Silicon Studio
Ideas × Art × Technology

CEDEC2023

Procedural PLATEAU: プロシージャル技術と3D都市モデル「Project PLATEAU」を組み合わせたファサード自動生成とテクスチャ付き3Dモデル自動生成の取り組み

主分野: ENG 関連分野: PRD VA

シリコンスタジオ株式会社
研究開発室
大道 博文

2023/0825

はじめに

- 大道 博文
  - シリコンスタジオ株式会社
    - 2021年新卒入社
    - 研究開発室所属
    - PLATEAUを活用した研究プロジェクトの開発・検証に従事

講演資料公開予定

はじめに

- 本セッションで重視にしていること
  - PLATEAU（3D都市モデル）のデータ活用方法
    - 基本的にデータの特徴を重視する
  - データの特徴
    - オープンデータ
    - 品質（位置情報）
    - 詳細度のある建物モデル
    - 属性情報

目次

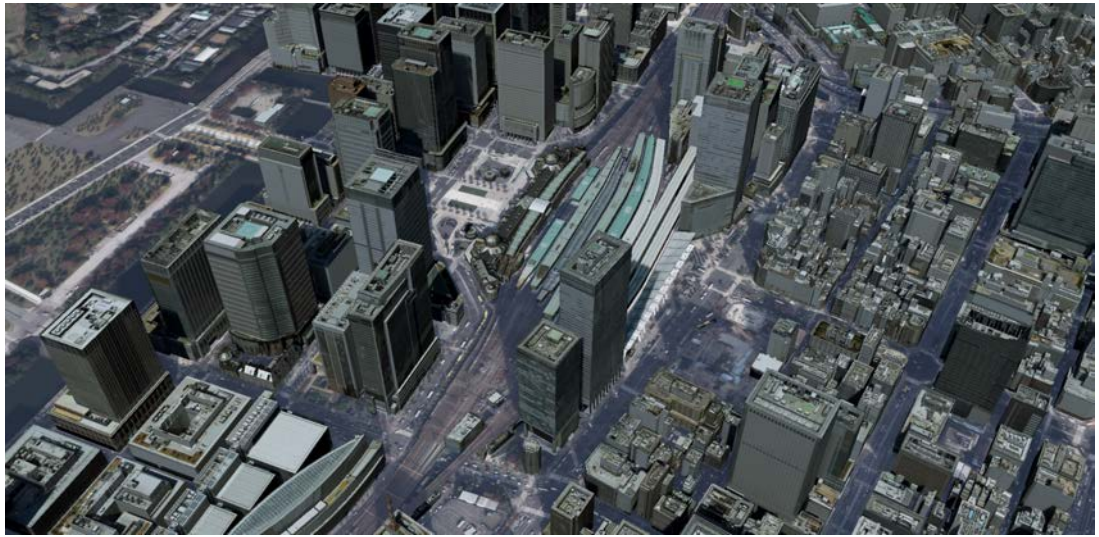
1. Project PLATEAUの紹介
2. 現状の課題と課題に対するアプローチ
3. プロシージャルモデリングとは？
4. Procedural PLATEAUの説明
5. Blenderアドオンでの手続き自動化
6. ゲームエンジン上でのパフォーマンス考慮点
7. まとめ

目次

Project PLATEAUとは？

Silicon Studio
Ideas × Art × Technology

- 日本全国の3D都市モデルオープンデータ化プロジェクト
  - 国土交通省が主導
  - まちづくりやドローン、防災などのソリューション開発が進む



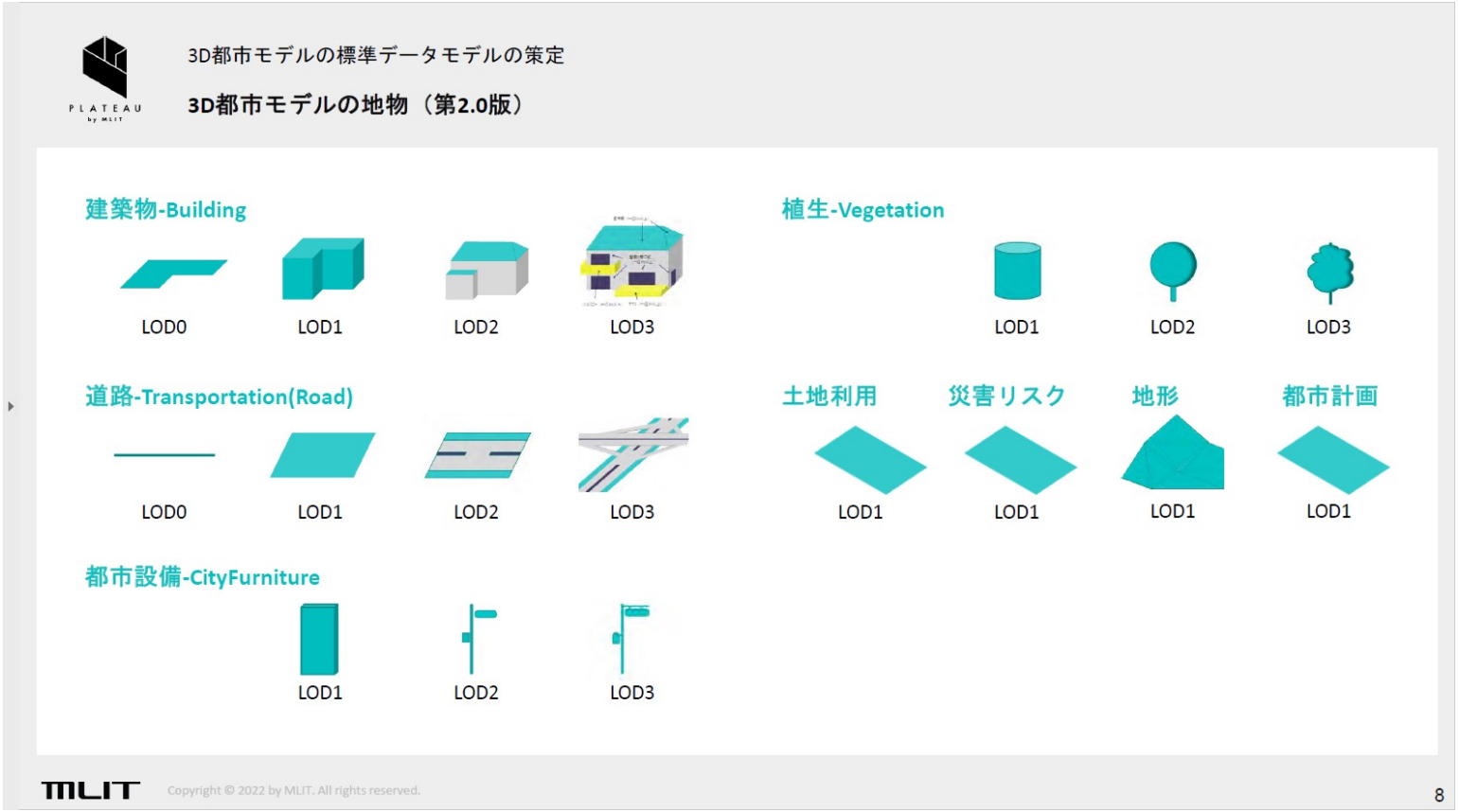
https://plateauview.mlit.go.jp/

PLATEAUの全国整備範囲

- オープンデータの数は約130都市（2022年度現在）
  - データセット：https://www.geospatial.jp/ckan/dataset/plateau

https://www.mlit.go.jp/plateau/open-data/より一部改変

PLATEAUのデータ



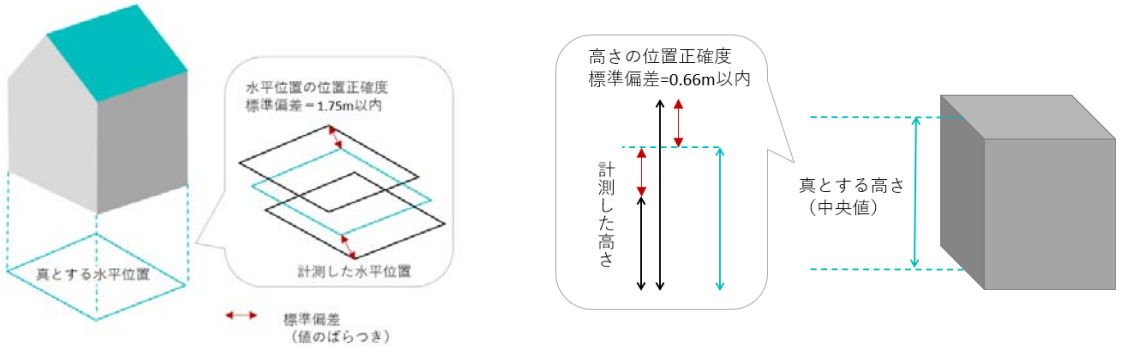
https://www.soumu.go.jp/main_content/000839747.pdf

PLATEAUのデータ

https://www.soumu.go.jp/main_content/000839747.pdf

PLATEAUのデータ　品質（位置情報）

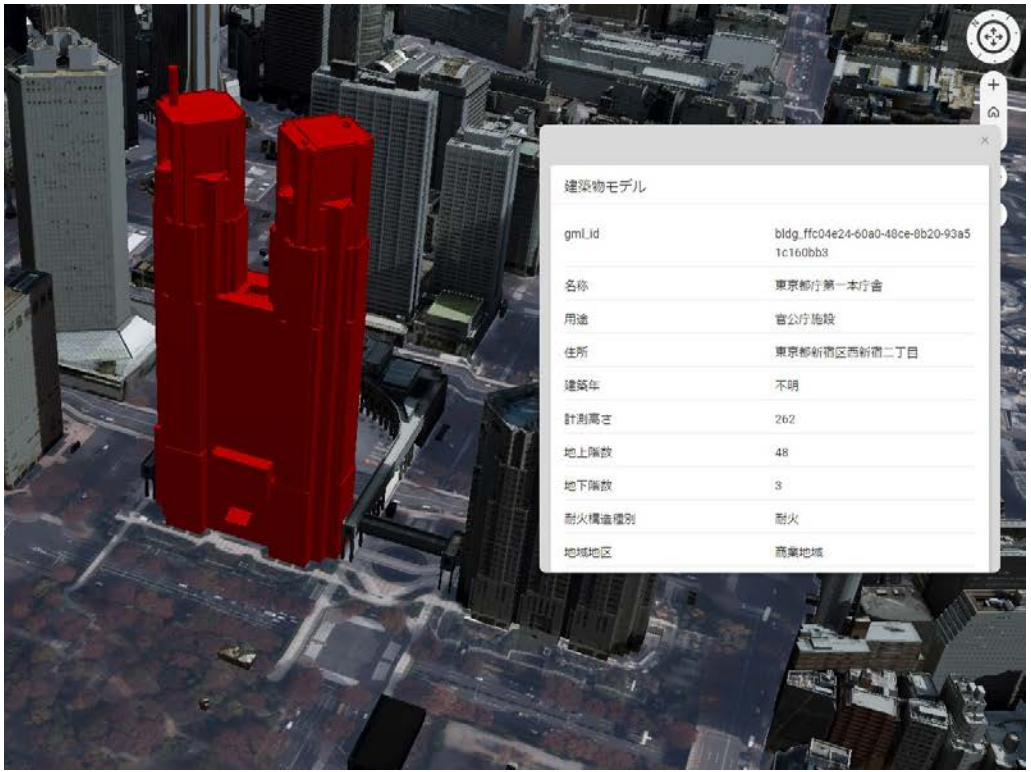
- 地図情報レベル2500（縮尺1/2,500）の位置正確度を基本として採用
  - 地図情報レベル：位置がどれほど正確なのかを示す指標
  - 位置正確度
    - 水平位置：1.75m以内（標準偏差）
    - 高さ：0.66m以内（標準偏差）



https://www.mlit.go.jp/plateau/learning/tpc03-3/#column_03_01

PLATEAUのデータ　属性情報

- 建物データ
  - 属性情報が付与
    - 用途
    - 住所
    - 建築年
    - 計測高さ
    - 地域地区 etc...

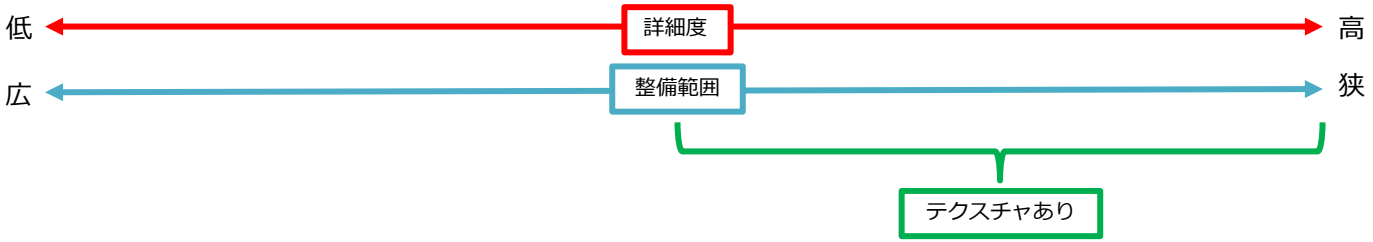


https://plateauview.mlit.go.jp/

PLATEAUのデータ　3D都市モデルにおけるLOD

- Level of Detail(LOD)
  - 詳細度のこと
  - LOD0~4まで明確に定義
    - 数値が大きいほどより詳細なモデル
      - ただし、整備範囲は狭くなる
  - LOD2以降はテクスチャあり

