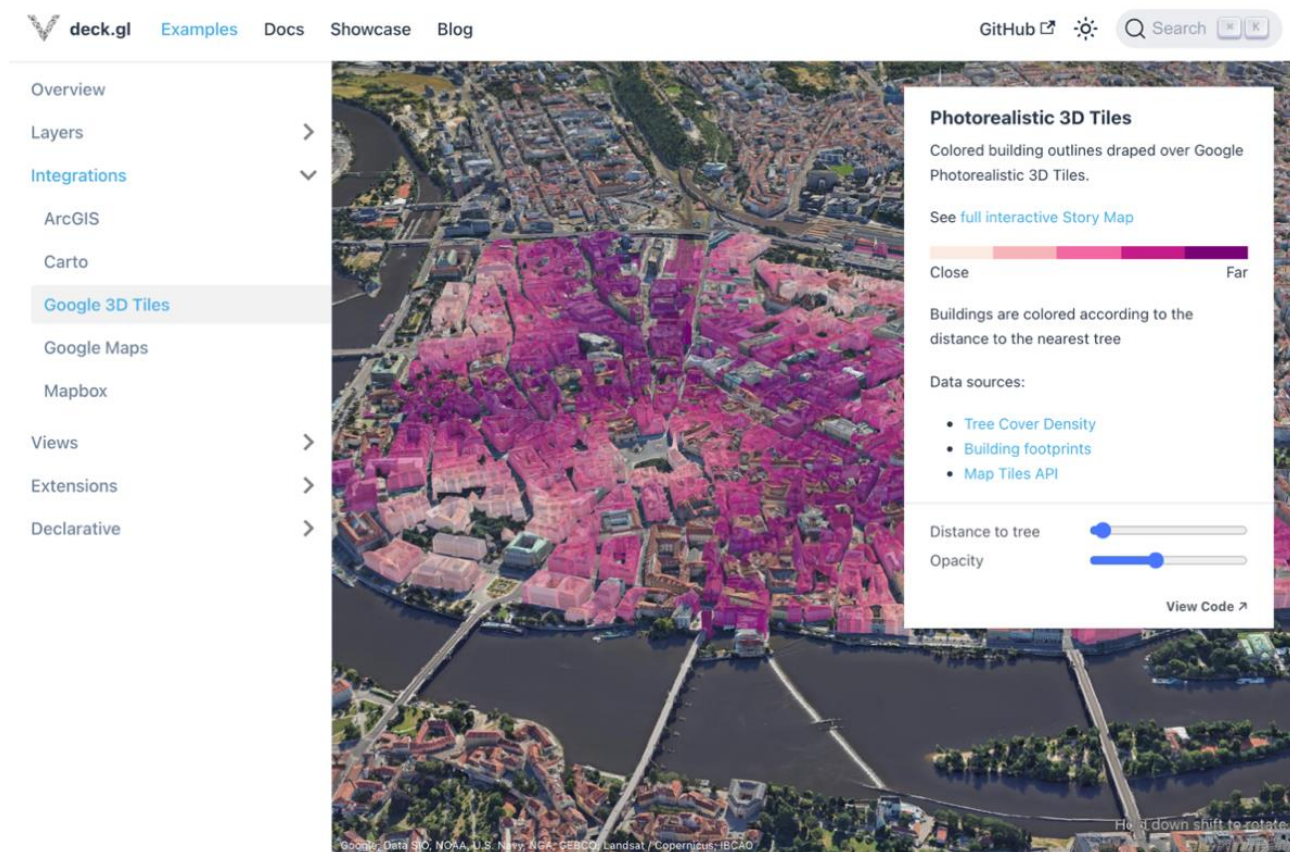


図 4-1 deck.gl 公式 Examples - 建物色変化の事例

deck.gl の Google Photorealistic 3D Tiles の実装例 (<https://deck.gl/examples/google-3d-tiles>)

deck.gl では TerrainExtension 機能により、2D のデータを、3D 表面上に沿って描画することが可能である。

例えば上記のサンプルでは、この機能を用い、街路樹からの距離に応じて、それぞれの建物の 3D 表面に色を付与している。またこの色を、画面上のスライダーにより動的に変更することができる。

この例を参考に、建築物モデルを用いて、都庁から個別の建物までの距離に応じて、建物の色を動的に変化させるデモアプリケーションを作成した。

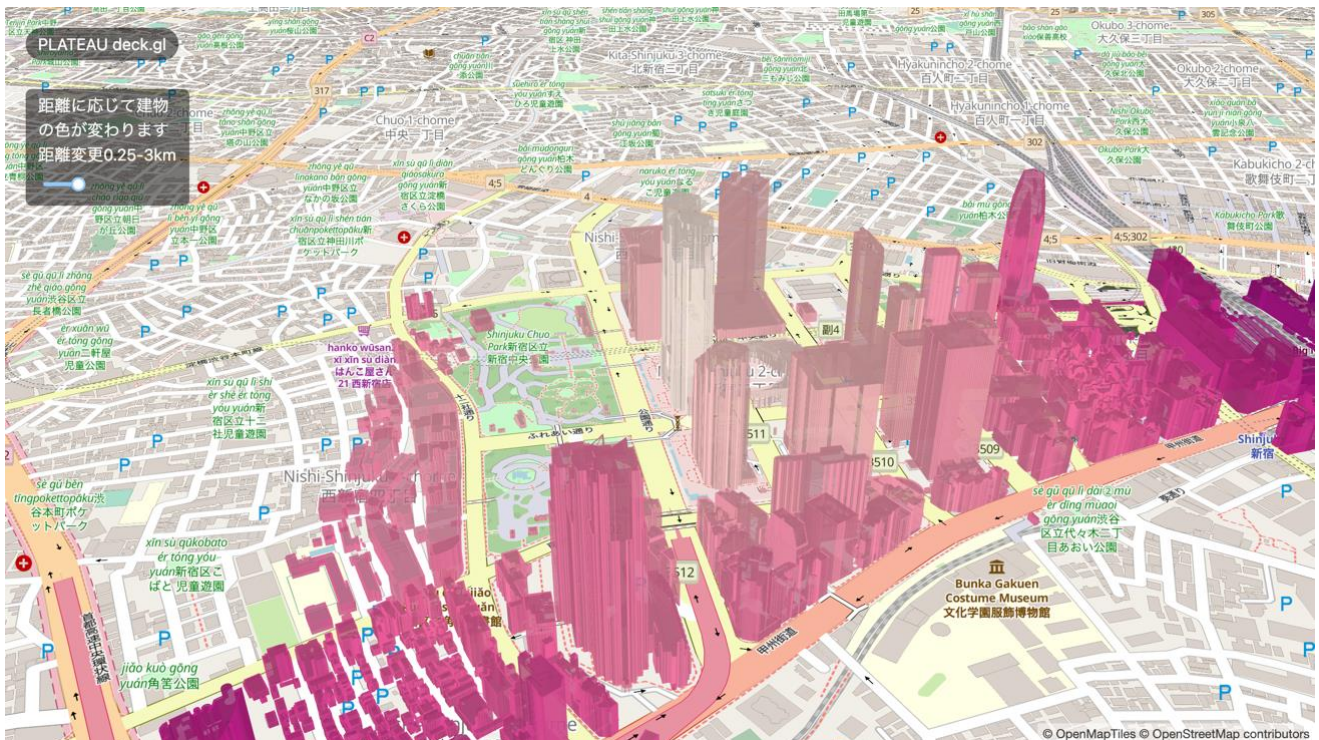


図 4-2 deck.gl - 条件に応じた建物色変更

デモアプリケーション STEP3 – deck.gl 距離に応じた建物色変化

<https://project-plateau.github.io/mapengine-survey/step3/deckgl-terrain/index.html>