LOD0
平面投影形状（高さ情報がない表現）



https://www.mlit.go.jp/plateau/learning/tpc01-2/より一部改変

PLATEAUのデータ　3D都市モデルにおけるLOD

- 複数のLODを持つデータ構造
  - e.g. LOD2に対応した建物データ
    - LOD0、LOD1、LOD2を持つ

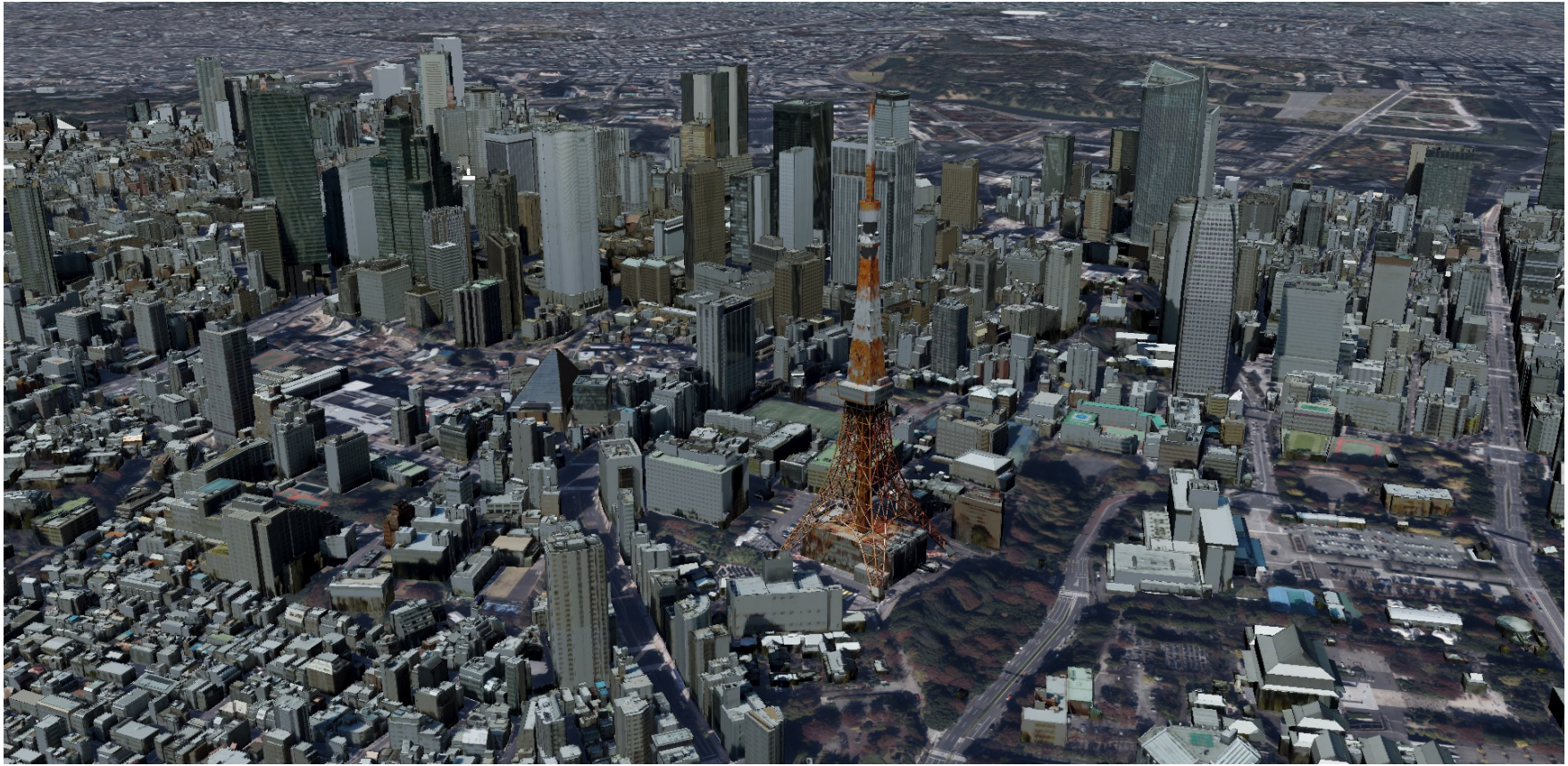
LOD0
平面投影形状（高さ情報がない表現）

LOD2に対応した建物

https://www.mlit.go.jp/plateau/learning/tpc01-2/より一部改変

実際の建物データでの比較（LOD2）

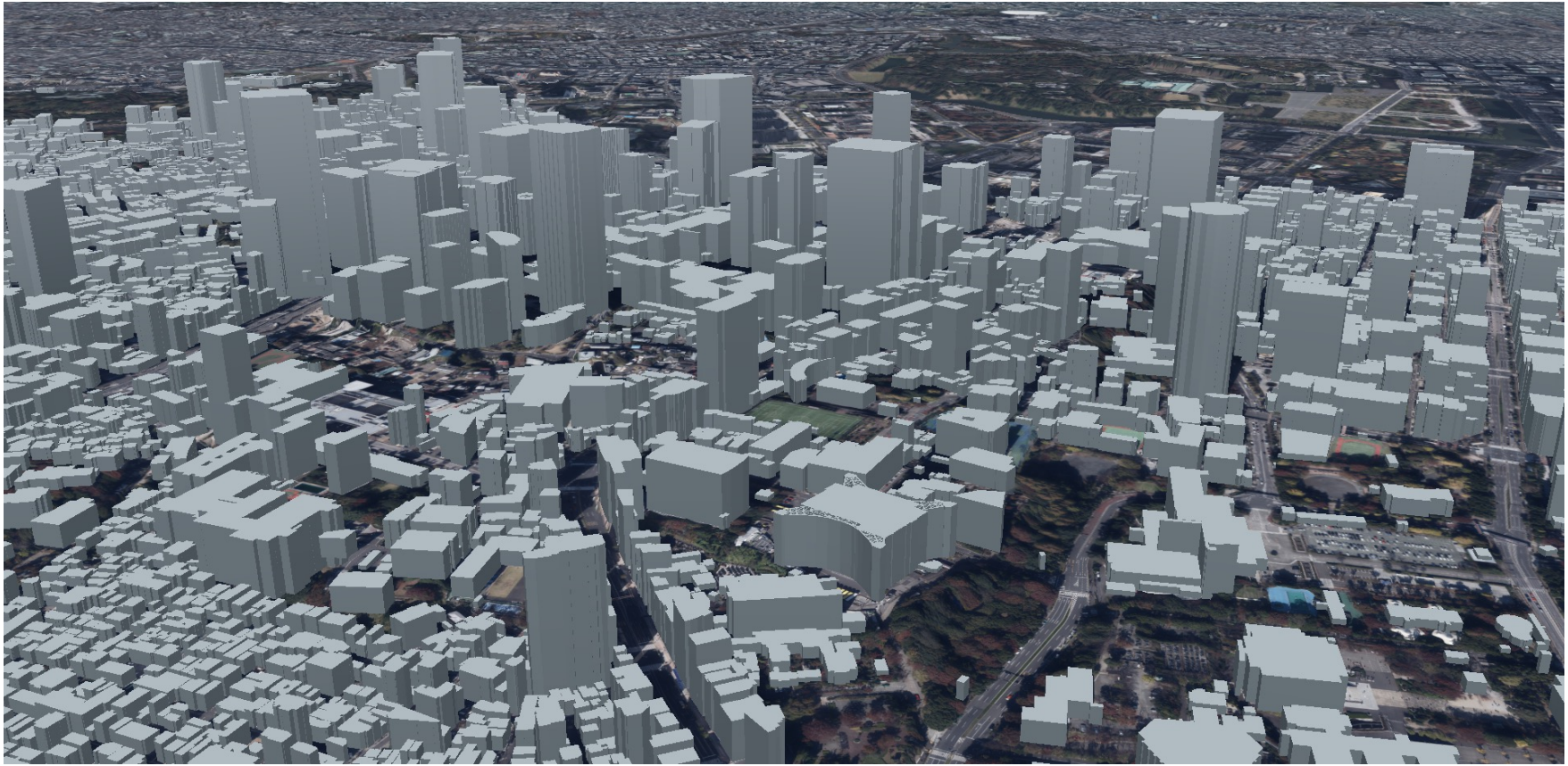
Silicon Studio
Ideas × Art × Technology



https://plateauview.mlit.go.jp/

実際の建物データでの比較（LOD1）

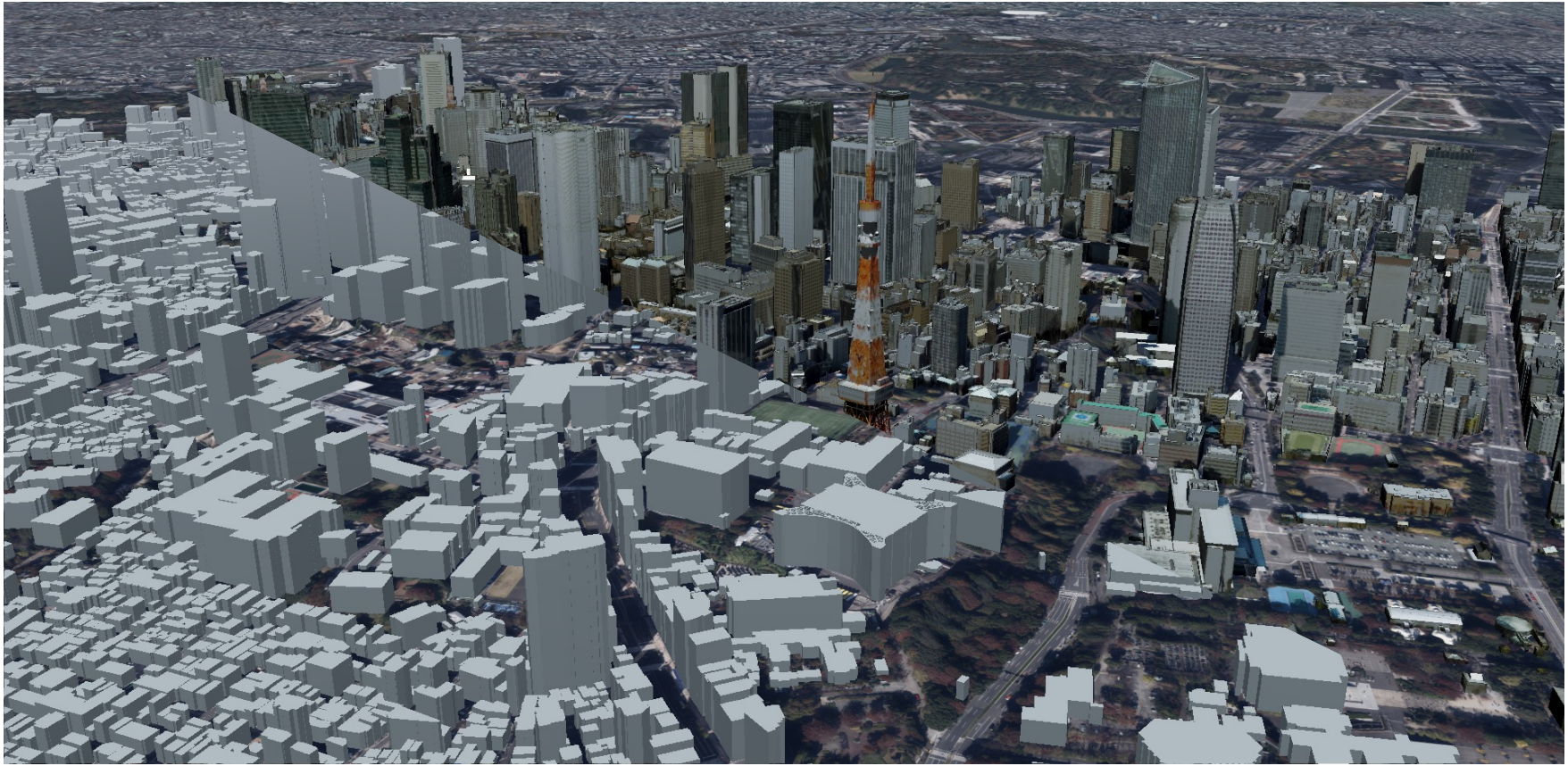
Silicon Studio
Ideas × Art × Technology



https://plateauview.mlit.go.jp/

実際の建物データでの比較（両者）

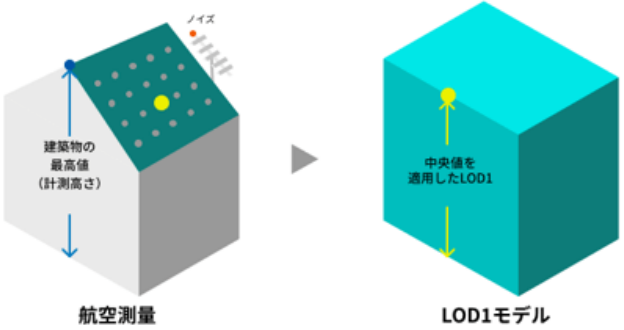
Silicon Studio
Ideas × Art × Technology

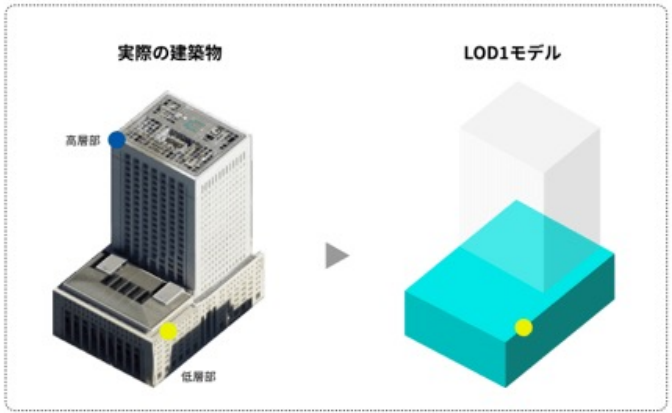


https://plateauview.mlit.go.jp/

LOD1モデルの建物の高さ

- LOD1モデルのコンセプト
  - 低コスト・簡易的な3D都市モデルの構築
  - 平面投影形状＋測量によって取得した高さ（中央値）
    - 中央値：測量した点群の高さを昇順に並べたときの中央値
  - 軽量な3Dモデル表現が強み





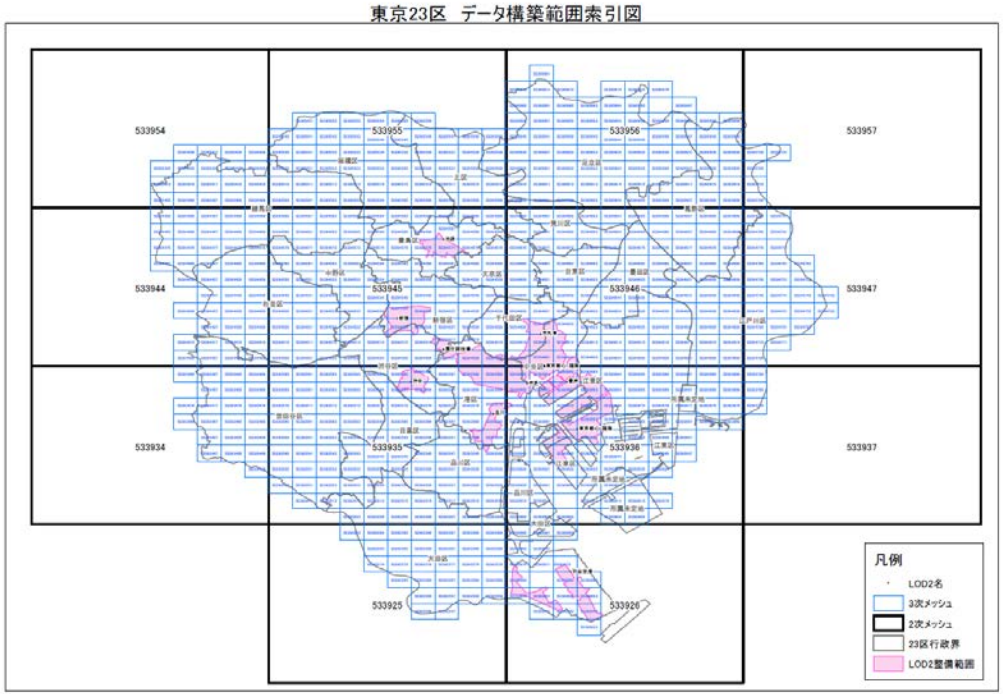
https://www.mlit.go.jp/plateau/learning/tpc03-4/#column_lod1

目次

建物データの課題

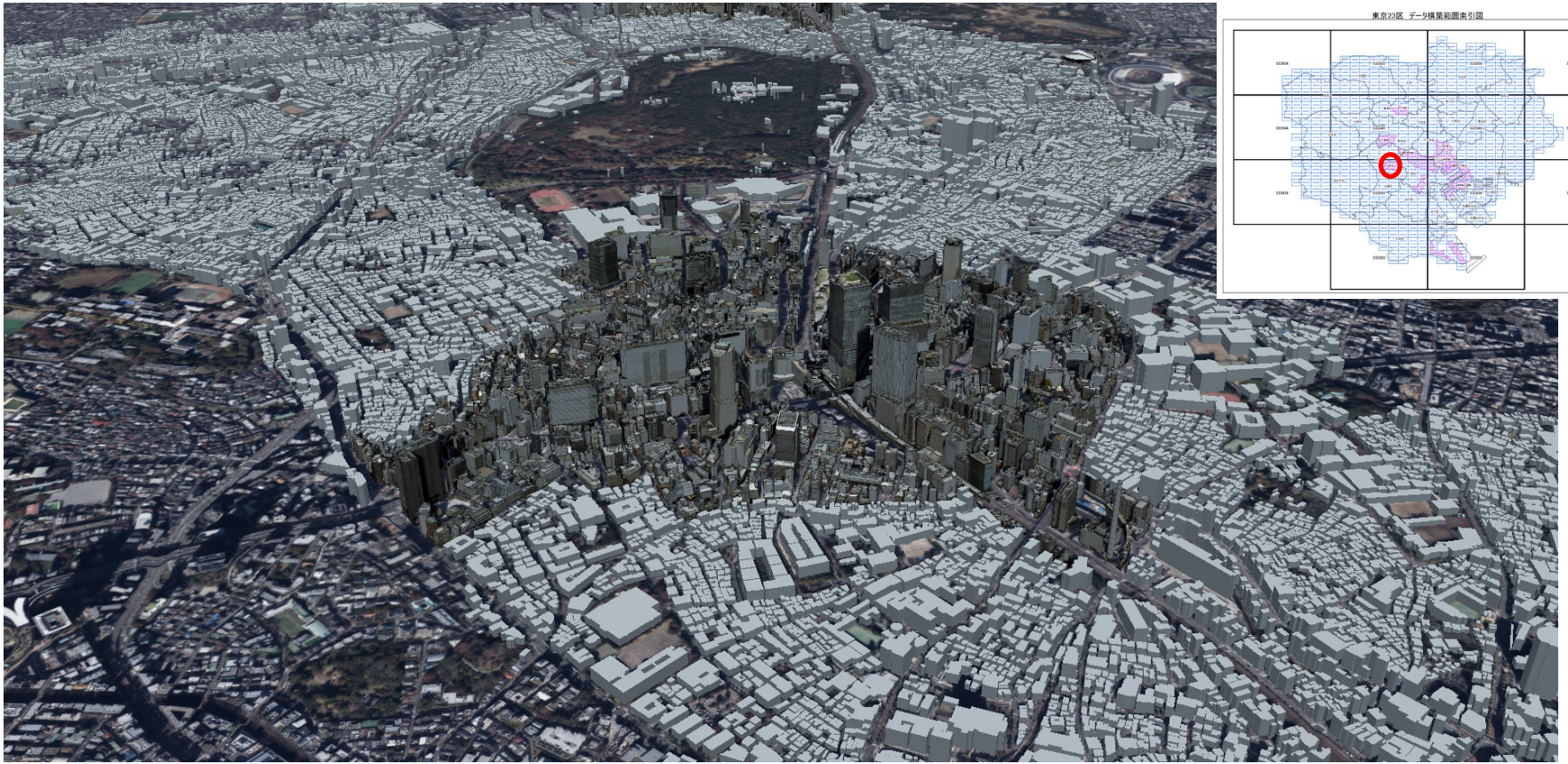
- LOD2の整備範囲
  - 東京23区(2020年度)
    - LOD1（全範囲）：627.57km^2
    - LOD2（ピンク）：32.02km^2
  - 他の自治体も同様の傾向
    - より狭い場合も

<u>テクスチャのない建物が大多数</u>



https://www.geospatial.jp/ckan/dataset/plateau-tokyo23ku

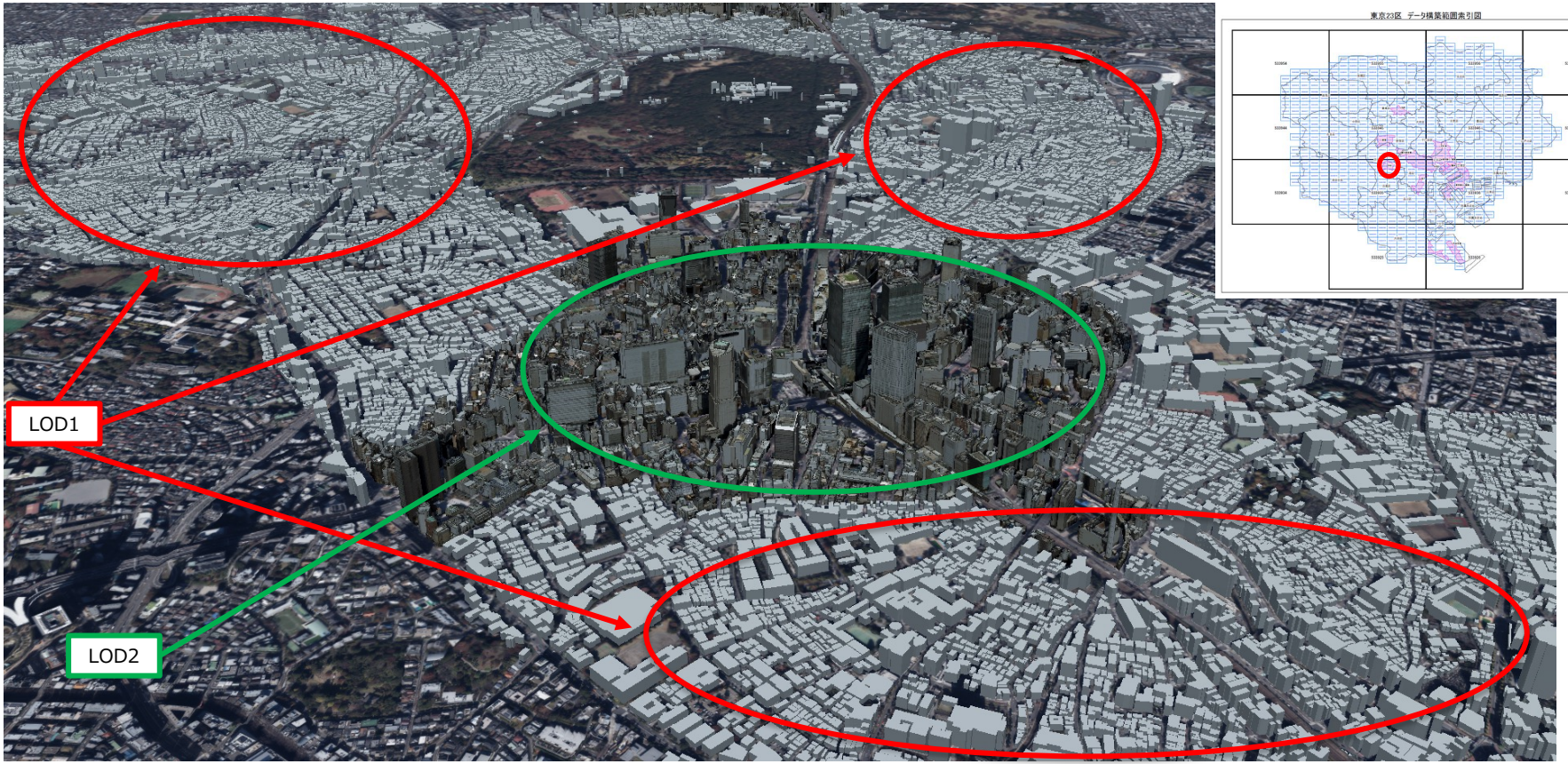
実際の建物データの整備状況（渋谷区）



https://plateauview.mlit.go.jp/

実際の建物データの整備状況（渋谷区）

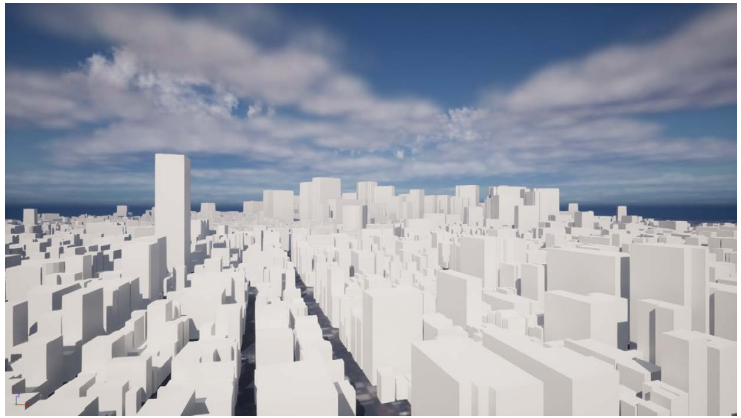
Silicon Studio
Ideas × Art × Technology



https://plateauview.mlit.go.jp/

課題に対するアプローチ

- 目的
  - PLATEAUのデータを活用したテクスチャ付きLOD1モデルの自動生成
    - 基本的にデータの特徴を重視する

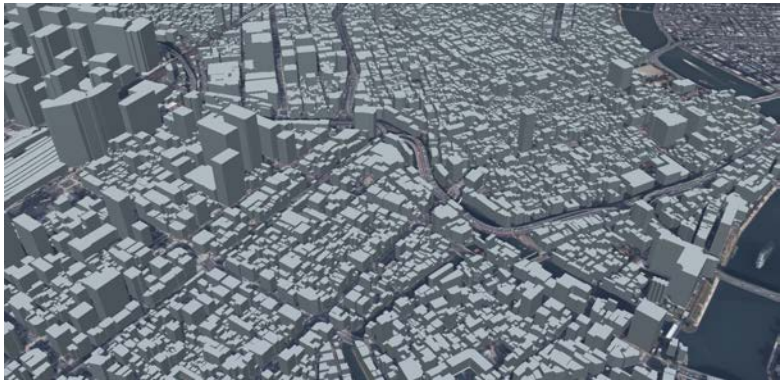


LOD1

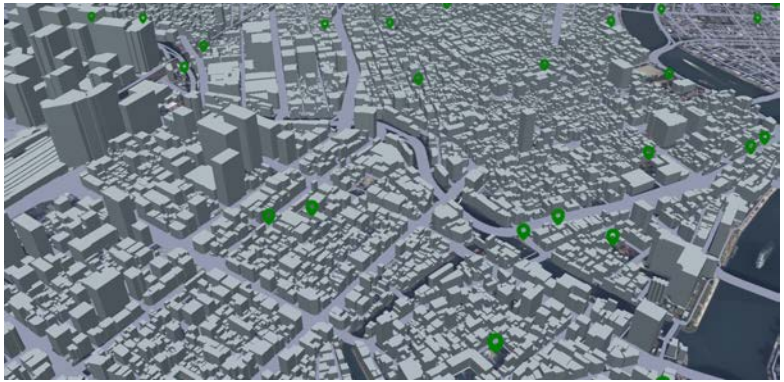
テクスチャ付きLOD1

重視するデータの特徴　その1

- 品質（位置情報）
  - 地理空間情報は様々なデータを重ねることで強みを発揮
    - 地理空間情報：空間上に位置する様々な情報のこと（e.g. 建物や道路など）
  - 品質を保つことで多様なデータを組み合わせることが可能
    - データ分析や意思決定に便利



建物のみを表示



建物と道路と公園の情報を表示

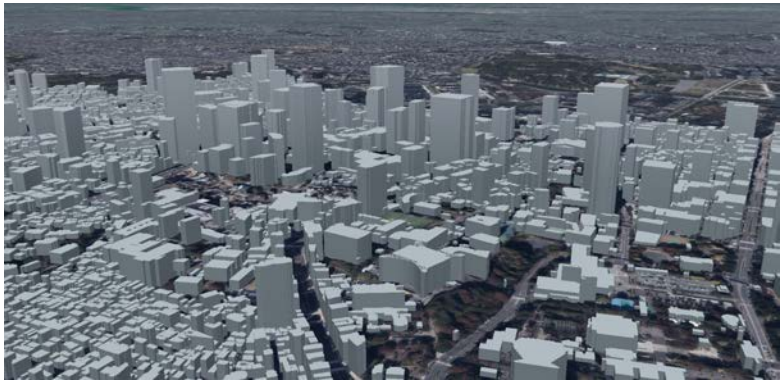
https://plateauview.mlit.go.jp/

重視するデータの特徴　その2

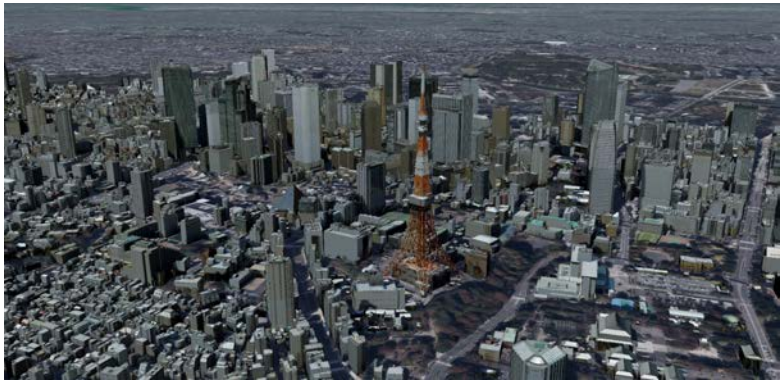
- 軽量さ
  - LOD1の軽量モデルから出来る限りデータ量を増やさない
    - テクスチャ付きでもリアルタイムで動作するように
  - LOD2だとデータ量が増えて動作が重い
    - この状態になることを避ける

参考：東京タワー近くの建物のデータ量比較

	LOD1	LOD2
建物数	915	
頂点数	22,968	246,176
テクスチャ数	0	915



LOD1（軽量）

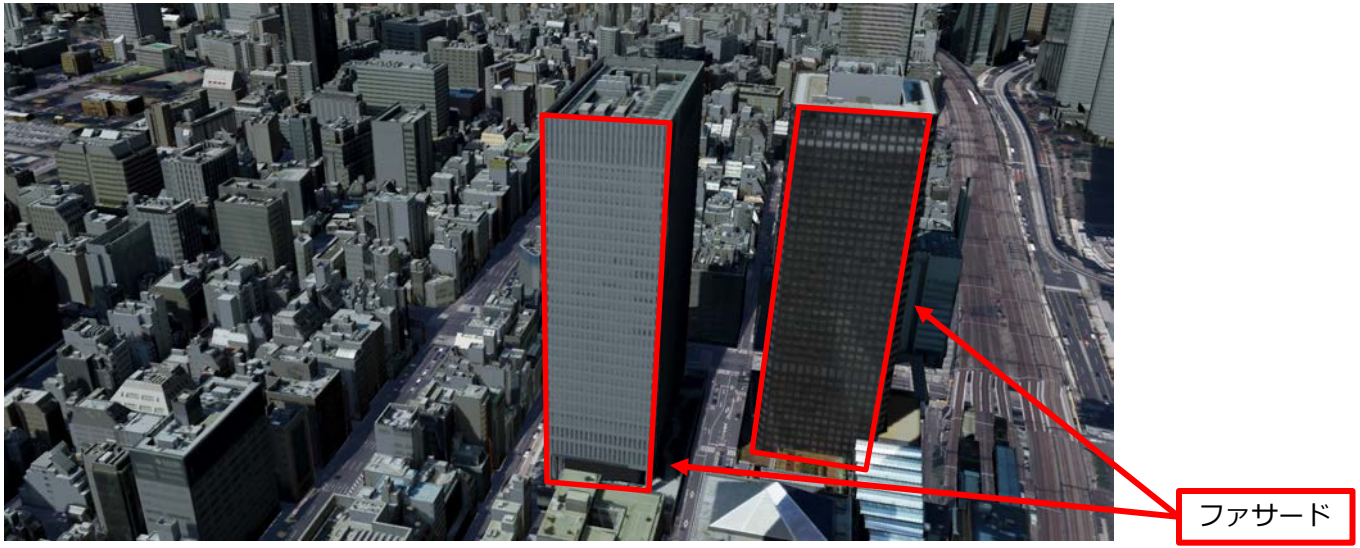


LOD2（重い）

https://plateauview.mlit.go.jp/

建物の外観（ファサード）作りの方向性

- ファサード（Facade）
  - 建物の正面から見た外観のこと
  - 本セッションでは側面や背面も含めて「ファサード」と呼ぶ



https://plateauview.mlit.go.jp/

建物の外観（ファサード）作りの方向性

Silicon Studio
Ideas × Art × Technology

- 方向性
  - 基本要素（重視するデータの特徴）
    - 元のLOD1の品質（位置情報）を保つこと
    - 元のLOD1の軽量さをできるだけ損ねないこと
  - 追加要素
    - 大量の建物を処理できること
    - 建物の実画像データが不足する中での多様な外観表現を可能にすること

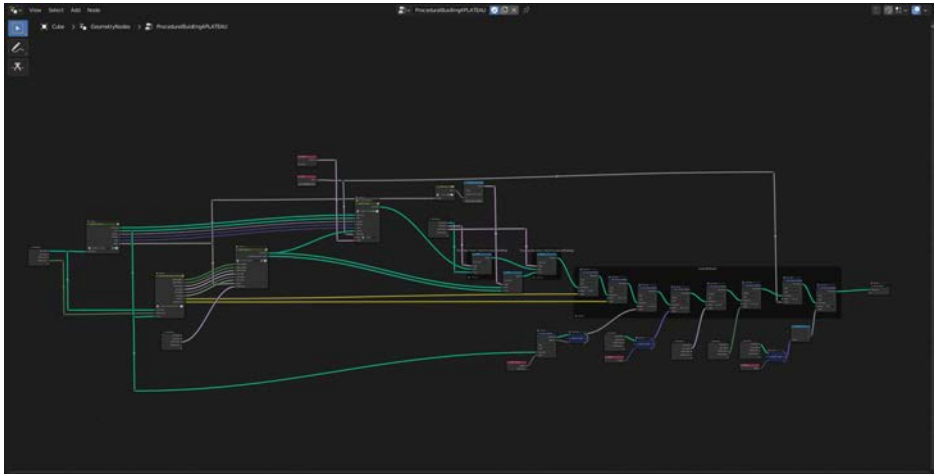
プロシージャルモデリングを用いて
ファサードを自動生成し、LOD1の再構成を行う

目次

プロシージャルモデリングとは？

Silicon Studio
Ideas × Art × Technology

- プロシージャルモデリング
  - プロシージャル：「手続きの」
  - ジオメトリと数式を組み合わせて手続き的に処理を行う手法
  - 代表的なツール：Houdini, Blender(Geometry Nodes), etc...