4-4-1-b. 特徴と考察

この例ではまず、3D Tiles により都庁周辺の建築物モデルを描画している。

加えて、各建築物モデルの 2D フットプリントを GeoJSON のポリゴンデータとして用意し、各ポリゴンの色を、都庁からの距離に応じて変化させた。フットプリントは、CityGML から QGIS プラグインを用いて作成した。

そして TerrainExtension により、その色付きポリゴンを 3D 表面、つまり建築物モデル上へ沿わせて描画している。

この色は、画面上のスライダーを操作することで、建築物モデルの色を動的に変化させることができる。

以下に、TerrainExtension を用いる際のプログラム例を示す。

```
new Tile3DLayer({
  id: "3d-tiles",
  pointSize: 1,
  operation: "terrain",
```

```
data: "https://example.com/tileset.json",  
loader: Tiles3DLoader,  
})
```

```
new GeoJsonLayer({  
  id: "geojson",  
  data: data,  
  filled: true,  
  pointRadiusMinPixels: 2,  
  pointRadiusMaxPixels: 5,  
  getFillColor: (obj) => {  
    const centerPoint = turf.center(obj.geometry);  
    const distance = turf.distance(centerPoint, distancePoint);  
    return colorScale(distance);  
  },  
  getLineColor: (obj) => {  
    const centerPoint = turf.center(obj.geometry);  
    const distance = turf.distance(centerPoint, distancePoint);  
    return colorScale(distance);  
  },  
  lineWidthMinPixels: 5,  
  opacity: 0.5,  
  extensions: [new TerrainExtension()],  
  updateTriggers: {  
    getFillColor: [colorScale.domain()],  
  },  
})
```

元の建築物モデルへ対して色をつけたデータを静的に作成するのではなく、このように動的な形で 2D と 3D を合わせて描画する機能は、他の地図エンジンには見られない、deck.gl だからこそ実現できるものであり、インタラクティブに組み合わせた活用を可能とする。

4-4-1-c. 参考文献

表 4-1 deck.gl - 条件に応じた建物色変化 - 参考文献

タイトル	出典
deck.gl TerrainExtension (Experimental)ドキュメント	https://deck.gl/docs/api-reference/extensions/terrain-extension

4-4-2. deck.gl - 経路表示

4-4-2-a. 概要



図 4-3 deck.gl 公式 Examples - 飛行機の飛行軌跡表現の事例
deck.gl の LineLayer の表例 (<https://deck.gl/examples/line-layer/>)

deck.gl の LineLayer では、3 次元上での始点と終点を指定し、パスを描画することができる。また、パスの色など見た目を属性によって変化させることができる。deck.gl は大規模なデータを扱うことに長けており、大量のパスを効率的に描画することが可能である。

上記のサンプルは、この機能を利用して飛行機の飛行軌跡を表現している。個々のパスは高度に応じて青色から赤色へと変化し、また高度に応じて透過度に変更されている。

この LineLayer を使用して、3D 都市モデルと、新宿区内の道路情報を可視化するデモアプリケーションを作成した。

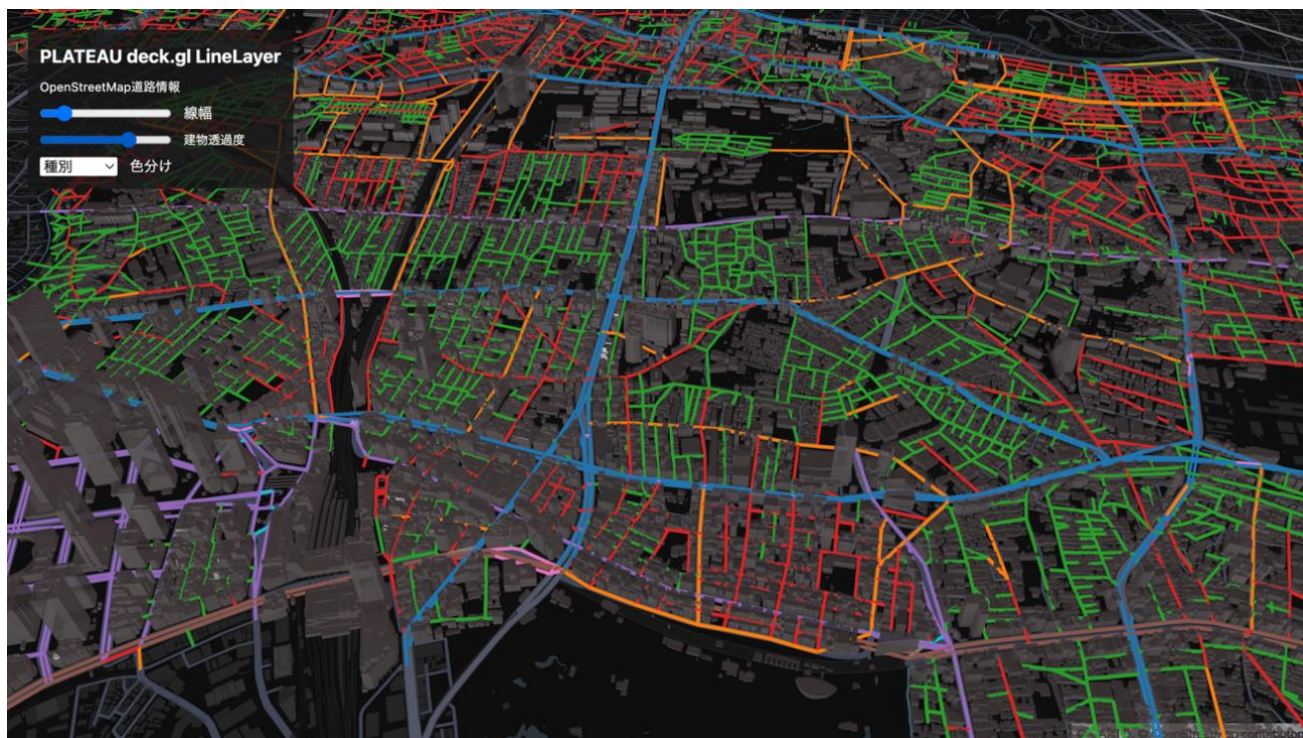


図 4-4 deck.gl - 新宿区道路経路表示

デモアプリケーション STEP3 – deck.gl 経路表示

<https://project-plateau.github.io/mapengine-survey/step3/deckgl-linelayer/index.html>

4-4-2-b. 特徴と考察

このサンプルでは、各道路を線として描画し、その幅を画面上のスライダーでインタラクティブに変更できるようになっている。加えて道路の属性情報によって、線の色が塗り分けられるようになっており、画面上のメニューから色分けを変更することが可能である。例えば、「長さ」「傾斜」では、その連続値に対応した色が適用され、「種別」「制限速度」「車線数」といった属性では、カテゴリカルにそれぞれの値に対応した個別の色が選択される。

プログラムでの利用に合わせて、始点と終点や属性のデータを独自に整理する必要がある。このサンプルでは、OpenStreetMap から道路情報を取得し整形した。以下に、今回用意したデータの例を示す。

```
{
  "from": [
    139.7074388,
    35.6979039,
    36
  ],
  "to": [
    139.7070166,
    35.697964,
    39
  ],
  "length": 38.715,
  "grade": 0.077,
  "oneway": true,
  "highway": "primary",
  "name": "職安通り",
  "maxspeed": "40",
  "lanes": "2"
}
```

始点と終点や属性のデータを汎用的な形に整理すれば、それをレイヤーに入力するだけで、追加のプログラミングなしにパスを描画することができ、またデータに合わせたスタイルの適用が可能である。またそのスタイルを動的に変更することも容易である。