Geometry Node Editor

プロシージャルモデリング　利点

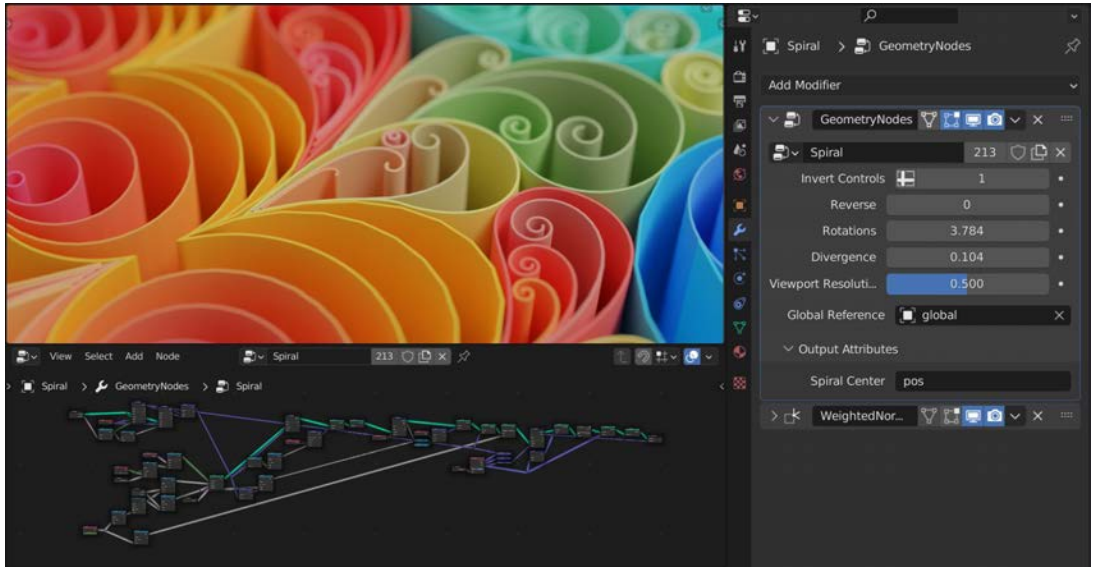
Silicon Studio
Ideas × Art × Technology

- 利点
  - 修正が容易
  - データ構造を理解しやすい
  - 仕組みを作れば大量のデータ処理が可能
  - 多様な表現を作り出せる

本セッションではPLATEAUのノウハウが蓄積されている
Blender（Geometry Nodes）を選択する

Geometry Nodes

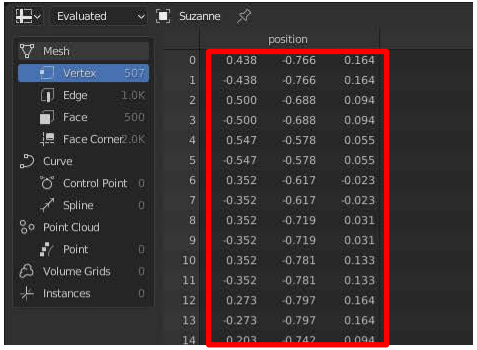
- Blenderの一機能
  - ノードベースモデリング
  - ノード（数式やジオメトリ）を組み合わせて手続き的に処理



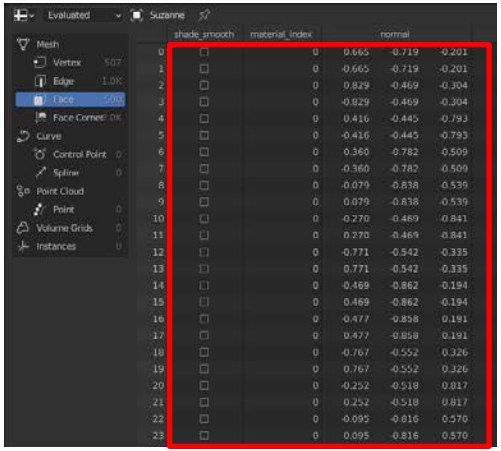
https://docs.blender.org/manual/ja/dev/modeling/geometry_nodes/introduction.html

アトリビュートの説明

- アトリビュート
  - 頂点や辺、面が所持している情報のこと
    - e.g. 頂点：「座標」、面：「法線」など
  - 自身で定義した情報を頂点や面に保存できる
  - Spreadsheetでアトリビュートを確認可能

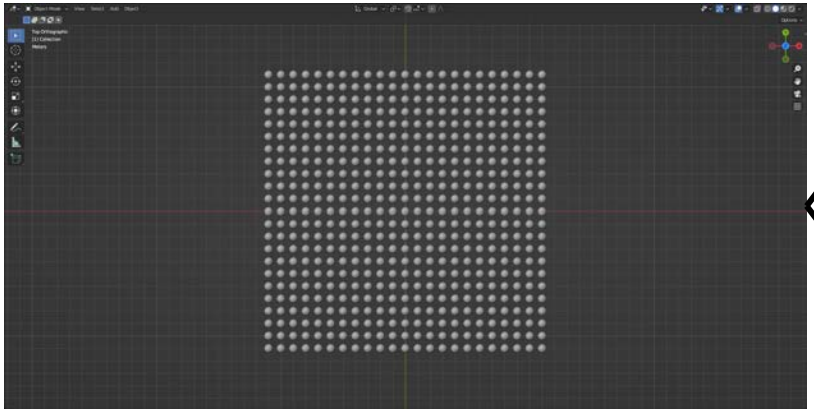


頂点：座標（x, y, z）

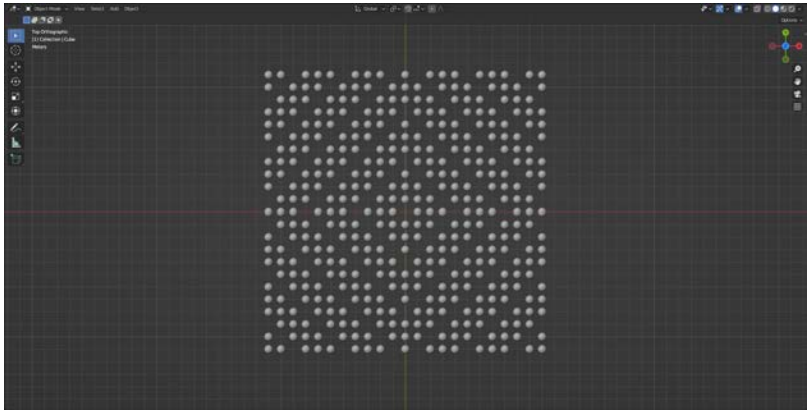


面：法線（x, y, z）など

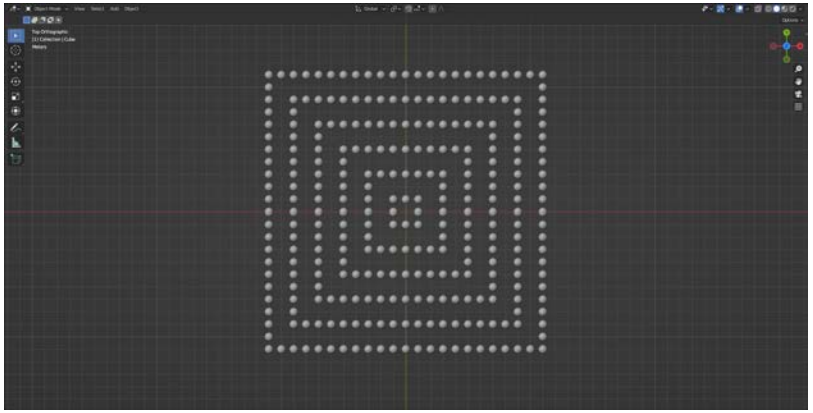
アトリビュートを用いたプロシージャルモデリング例1



22x22の球を配置

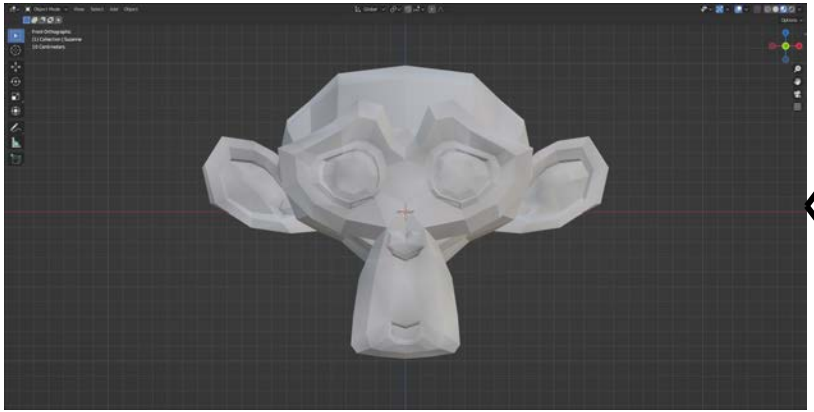


4の倍数のとき球を削除（マンハッタン距離）

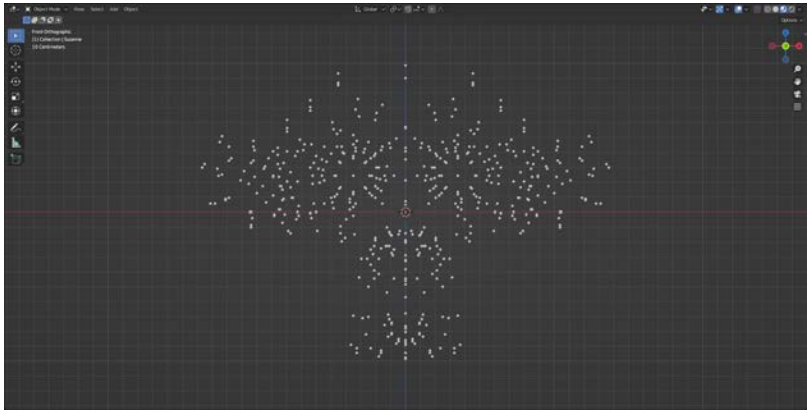


2の倍数のとき球を削除（チェビシェフ距離）

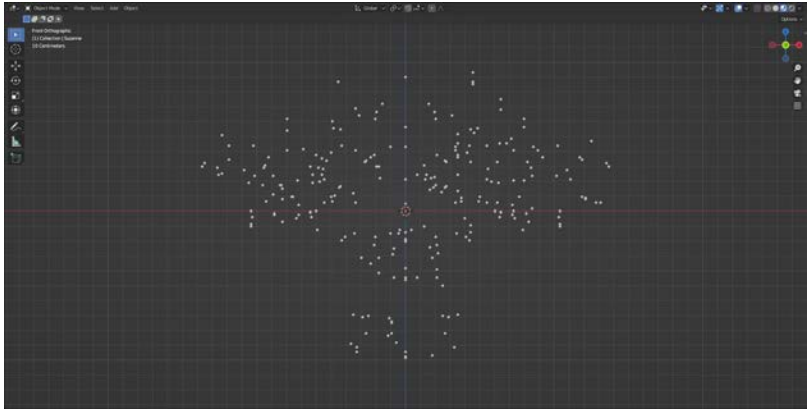
アトリビュートを用いたプロシージャルモデリング例2



スザンヌ（ノイズテクスチャ付き）



頂点に球を配置



頂点に球を配置（ノイズの値が0.5以上の場合）

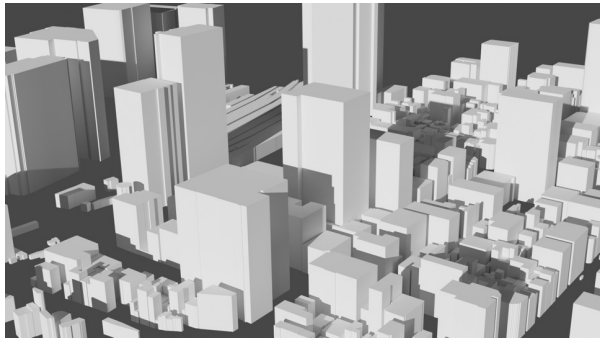
目次

Silicon Studio
Ideas × Art × Technology

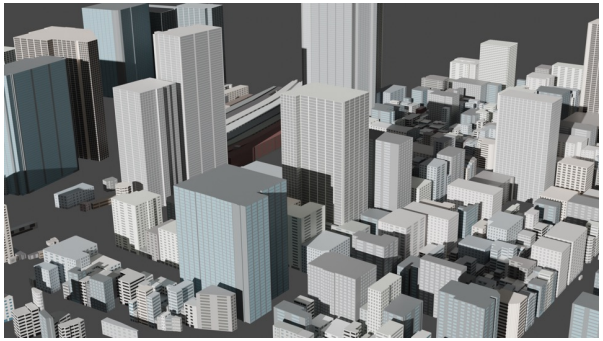
Blenderのバージョン

- Blender: 3.3.0 LTS

Procedural PLATEAU　概要



Geometry Nodes



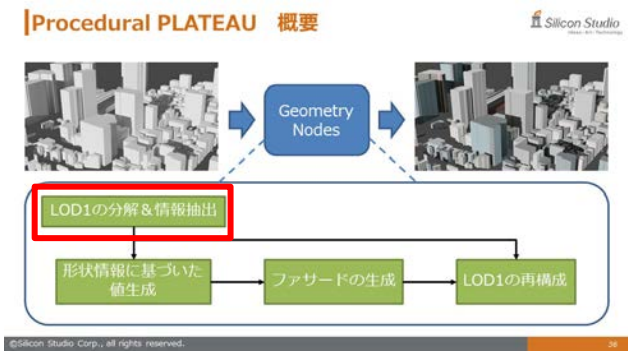
LOD1の分解＆情報抽出

形状情報に基づいた値生成

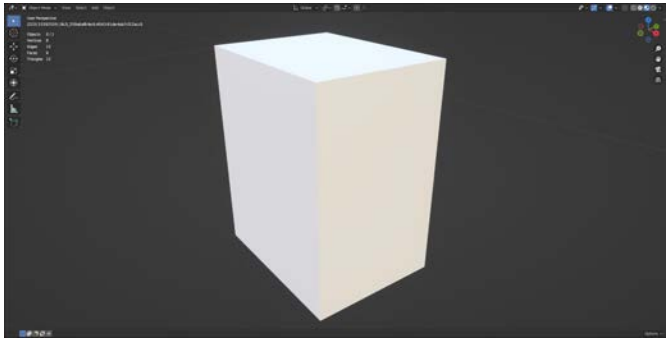
ファサードの生成

LOD1の再構成

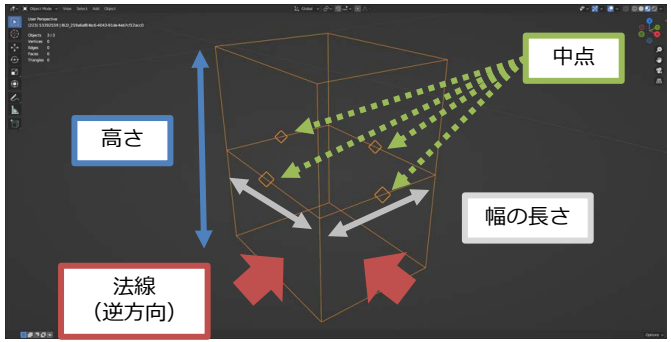
Procedural PLATEAU Flow



- LOD1の分解＆情報抽出
  - 建物の情報
    - 高さ
    - 各側面の中点
    - 各側面の法線（逆方向）
    - 各側面の幅の長さ



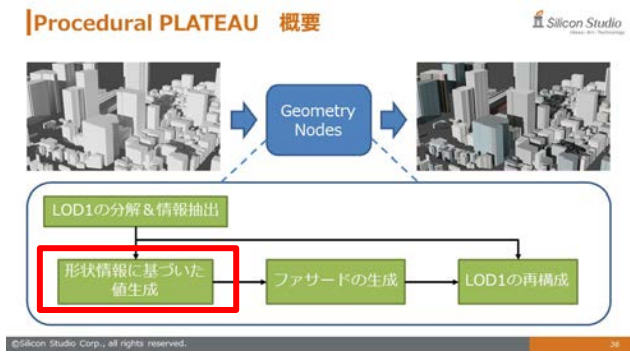
LOD1

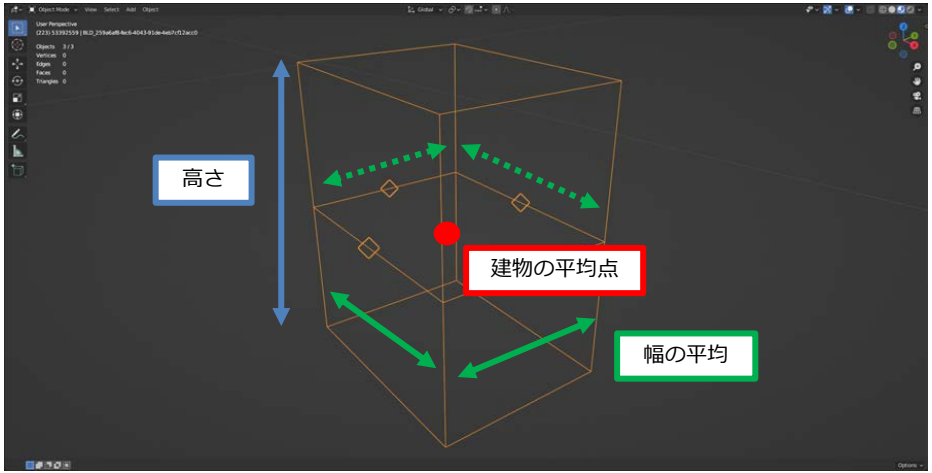


LOD1（分解）

Procedural PLATEAU Flow

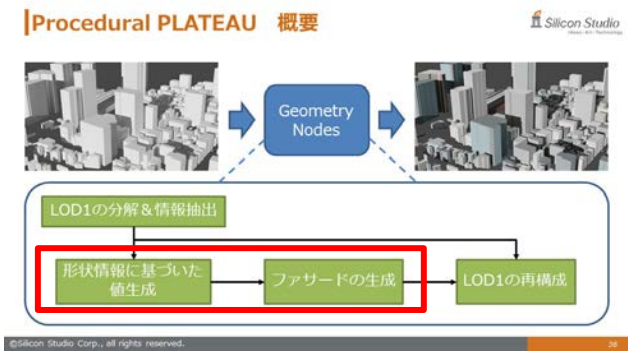
- 形状情報に基づいた値生成
  - 利用する形状情報
    - 高さ
    - （側面の）幅の平均
    - 建物の平均点