これにより、道路や飛行経路など、始点と終点を持つ地理空間情報と 3D 都市モデルをシンプルに組み合わせ可視化することができる。

4-4-2-c. 参考文献

表 4-2 deck.gl - 経路表示 - 参考文献

タイトル	出典
deck.gl LineLayer ドキュメント	https://deck.gl/docs/api-reference/layers/line-layer

4-4-3. deck.gl - 移動体の走行軌跡

4-4-3-a. 概要

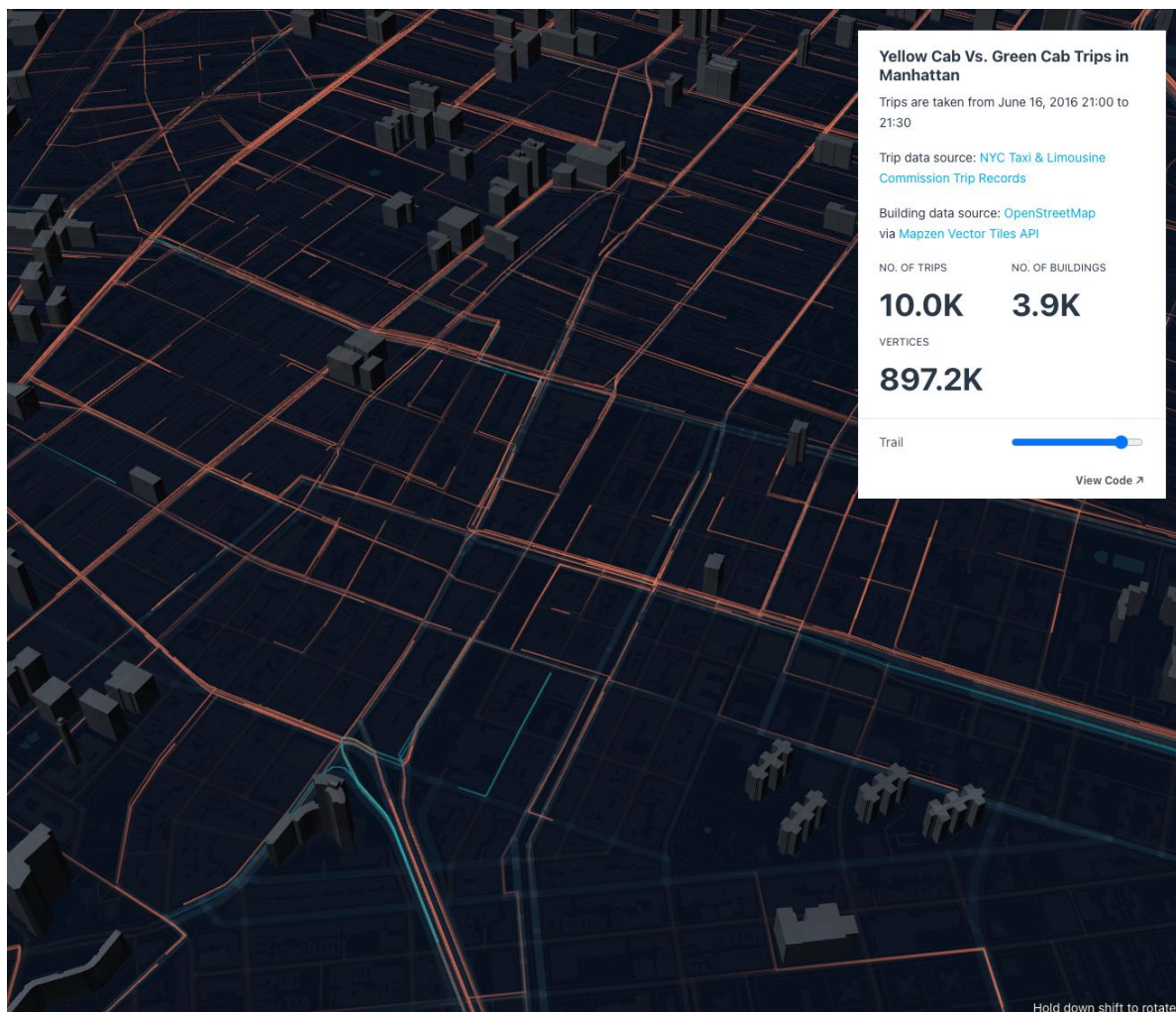


図 4-5 deck.gl 公式 Examples - タクシーの乗車走行軌跡表現の事例
deck.gl の TripsLayer の実装例 (<https://deck.gl/examples/trips-layer>)

deck.gl の TripsLayer では、3 次元上での始点と終点を指定し、その間での軌跡アニメーションを描画することができる。

上記のサンプルでは、この機能を用いて、ニューヨークでのタクシーの各経由地点の座標と時刻の情報を元に、乗車走行軌跡をアニメーション表示している。

この例を参考に、3D 都市モデルと組み合わせて、新宿区内の道路での走行軌跡を表示するデモアプリケーションを作成した。

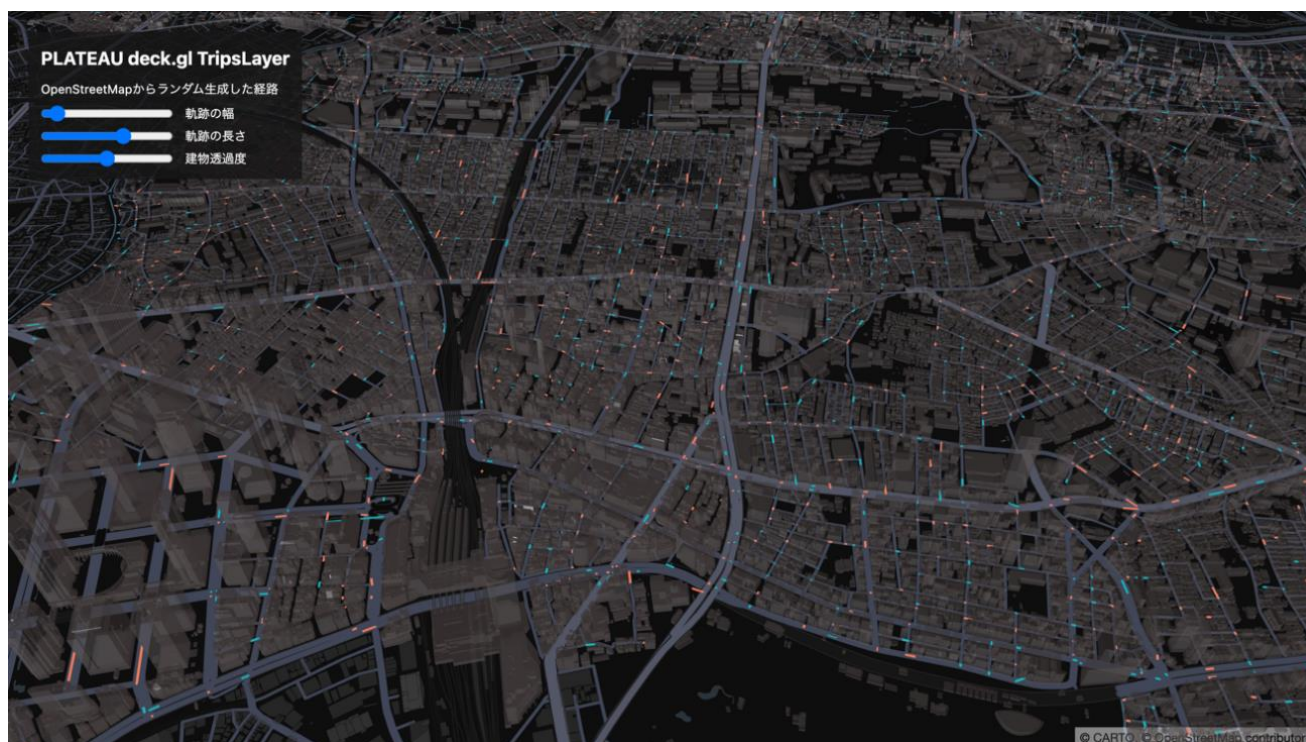


図 4-6 deck.gl - 新宿区道路走行軌跡表示

デモアプリケーション STEP3 – deck.gl 移動体の走行軌跡

<https://project-plateau.github.io/mapengine-survey/step3/deckgl-tripslayer/index.html>