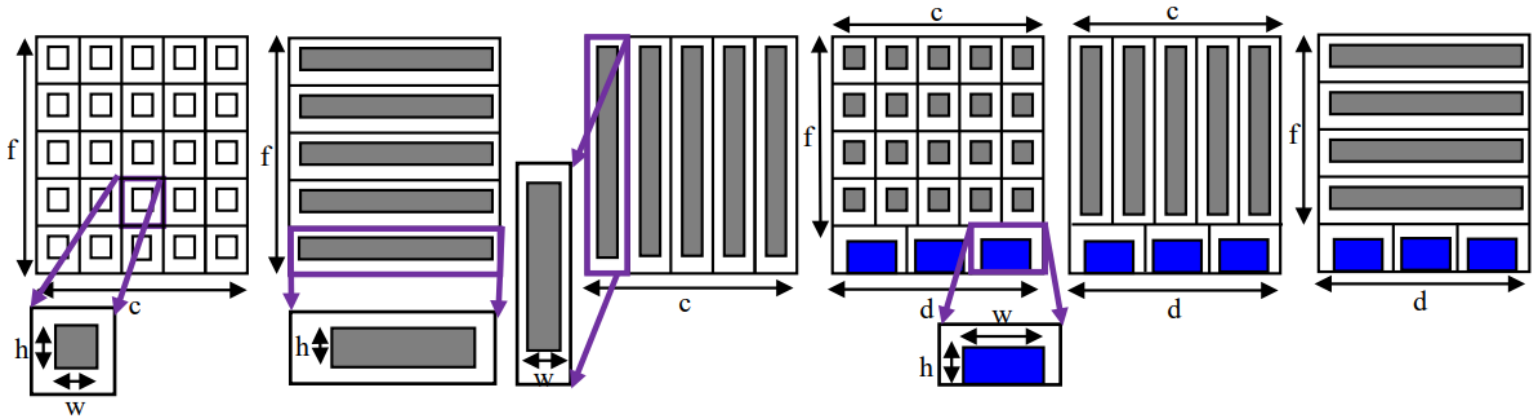




Procedural PLATEAU Flow

- ファサードの定義
  - [Zhang 2020]の文法を参考
  - 窓とドアで構成
  - 全6パターン

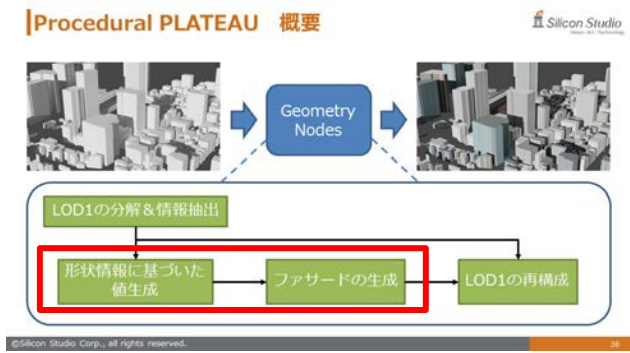


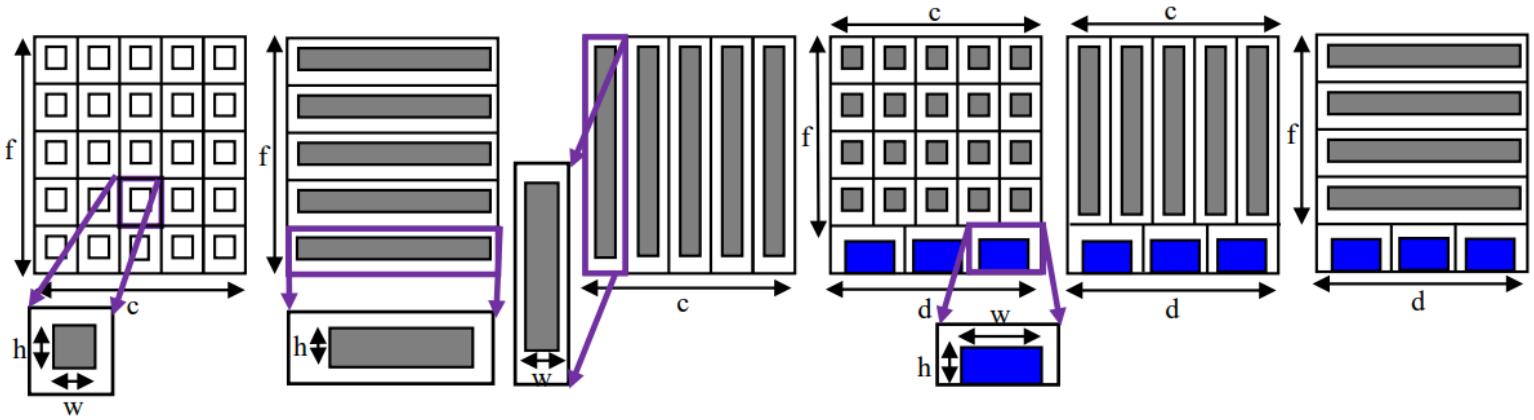


[Zhang 2020] X. Zhang et al.: Synthesis and Completion of Facades from Satellite Imagery. ECCV, 2020

Procedural PLATEAU Flow

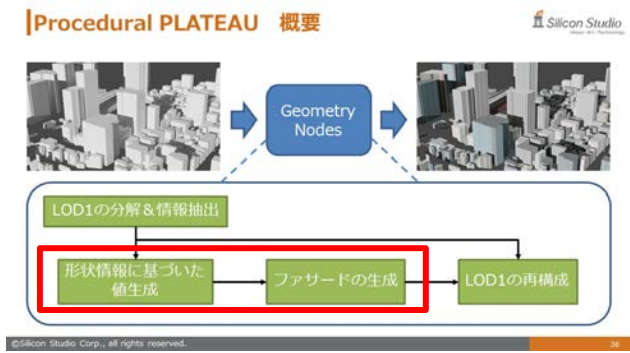
- ファサードの定義
  - 文法表現
    - Window Style
    - Door Style



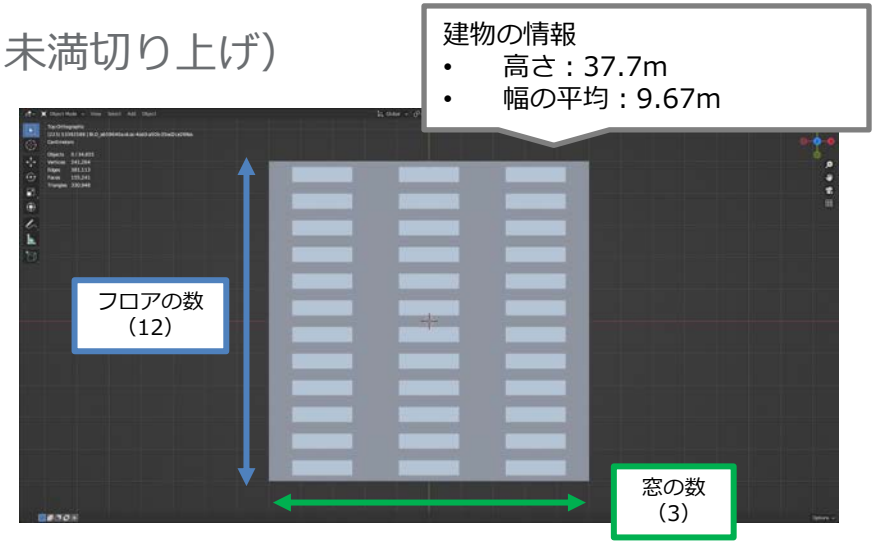


[Zhang 2020] X. Zhang et al.: Synthesis and Completion of Facades from Satellite Imagery. ECCV, 2020

Procedural PLATEAU Flow



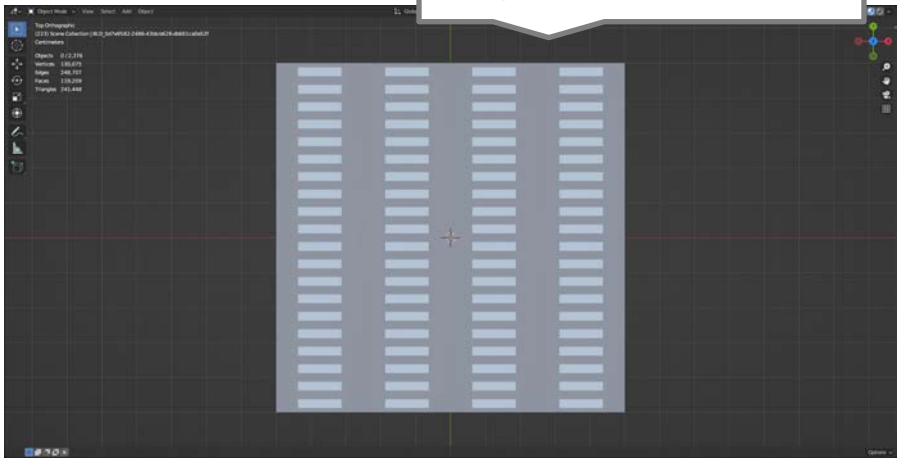
- Window Style
  - フロアの数
    - 高さ/3m（小数点以下切り捨て&1未満切り上げ）
  - （1フロアあたりの）窓の数
    - 幅の平均/3m（小数点以下切り捨て&1未満切り上げ）
  - 窓のスケール
    - フロアの数を重みとして調整
      - 多少ばらつきが得られるように調整



Procedural PLATEAU Flow

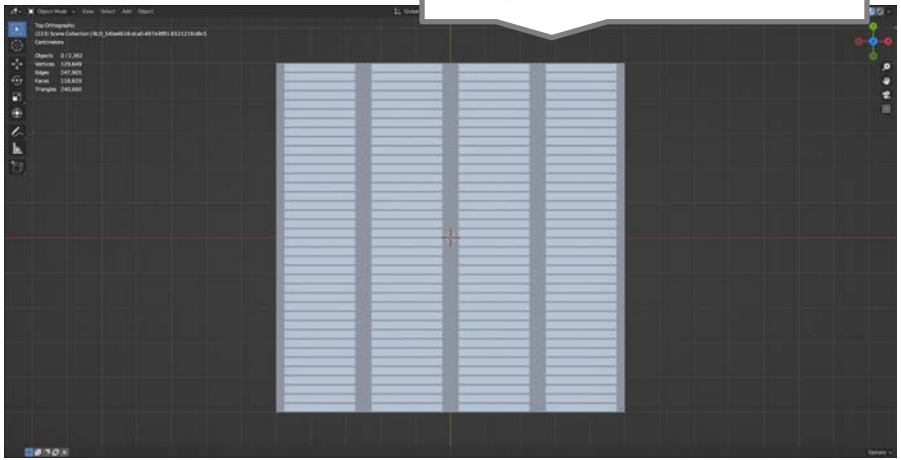
- フロアの数の違いによる窓のスケール比較（イメージ）

建物の情報
- 高さ：62.3m
- 幅の平均：14.5m



フロアの数：20
窓の数：4
窓のスケール：1.0

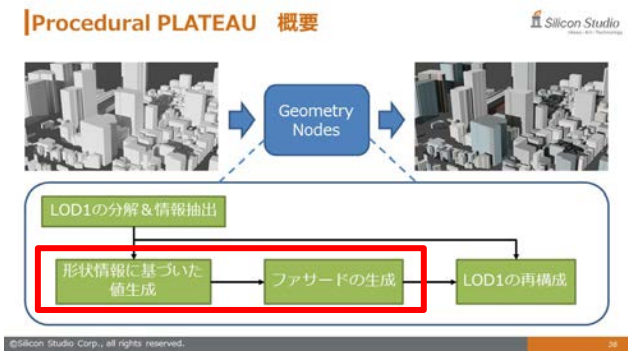
建物の情報
- 高さ：117m
- 幅の平均：13.8m

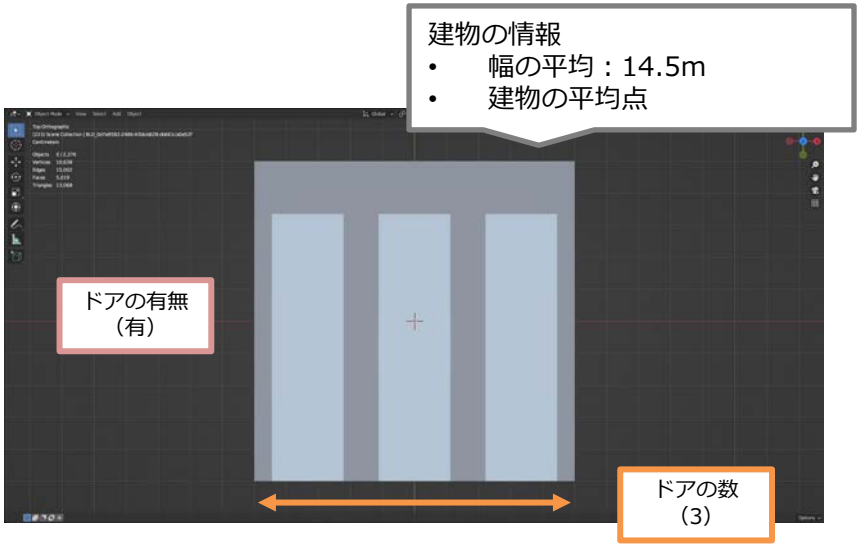


フロアの数：39
窓の数：4
窓のスケール：1.6

Procedural PLATEAU Flow

- Door Style
  - ドアの有無
    - 乱数で調整
      - Seed値は建物の平均点
  - ドアの数
    - 窓の数を最大値とした乱数
      - 乱数の範囲：1~窓の数
      - Seed値は建物の平均点