4-4-3-b. 特徴と考察

今回の例では、移動軌跡データは OpenStreetMap の道路情報をもとにランダムに生成した。一般的な移動体のデータ（GPX や KML 等）をそのまま使用することはできず、汎用的な形に整形する必要がある。以下に、用意したデータの一例を示す。各経由点の座標情報（経緯度）と、各地点に対応する時刻の情報であることがわかる。

```
{
  "waypoints": [
    [139.62946813171357, 35.30919271889568],
    [139.6287223, 35.3104805],
    [139.62956417151852, 35.30915569822243]
  ],
  "timestamps": [
    0, 1, 2
  ]
}
```

移動軌跡は、各地点の時刻に合わせて、時系列でアニメーション表示される。走行後の軌跡がフェードアウトしていくのが特徴的であり、軌跡の長さ、線の色、再生スピード等は、パラメータで調整可能である。

4-4-3-c. 参考文献

表 4-3 deck.gl - 移動体の走行軌跡 - 参考文献

タイトル	出典
deck.gl TripsLayer ドキュメント	https://deck.gl/docs/api-reference/geo-layers/trips-layer

4-4-4. deck.gl - 拠点間の移動状況

4-4-4-a. 概要

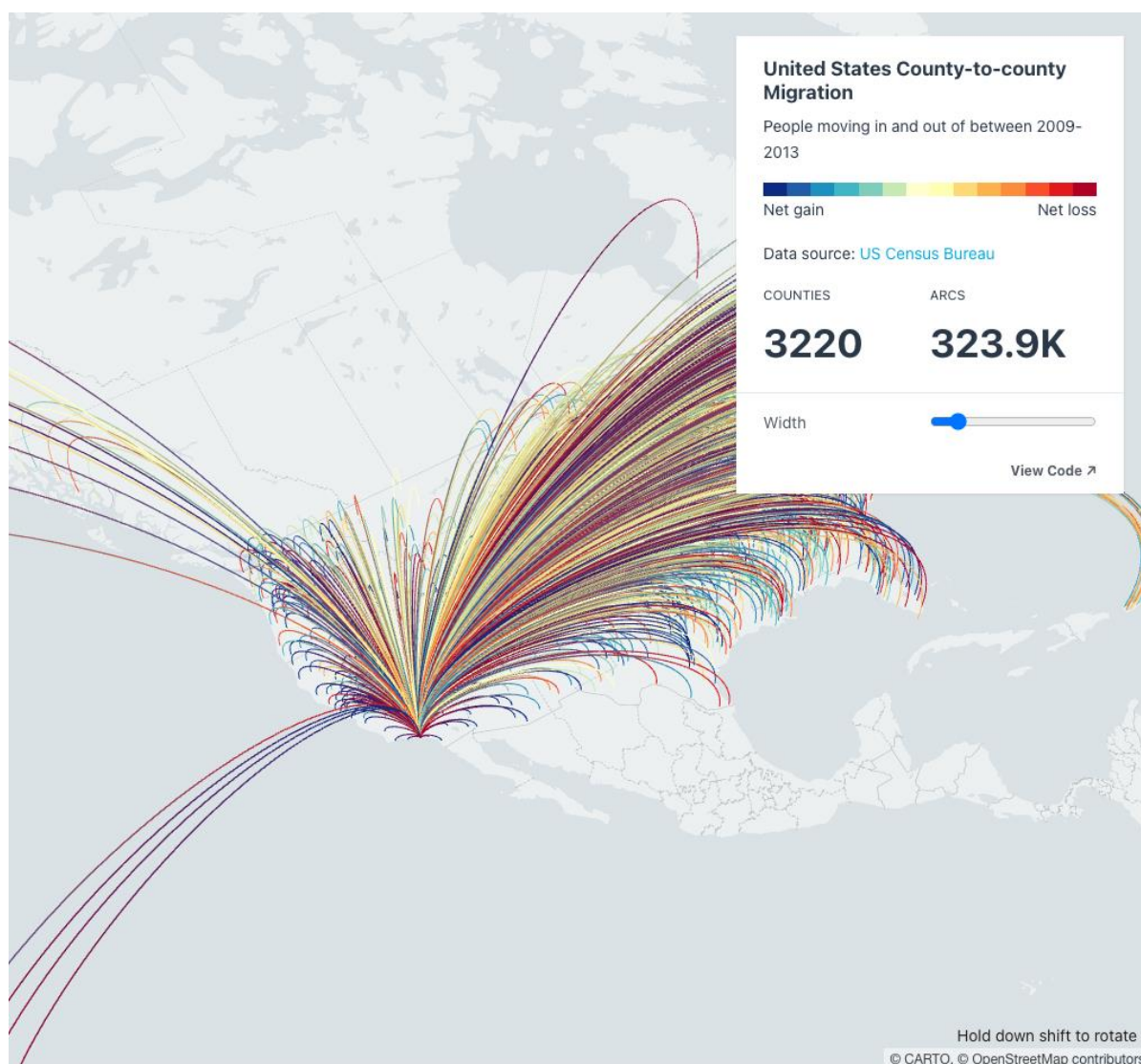


図 4-7 deck.gl 公式 Examples - 米国郡間移動の表現
deck.gl の ArcLayer の実装例 (<https://deck.gl/examples/arc-layer>)

deck.gl の ArcLayer では、始点と終点を指定し、弧（Arc）を描画することができる。
上記のサンプルでは、この機能を用いて、米国の群から群への移住者数を弧として表現している。弧の色は移住者数に基づいて決定されている。また、群をクリックすることで、始点となる群を変更することができる。

この例を参考に、3D 都市モデルと、行政区間移動人員を弧として表示するデモアプリケーションを作成した。

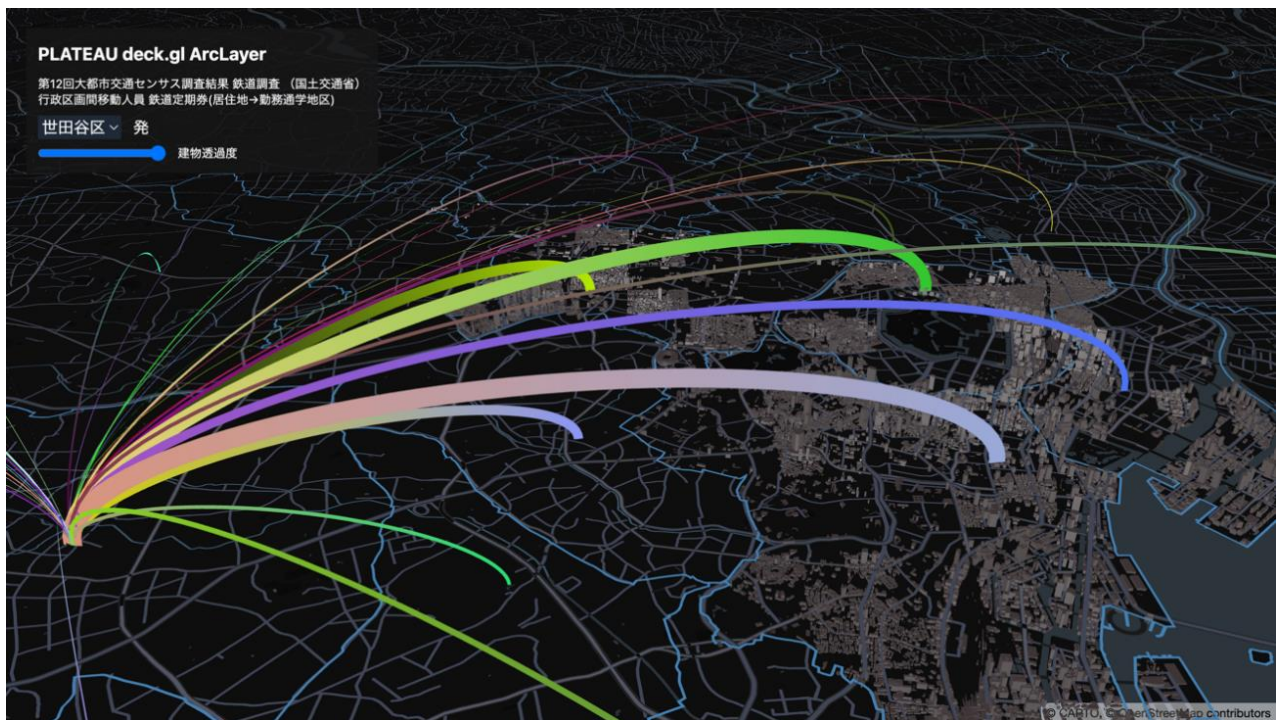


図 4-8 deck.gl - 東京 23 区、区間移動人員表現

デモアプリケーション STEP3 - deck.gl 拠点間の移動状況

<https://project-plateau.github.io/mapengine-survey/step3/deckgl-arclayer/index.html>

4-4-4-b. 特徴と考察

今回は国土交通省による第 12 回大都市交通センサス調査結果の「行政区画間移動人員 鉄道定期券(居住地→勤務通学地区)」

(https://www.mlit.go.jp/sogoseisaku/transport/sosei_transport_tk_000035.html) を使用し、東京 23 区内での区間移動人数を可視化した。始点・終点としては、区役所の位置を指定している。以下に、ArcLayer のために整形したデータの一部を示す。始点・終点の座標情報と、属性情報として人数が指定されていることがわかる。

```
[
  {
    "bound": 568,
    "from": {
      "name": "千代田区",
      "coordinates": [139.753634, 35.694003]
    },
    "to": {
      "name": "中央区",
      "coordinates": [139.772127, 35.670766]
    }
  },
  {
    "bound": 2264,
    "from": {
      "name": "千代田区",
      "coordinates": [139.753634, 35.694003]
    },
    "to": {
      "name": "港区",
      "coordinates": [139.751546, 35.658073]
    }
  }
]
```

描画に際しては、線の太さで、移動人数の大小を表現し、色はランダムに指定している。

また、その地域へマウスオーバーすることで出発区を変更し、インタラクティブに可視化を変えることができる。

この例で見られるように ArcLayer は、拠点間の数量を視覚的に表現することができる。N:N の表現もできるが、基本的には 1:N（ひとつの始点に対して複数の終点）での利用が現実的である。

また ArcLayer は、弧を描く表現であるがゆえ、建物と同じスケール感で描画すると、弧が建物に干渉してしまうことが多い。そのためこれは、ある程度広いスケールでの表示に向いている表現である。

4-4-4-c. 参考文献

表 4-4 deck.gl - 拠点間の移動状況 - 参考文献

タイトル	出典
deck.gl ArcLayer ドキュメント	https://deck.gl/docs/api-reference/layers/arc-layer

4-4-5. deck.gl - ヒートマップ

4-4-5-a. 概要

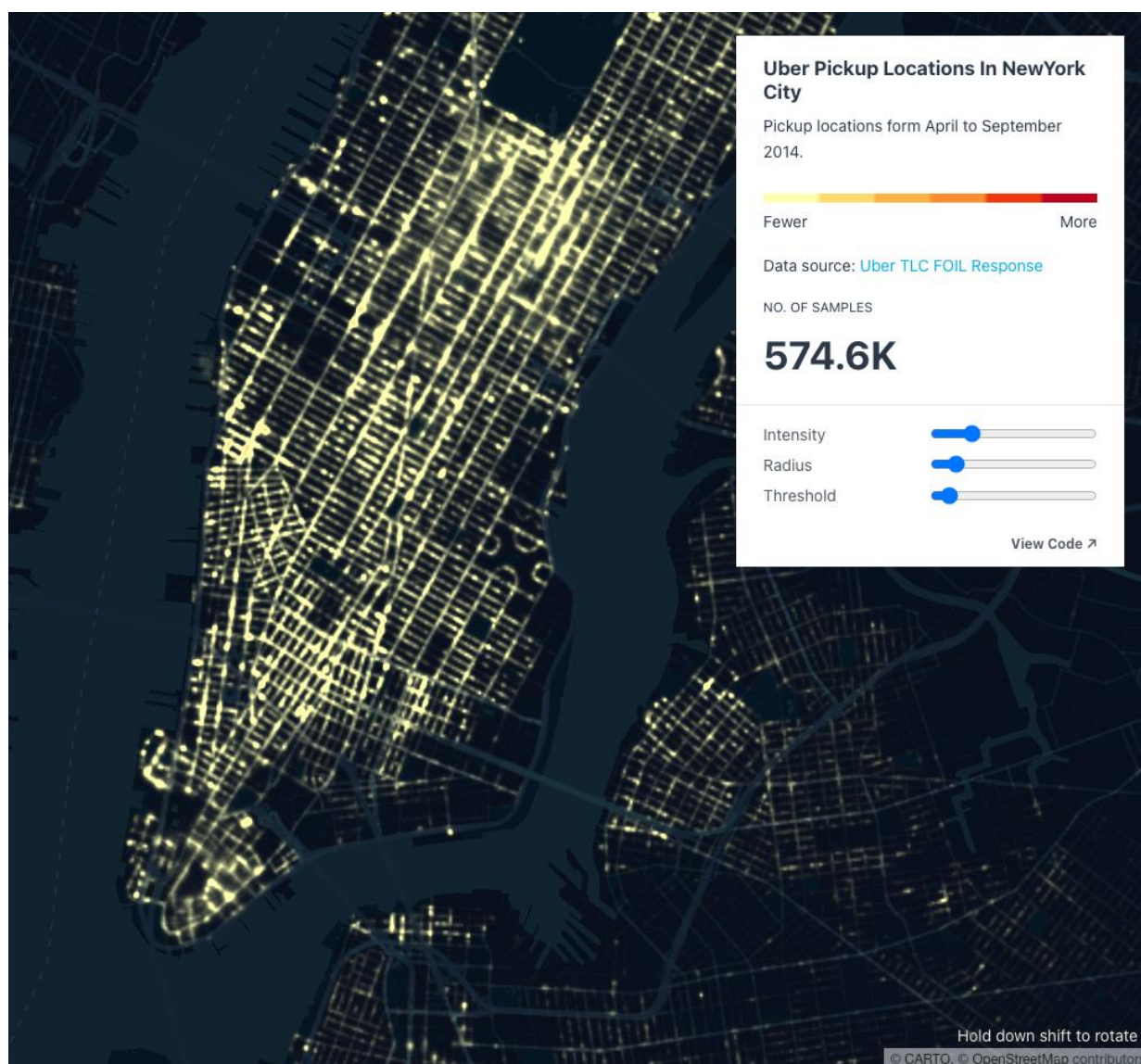


図 4-9 deck.gl 公式 Examples - Uber ピックアップ拠点のヒートマップ表現
deck.gl の HeatmapLayer の実装例 (<https://deck.gl/examples/heatmap-layer/>)

deck.gl の HeatmapLayer は、多数のポイント情報をもとに、その過多を色や濃淡として表現するヒートマップを実現するための機能である。

上記のサンプルでは、この機能を用いて、ニューヨークでのライドシェアのピックアップ場所をヒートマップで表現している。ズームレベルに応じて、ポイントの集中する箇所が強調される。