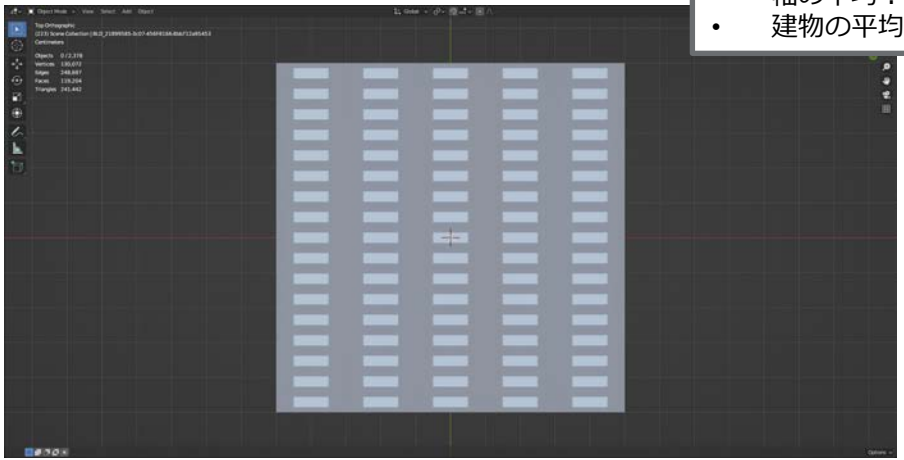




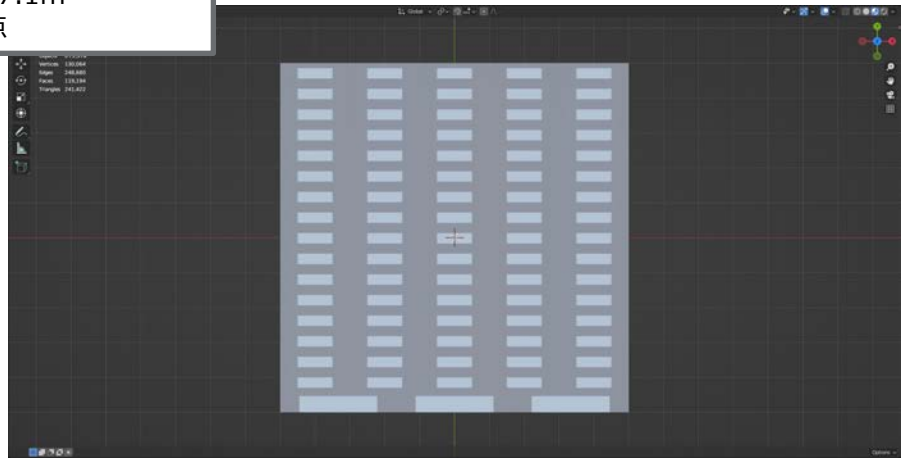
Procedural PLATEAU Flow

- 組み合わせの有無の比較（イメージ）

建物の情報
- 高さ：53.9m
- 幅の平均：17.1m
- 建物の平均点



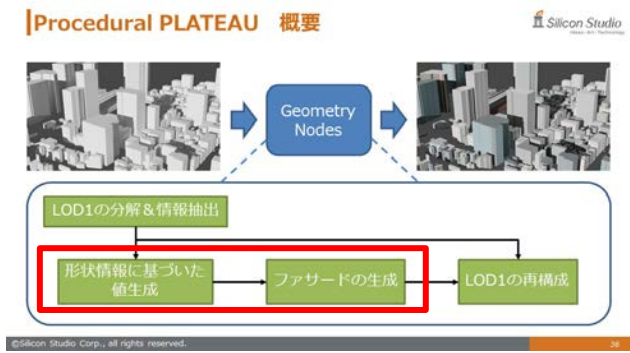
Window Styleのみ※



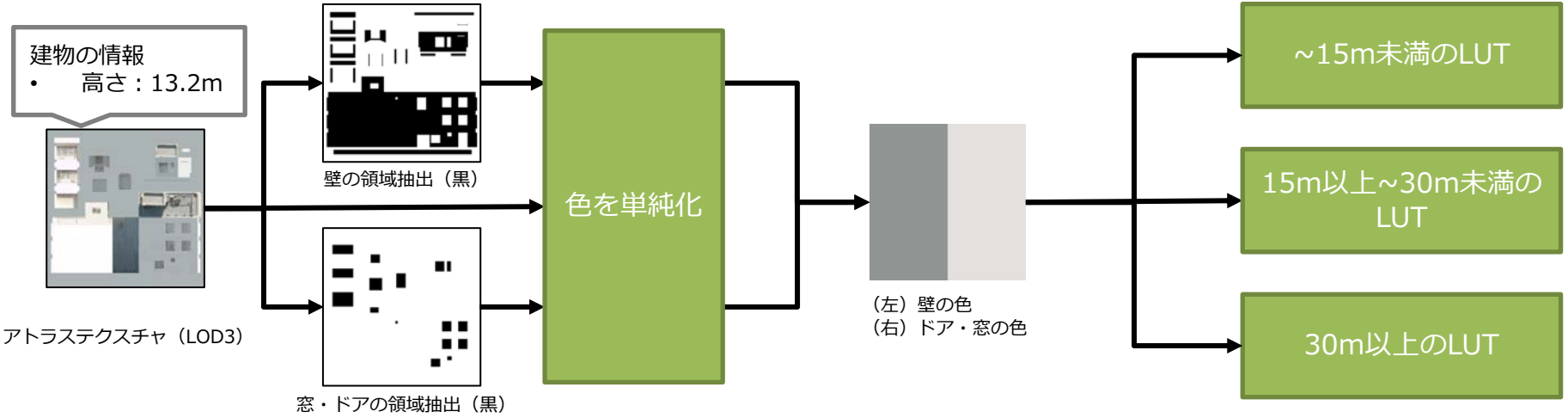
Window Style + Door Style

※ドアの有無のパラメータが無のとき

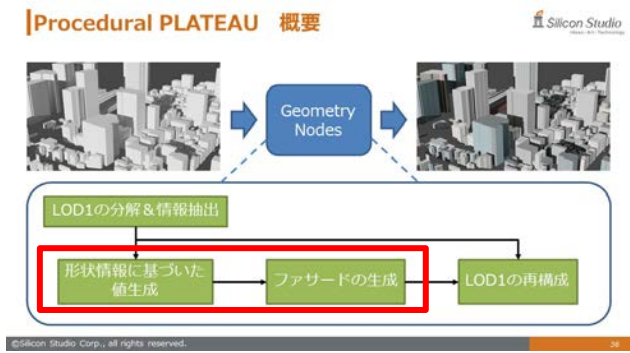
Procedural PLATEAU Flow



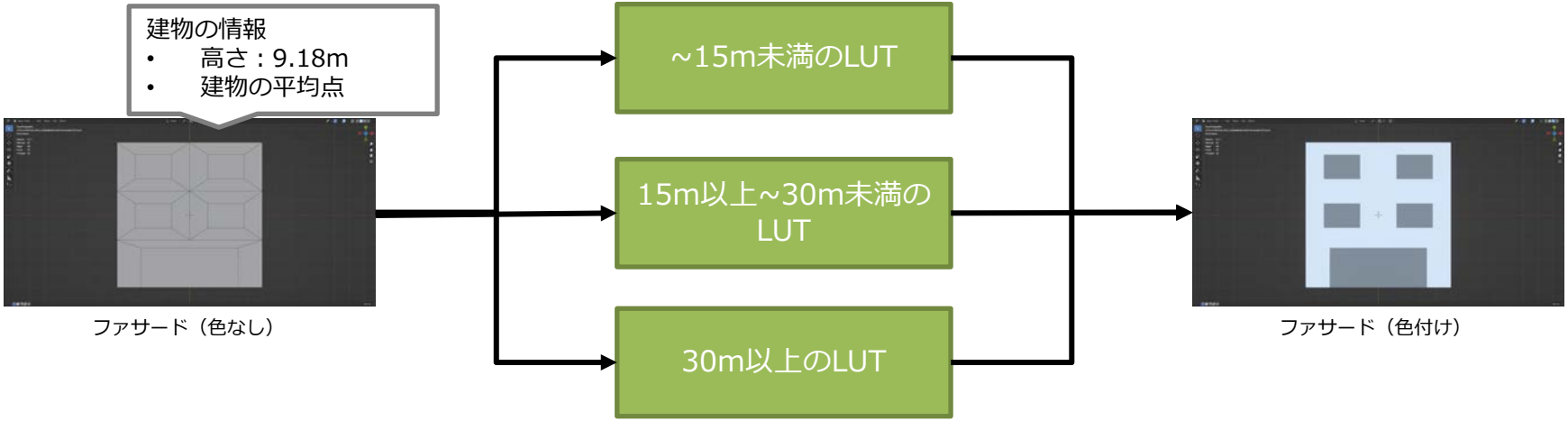
- ファサードの色（前処理）
  - 静岡県沼津市のLOD3モデルを使用
    - ドアや窓などが構造化されている
    - ドア・窓と壁のUV座標を用いて色を単純化
    - 抽出した色の建物の高さに応じてLUTを作成（グループ化）



Procedural PLATEAU Flow

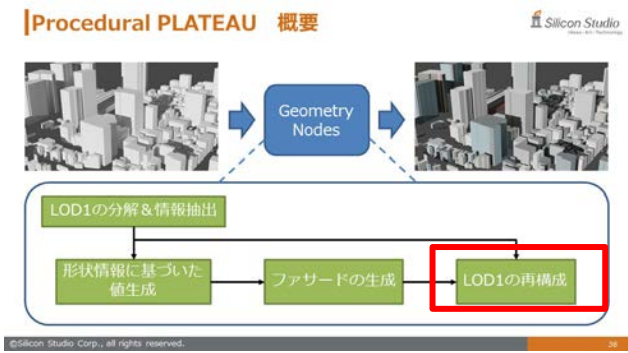


- ファサードの色（Geometry Nodes）
  - LOD1の建物の高さから参照するLUTを選択
    - LUT内の色ペアの選び方は乱数で調整
  - 出力された色ペアをファサードのそれぞれの領域に割り当てる

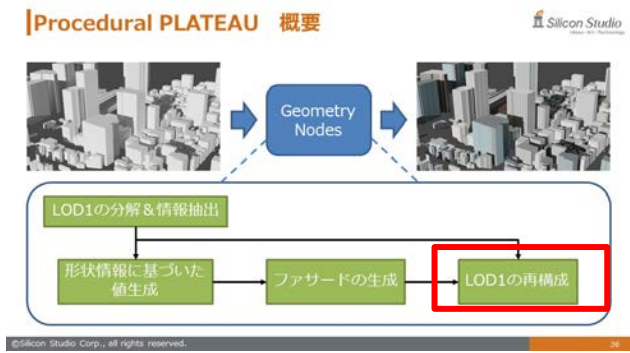


Procedural PLATEAU Flow

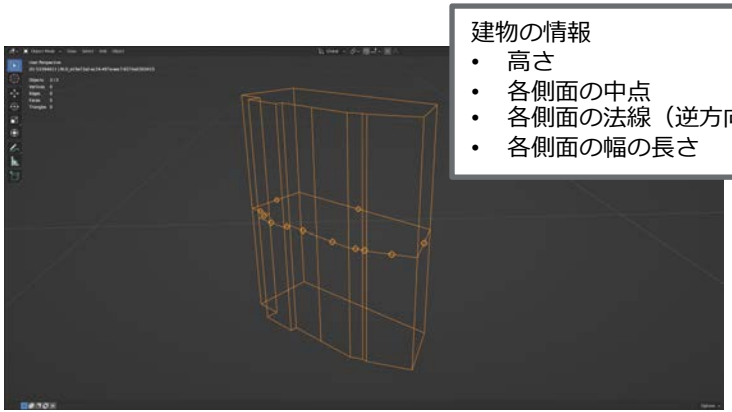
- LOD1の再構築
  1. 側面は元のLOD1の情報とファサードから作成
  2. 底面と上面は元のLOD1を使用
  3. 一定の条件で単色のファサードを表示



Procedural PLATEAU Flow



- LOD1の再構築
  1. 側面は元のLOD1の情報とファサードから作成
  2. 底面と上面は元のLOD1を使用
  3. 一定の条件で単色のファサードを表示

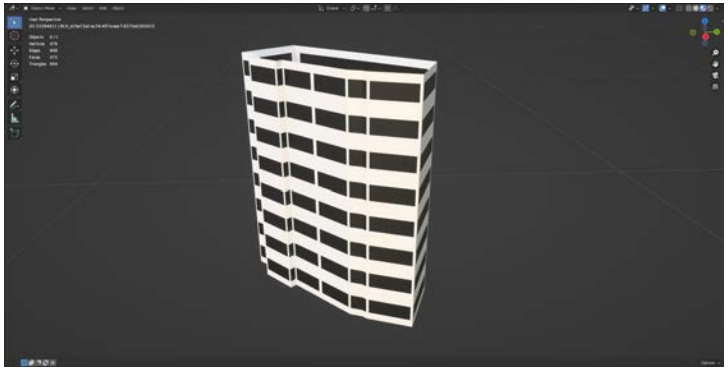


建物の情報

- 高さ
- 各側面の中点
- 各側面の法線（逆方向）
- 各側面の幅の長さ

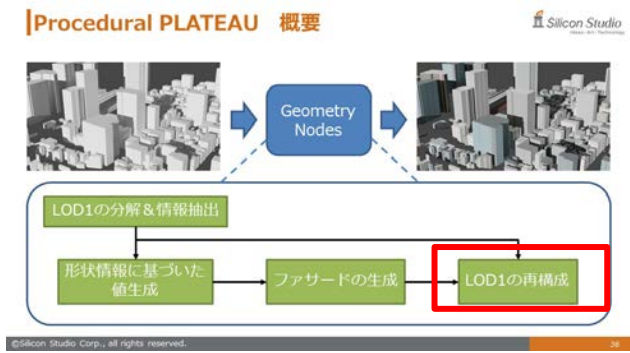
建物の情報

生成されたファサード

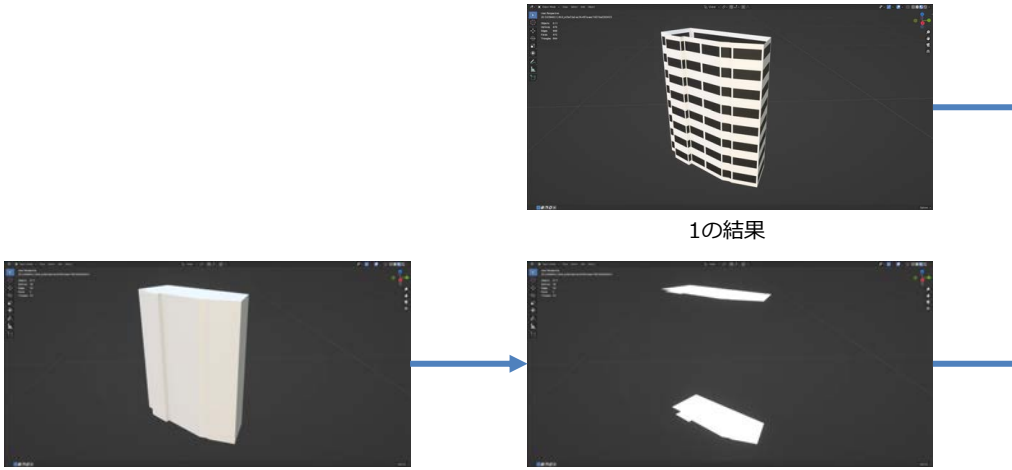


建物の情報を基にファサードを側面に配置

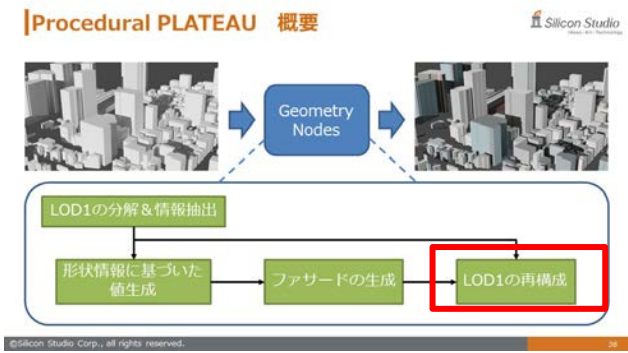
Procedural PLATEAU Flow



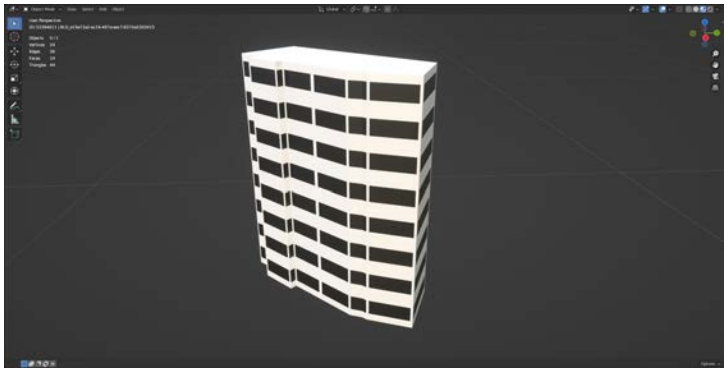
- LOD1の再構築
  1. 側面は元のLOD1の情報とファサードから作成
  2. 底面と上面は元のLOD1を使用
  3. 一定の条件で単色のファサードを表示