この例を参考に、3D 都市モデルと組み合わせて、東京消防庁の消火栓位置をヒートマップで表現するデモアプリケーションを作成した。



図 4-10 deck.gl - 東京消防庁、消火栓位置のヒートマップ表現

デモアプリケーション STEP3 - deck.gl ヒートマップ

<https://project-plateau.github.io/mapengine-survey/step3/deckgl-heatmaplayer/index.html>

4-4-5-b. 特徴と考察

今回利用した元データは、東京都オープンデータカタログサイトで公開されている「東京消防庁 消火栓及び防火水槽等」 (<https://catalog.data.metro.tokyo.lg.jp/dataset/t000017d00000000007>) で、約 13 万のポイントを含む。これをもとに、deck.gl の HeatmapLayer で用いるために汎用的な形へ整形した。以下に作成したデータの一部を示す。経緯度の情報に加え、重み（この場合は 1）を付与することができる。

```
[  
  [139.761033, 35.690628, 1],  
  [139.7610053, 35.6896143, 1],  
  [139.7590165, 35.6906252, 1]  
]
```

ヒートマップは、地点データの密度可視化に優れている表現である。

HeatmapLayer では、そのスタイルや、ズームレベルに応じた振る舞いなどを各種のパラメータで設定することができる。ズームレベルに応じてその表現を動的に変化させることが可能なため、広域な視点と詳細な視点の両方で観察することができる。

4-4-5-c. 参考文献

表 4-5 deck.gl - ヒートマップ - 参考文献

タイトル	出典
deck.gl HeatmapLayer ドキュメント	https://deck.gl/docs/api-reference/aggregation-layers/heatmap-layer

4-5. MapLibre GL JS の活用

4-5-1. MapLibre GL JS - ヒートマップ

4-5-1-a. 概要

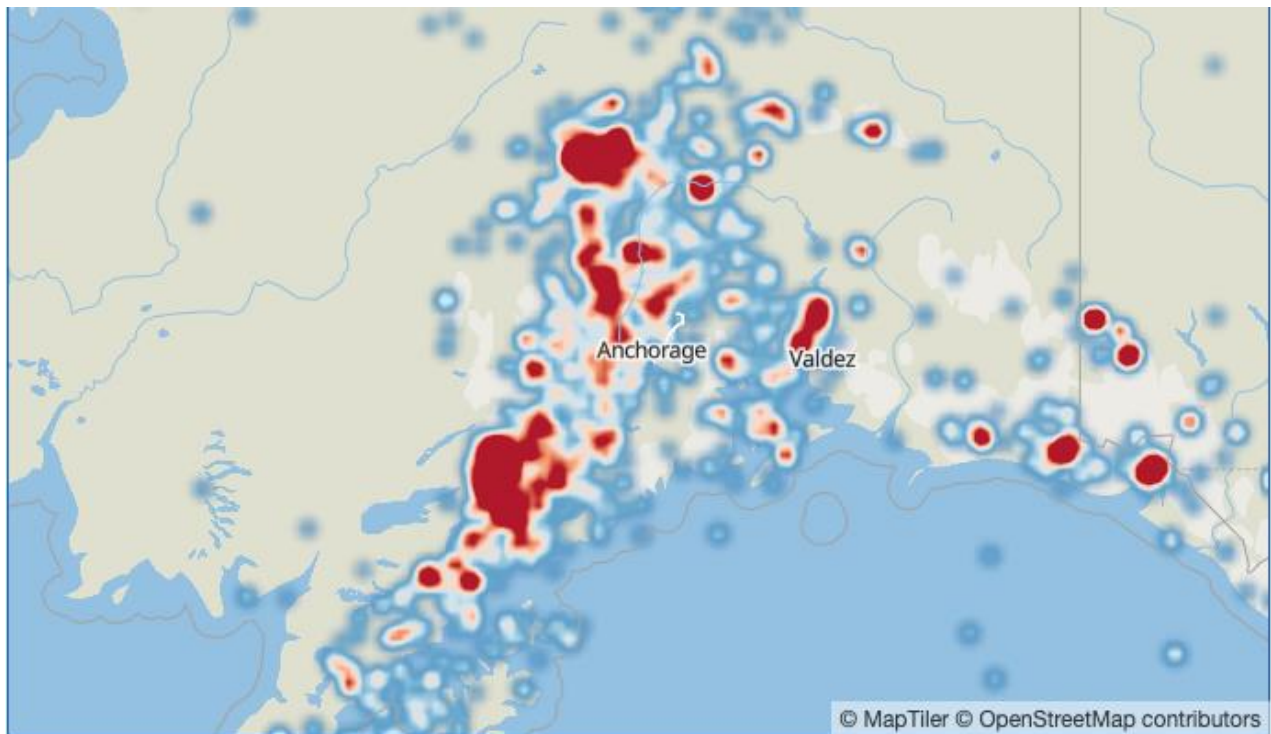


図 4-11 MapLibre 公式 Examples - 震源地ヒートマップ事例

MapLibre GL JS の heatmap layer の実装例 (<https://maplibre.org/maplibre-gl-js/docs/examples/heatmap-layer/>)

MapLibre GL JS にはヒートマップレイヤーの機能が標準で備わっており、多数のポイント情報をもとに、その過多を色や濃淡として表現するヒートマップを実現することができる。