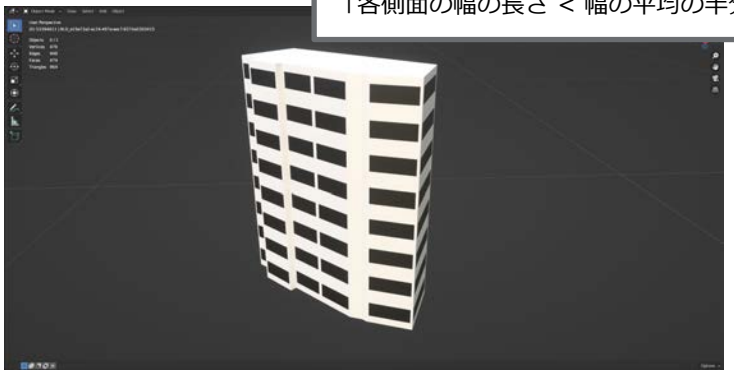
Procedural PLATEAU Flow



- LOD1の再構築
  1. 側面は元のLOD1の情報とファサードから作成
  2. 底面と上面は元のLOD1を使用
  3. 一定の条件で単色のファサードを表示



2の結果

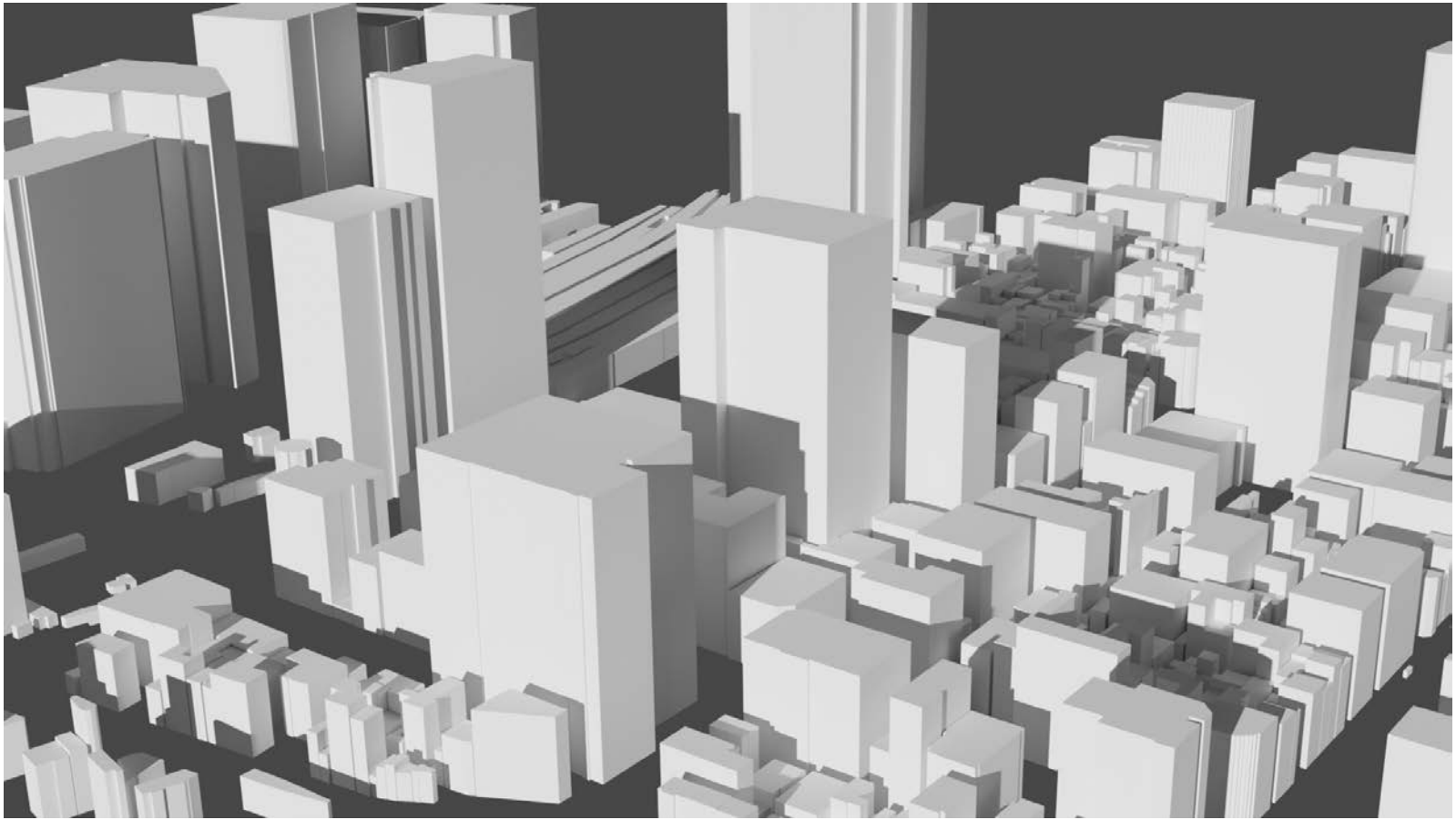


一定の条件で単色のファサードを表示

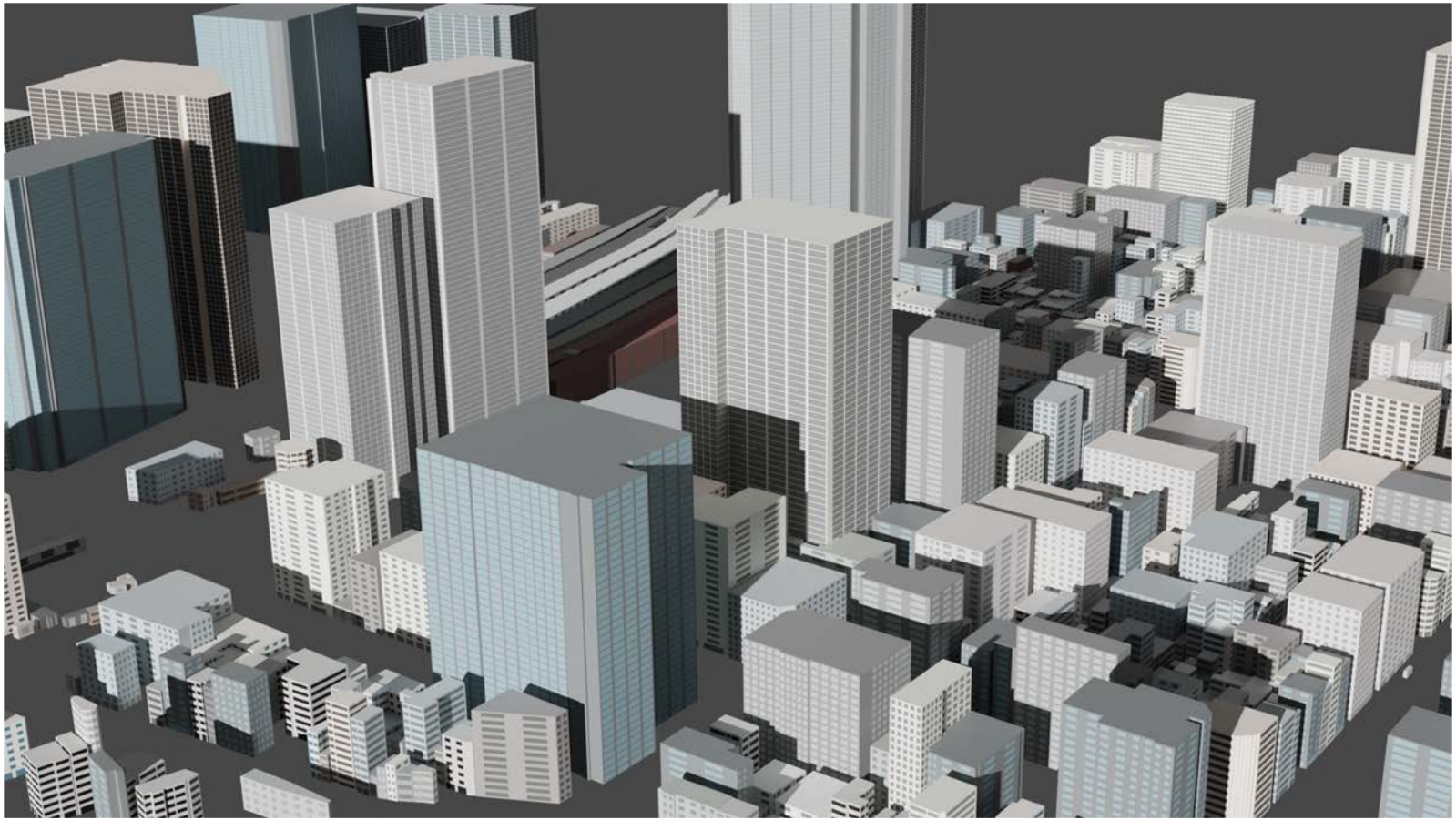
一定の条件
「各側面の幅の長さ < 幅の平均の半分」

LOD1の比較（オリジナル）

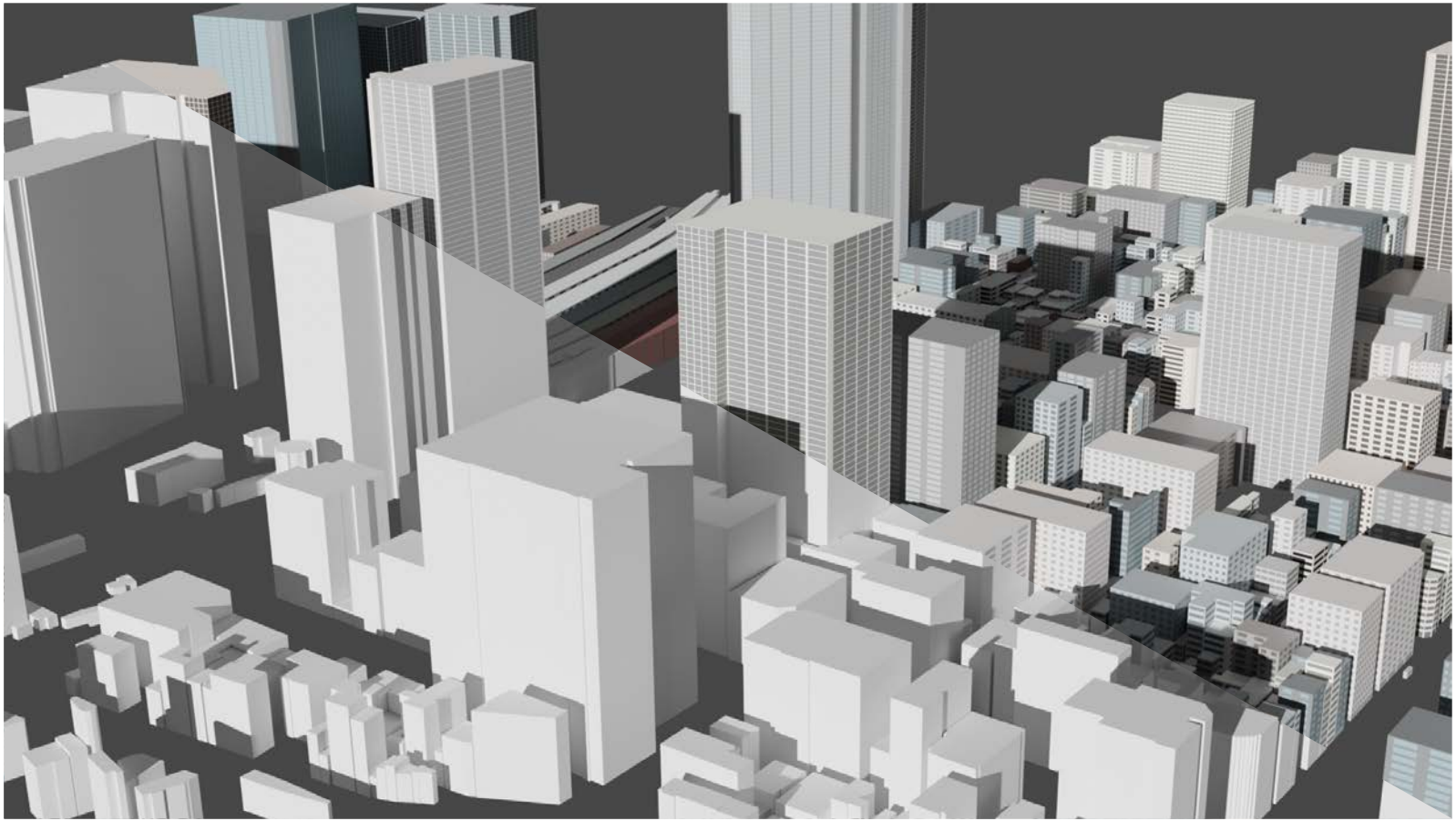
Silicon Studio
Ideas × Art × Technology



LOD1の比較（再構成したLOD1）



LOD1の比較（両者）

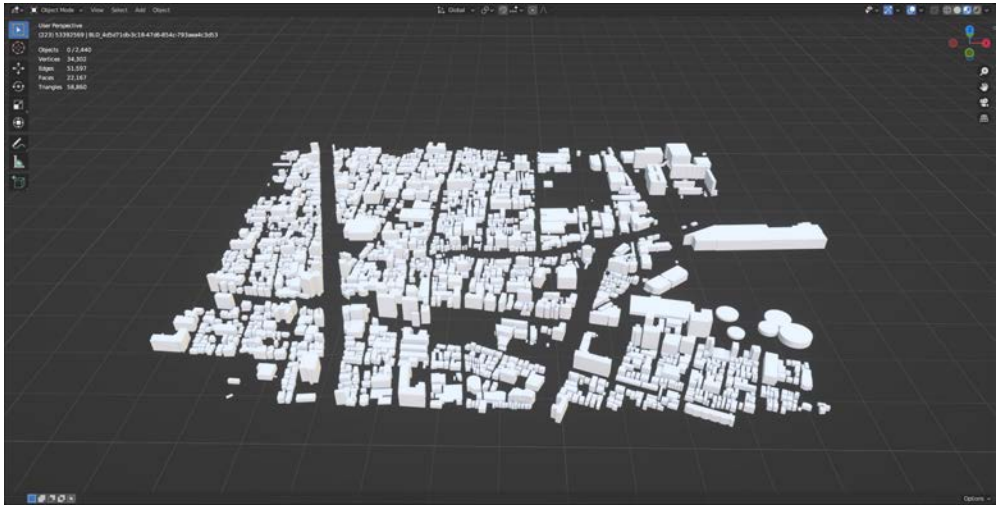


目次

Blenderアドオンの開発経緯

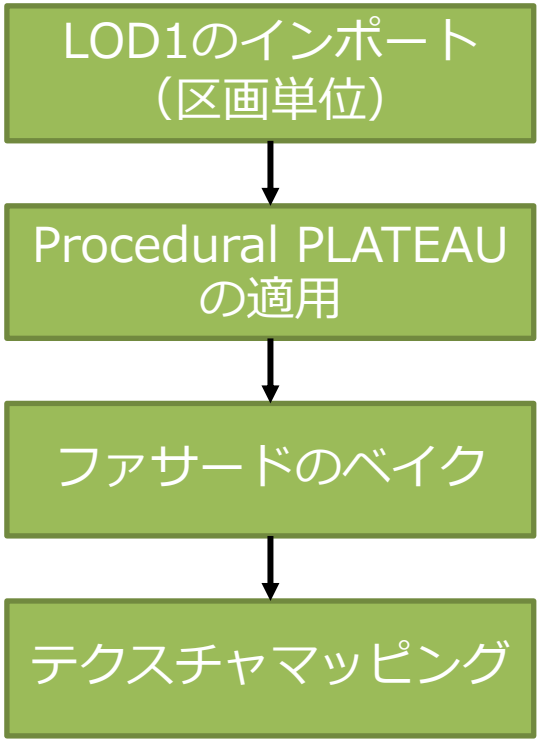
Silicon Studio
Ideas × Art × Technology

- 建物の数が多い
  - 都市部では約1km四方に数千単位の建物がある
  - 手動ですべての建物にテクスチャ付きLOD1を作るのは困難
    →アドオンで自動化