上記のサンプルでは、この機能を用いて、場所ごとの地震の頻度をヒートマップで表現しており、ズームレベルに応じて震源の集中している箇所が強調表現される。

この例を参考に、3D 都市モデルと、東京消防庁の消火栓位置をヒートマップ表現するデモアプリケーションを作成した。

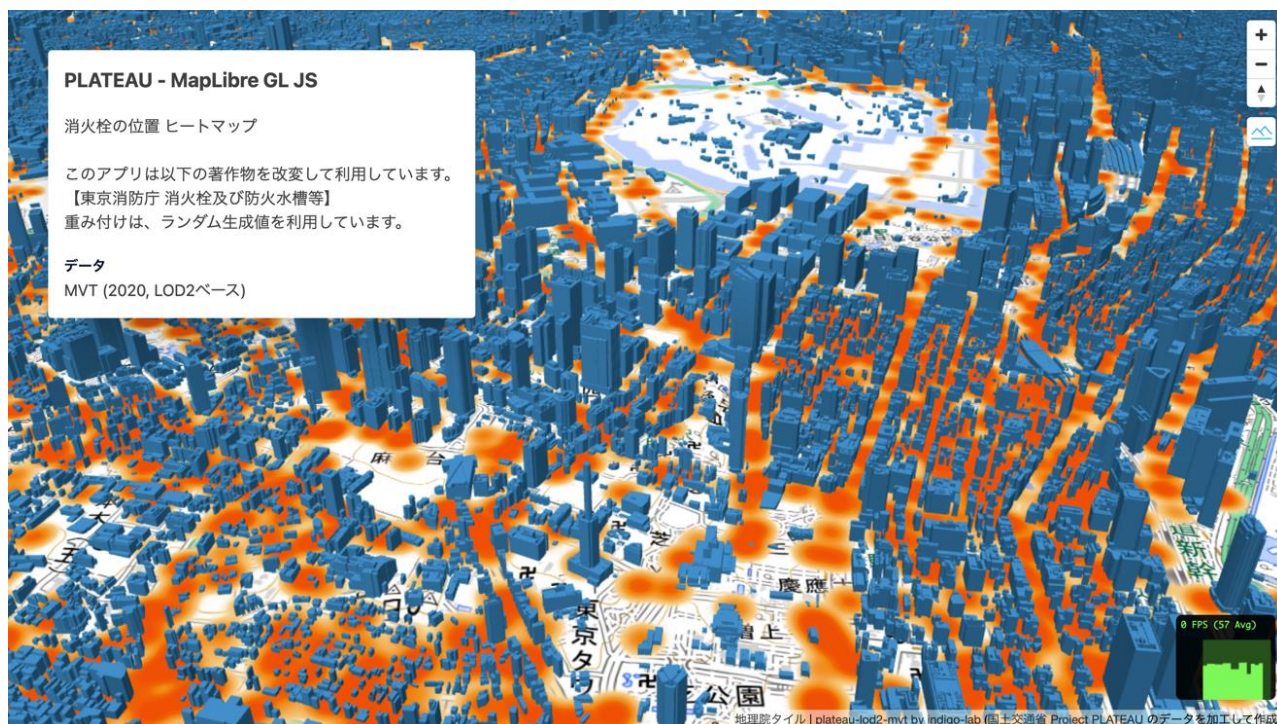


図 4-12 MapLibre GL JS - 東京消防庁、消火栓位置のヒートマップ表現

デモアプリケーション STEP3 - MapLibre GL JS - ヒートマップ

<https://project-plateau.github.io/mapengine-survey/step3/maplibre-heatmap/index.html>

4-5-1-b. 特徴と考察

deck.gl と異なり MapLibre GL JS では、一般的な地理空間情報のデータ形式である GeoJSON をそのまま用いることができる。

今回利用した元データは、東京都オープンデータカタログサイトで公開されている「東京消防庁 消火栓及び防火水槽等」（<https://catalog.data.metro.tokyo.lg.jp/dataset/t000017d00000000007>）で、約 13 万のポイントを含む。この元となる CSV 形式のファイルを、GeoJSON 形式のデータへ加工して利用した。

ヒートマップの密度や色、半径や透過度は、MapLibre GL JS の地図スタイルとして各種パラメータで設定することができる。

今回のデモでは、MVT による 3D 都市モデルの押出表現と組み合わせて、ヒートマップレイヤーは非常に軽量に動作することが確認できた。

このように、既存の GeoJSON ファイルをそのまま用いることができ、パフォーマンスに優れた機能は、地理空間情報の可視化を手軽に実現することに有益である。

4-5-2. MapLibre GL JS - クラスター

4-5-2-a. 概要

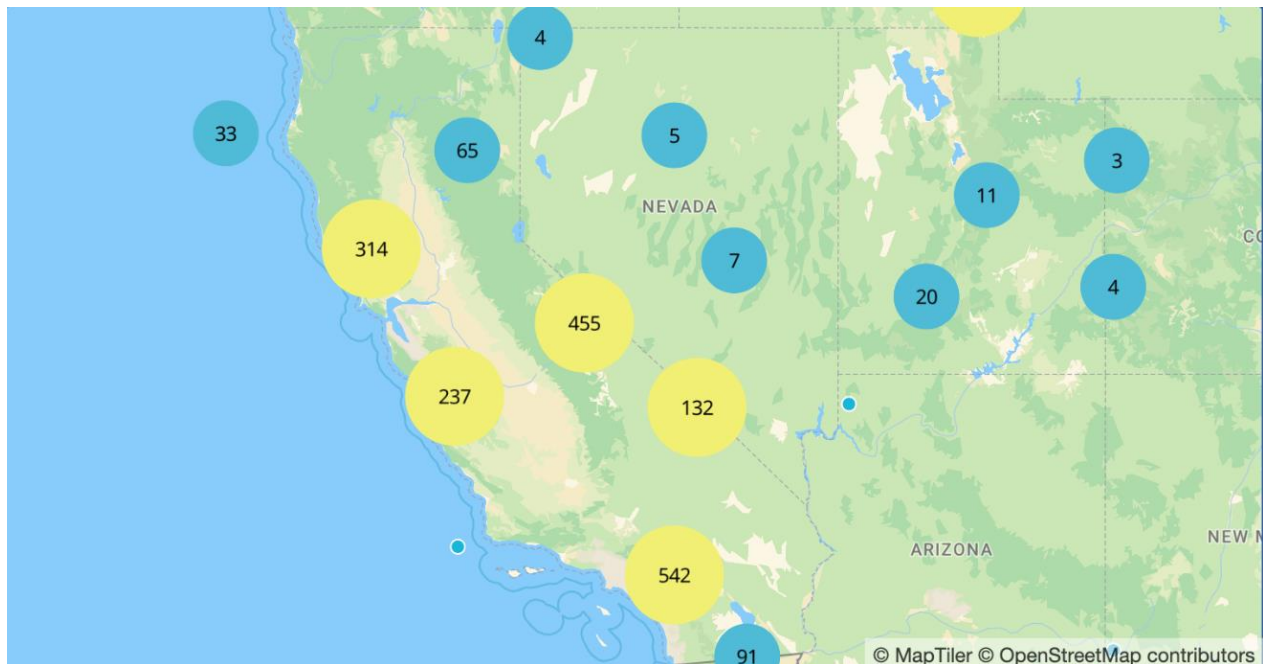


図 4-13 MapLibre GL JS 公式 Examples - 震源地クラスター表現の事例

MapLibre GL JS の Create and style clusters の実装例 (<https://maplibre.org/maplibre-gl-js/docs/examples/cluster/>)

MapLibre GL JS では、組み込み関数を利用し、多数のポイントをまとめ上げたクラスターとして表示することができる。

上記のサンプルは、この機能を用いて、地震の発生した地点をクラスターとして表している。ズームレベルに応じて、ポイントの集中している箇所がまとめられ、円の大きさや色、数字が動的に変化する。この例を参考に、3D 都市モデルと、東京消防庁の消火栓位置をクラスター表現するデモアプリケーションを作成した。

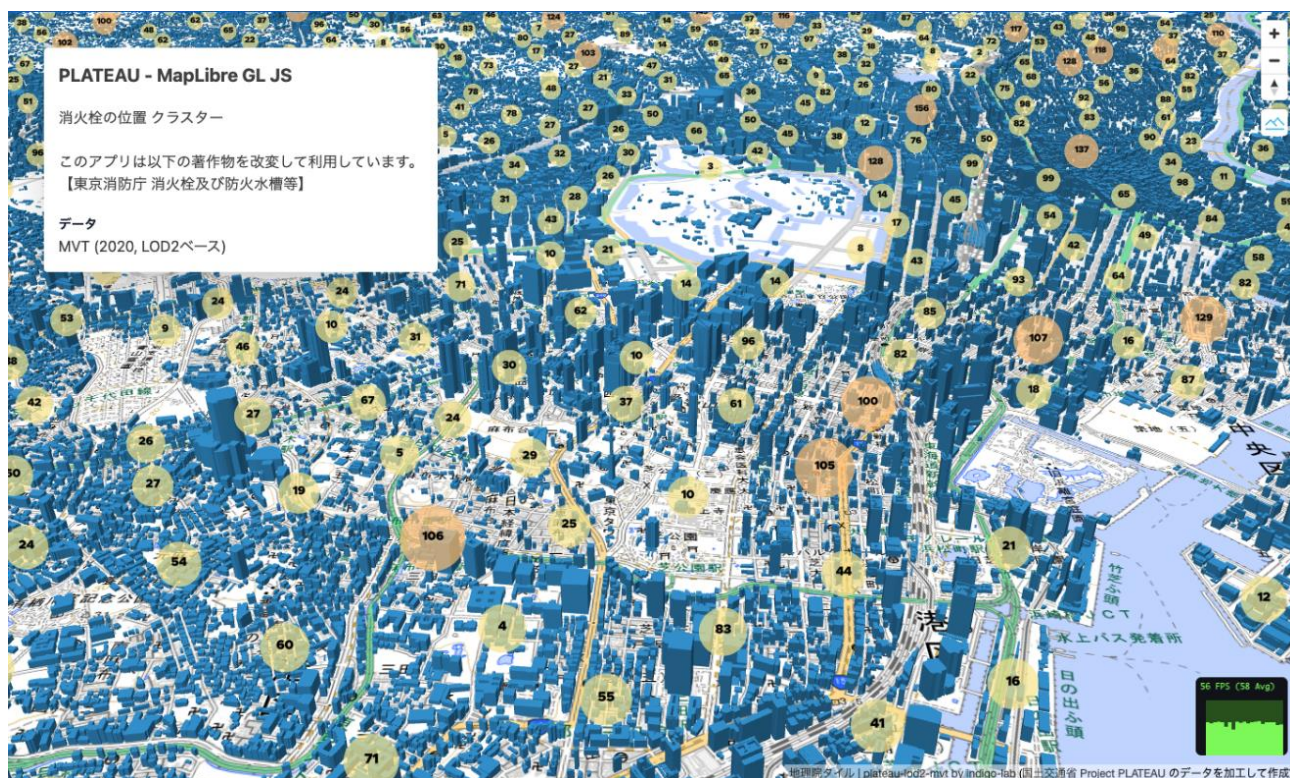


図 4-14 MapLibre GL JS 東京消防庁 - 消火栓位置のクラスター表現

デモアプリケーション STEP3 - MapLibre GL JS - クラスター

<https://project-plateau.github.io/mapengine-survey/step3/maplibre-cluster/index.html>

4-5-2-b. 特徴と考察

deck.gl と異なり MapLibre GL JS では、一般的な地理空間情報のデータ形式である GeoJSON をそのまま用いることができる。

今回利用した元データは、東京都オープンデータカタログサイトで公開されている「東京消防庁 消火栓及び防火水槽等」（<https://catalog.data.metro.tokyo.lg.jp/dataset/t000017d00000000007>）で、約 13 万のポイントを含む。この元となる CSV 形式のファイルを、GeoJSON 形式のデータへ加工して利用した。

MapLibre GL JS でのクラスター表示は、ズームレベルに応じて、円の大きさや色、数字を動的に変化させることができる。この見た目や振る舞いは、地図スタイルの各種パラメータで設定可能である。今回のデモアプリケーションでは、MVT による 3D 都市モデルの押出表現と組み合わせて、クラスター表示は非常に軽量に動作することが確認できた。

広域な視点では、クラスターに集約することで人間にとって認知が容易な形で提示可能となる。一方で、詳細な視点では、クラスターを分解して個別のポイントをそのまま提示することができる。これにより、どちらの視点でも適切な粒度で、有益な形で情報の提示をすることが可能になる。

5. 成果と課題、今後の展望

5-1. 成果と課題

本調査では、Project PLATEAU においてスタンダードになりつつある CesiumJS 以外の地図エンジンの可能性について調査を行った。

まず、STEP1 では「地図エンジンの調査」を行い、3D 都市モデルを描画することが可能な地図エンジンの特徴や、扱うことが可能なデータ形式などを網羅的に整理した。

次に STEP2 では「3D 都市モデルの表示性能検証」として、STEP1 で把握した地図エンジンを利用して実際に 3D 都市モデルの可視化環境を構築することで、実運用上の 3D 都市モデルの表示パフォーマンスを確認した。これらの調査・検証から、deck.gl および MapLibre GL JS が、実用的に使用可能な地図エンジンとして有望であることが判明した。これらの地図エンジンは、動的なスタイル変化や、美しいグラフィックス表現、ヒートマップやクラスターなど高度な表現手法に対応している。

最後に STEP3 では「表現方法の調査」として、deck.gl と MapLibre GL JS を用いて、CesiumJS では実現が難しい高度な表現方法を、実際にデモアプリケーションを作成し確認した。deck.gl は、3D Tiles をはじめとする様々なデータ形式に対応しており、また経路表示や移動体の軌跡可視化といった他のライブラリにはない高度な可視化機能を備えている。3D 都市モデルと様々な地理空間情報と組みあわせて一定以上のクオリティのビジュアルで可視化表現を行いたい際には有力な選択肢となる。一方で、3D Tiles の描画パフォーマンスや属性情報の扱いは CesiumJS に比べて劣るため、シンプルに 3D Tiles を描画したい場合や属性情報を扱いたい場合には CesiumJS が有力な選択肢となる。

MapLibre GL JS では、3D Tiles など用いた精緻な 3D 都市モデルの表現は行えないが、MVT の押出による簡易的な表現は可能であり、これは非常に軽量に動作する。またあわせて、ヒートマップやクラスターといった機能が標準で用意されており、一般的な GeoJSON 形式などに対応し、既存のデータをそのまま利用することが可能である。このことから、簡易的な 3D 都市モデルの表現で十分である場合には適した選択肢であると言える。

また、deck.gl や MapLibre GL JS での例で述べたように、両者のレイヤーを重ね合わせることで、複数のデータや異なる可視化表現を組み合わせ、多角的な観点からデータを観察することが可能である。

これらのことから、精緻な 3D 都市モデル表現を求める場合や、個別の建物の属性に着目するケースでは CesiumJS、データ全体を俯瞰した分析や、他のデータとの比較、高度な可視化を行いたい場合には deck.gl および MapLibre GL JS というように、利用シーンごとに使い分けることによって、3D 都市モデルの価値を十分に引き出すことができると考えられる。

5-2. 今後の展望

2023 年度に開発された PLATEAU VIEW 3.0 では、「建築物など 3D モデルの細部の陰影」「地形の影の表現」「ヒートマップ表現」など、CesiumJS では難しかった、従来よりも高いレンダリング品質が実現された。しかし、この実装にあたっては、CesiumJS の非公開 API を使用したり、シェーダの実装を上書きしたりするなど、一部、本来の CesiumJS の使用方法から逸脱する手法が用いられた。理想的には、このような方法を用いなくとも、地図エンジンが最初から、このような高いレンダリング品質を実現できることが望ましい。

他方、今回の調査で見たように、deck.gl や MapLibre GL JS といった CesiumJS 以外の地図エンジンでは、一部、CesiumJS では難しい高度な可視化表現が可能であることが確認された。しかし、各地図エンジンの調査を行った際に判明したように、3D Tiles による緻密な 3D 都市モデル表現は CesiumJS と比べてパフォーマンスや属性の取り扱いなどで劣り、PLATEAU VIEW の地図エンジンとして置き換えるには不十分である。

これらを踏まえ、将来的に CesiumJS 以外の地図エンジンでも、3D Tiles の扱いについて CesiumJS と同等レベルに表示可能になるように、地図エンジンの機能改善など開発の発展が望まれる。

加えて長期的には、これまで調査して判明した既存地図エンジンの課題を解決するような、新しい地図エンジンを新規に開発することも考えられる。

WebGIS 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：株式会社ユーカリヤ