Blenderアドオンの概要

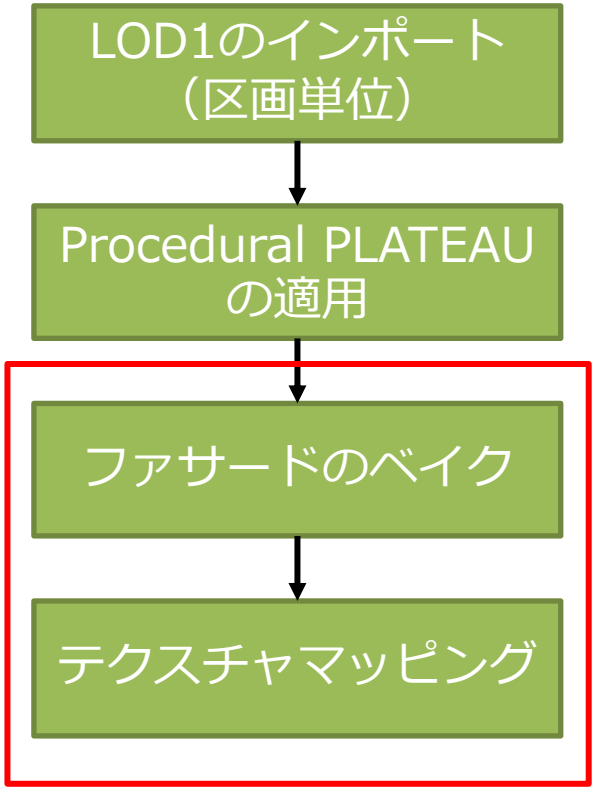
- アドオンの処理内容
  1. LOD1のインポート
  2. LOD1に対してProcedural PLATEAUの適用
  3. ファサードのベイク
  4. テクスチャマッピング



Blenderアドオンの概要

- アドオンの処理内容
  1. LOD1のインポート
  2. LOD1に対してProcedural PLATEAUの適用
  3. ファサードのベイク
  4. テクスチャマッピング

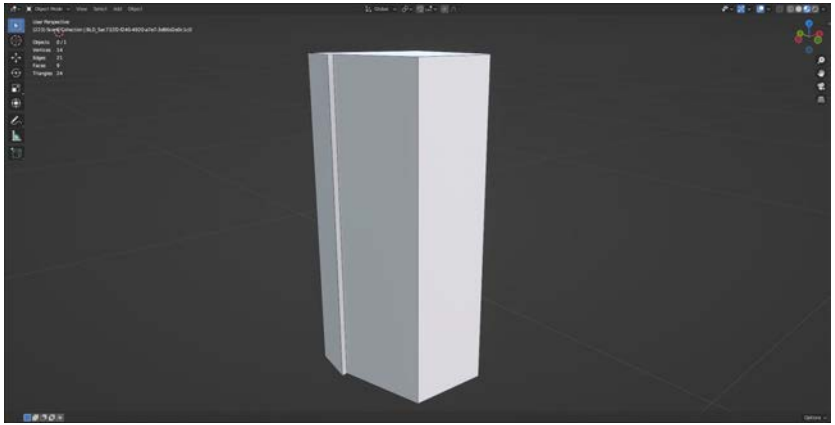
時間の都合上、
ファサードのベイクとテクスチャマッピングの処理を説明



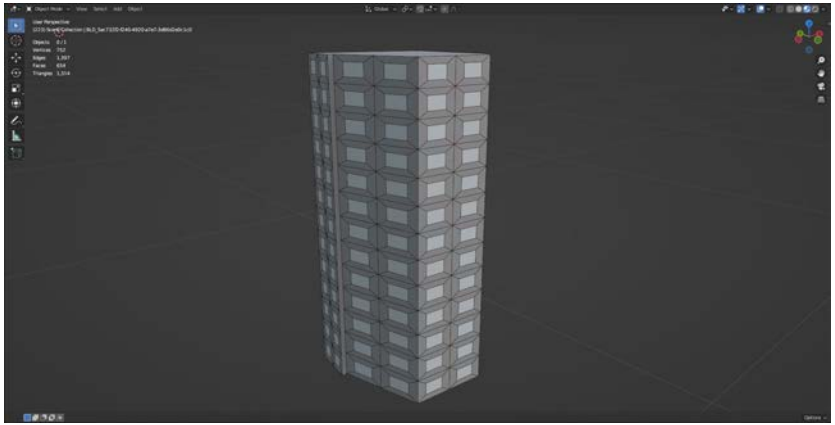
目次

ファサードをベイクする理由

- LOD1の軽量さを維持



LOD1



再構成したLOD1

Vertices: 14
Edges: 21
Faces: 9
Triangles: 24

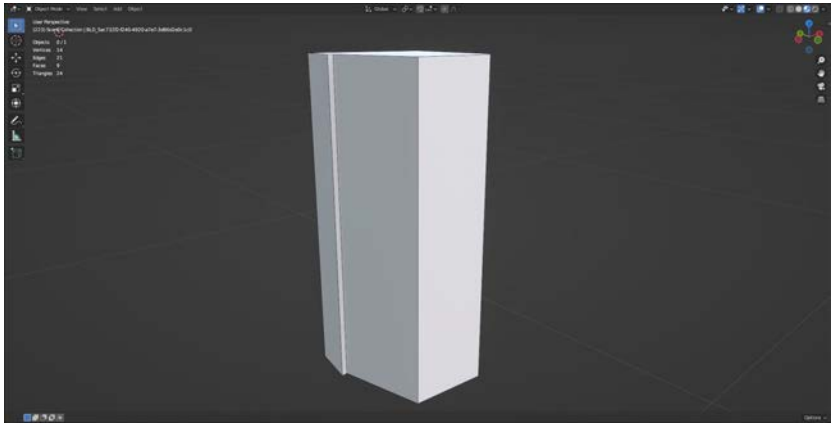
<

Vertices: 752
Edges: 1397
Faces: 654
Triangles: 1314

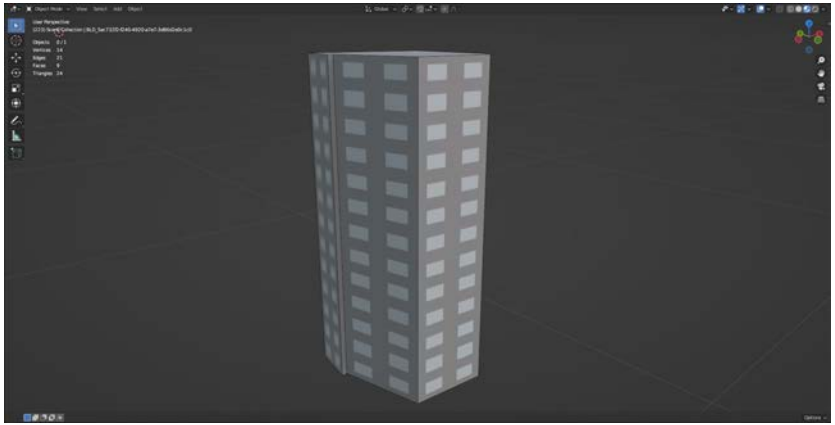
※Realize Instances時

ファサードをベイクする理由

- LOD1の軽量さを維持



LOD1



テクスチャ付きLOD1

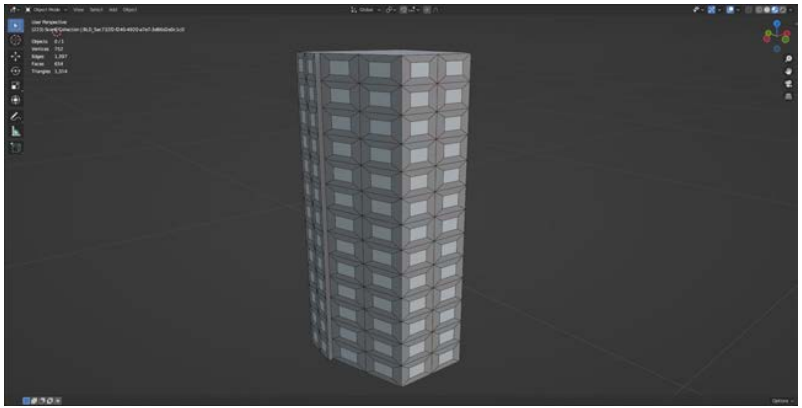
Vertices: 14
Edges: 21
Faces: 9
Triangles: 24

=

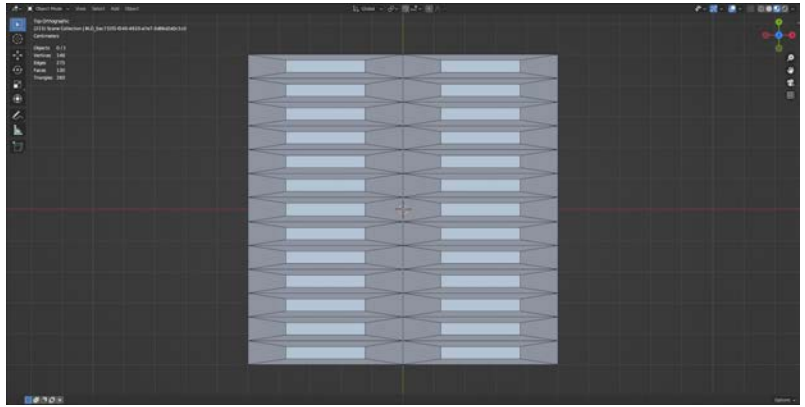
Vertices: 14
Edges: 21
Faces: 9
Triangles: 24

再構成したLOD1からファサードへ

- Procedural PLATEAUの処理を巻き戻し
  - ファサードを縦横1mの正方形に
  - 窓・ドアの伸び縮みは許容
  - ファサード（Geometry Nodes）をメッシュに変換



再構成したLOD1



ファサード（メッシュ化済み）

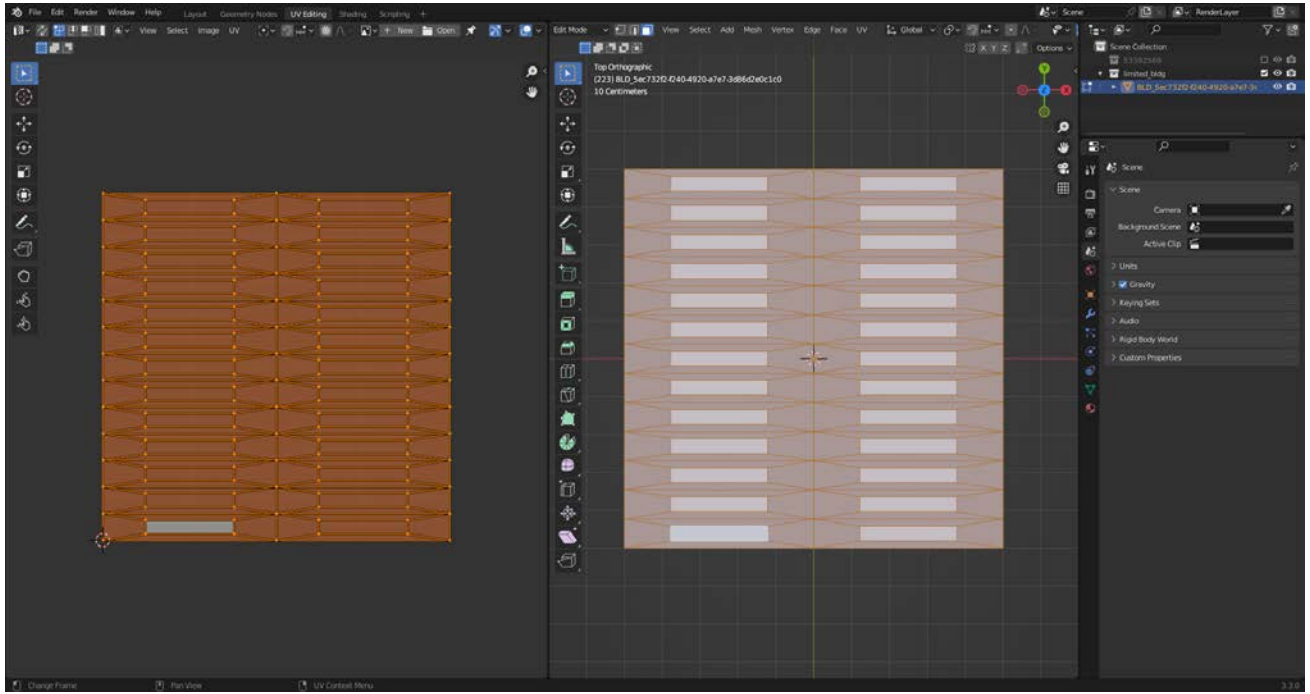
ベイク設定

Silicon Studio
Ideas × Art × Technology

- ベイク設定
  - レンダリングエンジン
    - Cycles
  - サンプル数
    - 1に設定（単純な色しかないため、1で十分）
  - ベイクタイプ
    - DIFFUSEに設定
    - アルベドのみを使用

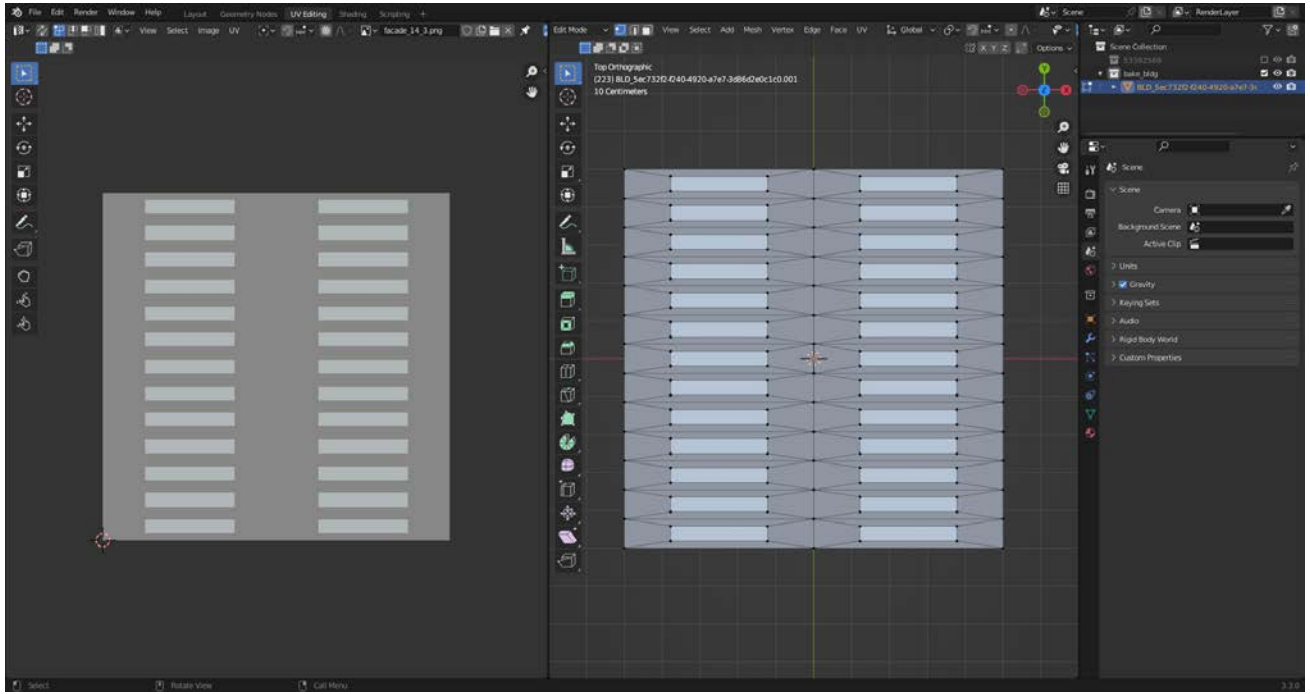
ファサードのUV展開

- ファサードを1つの画像として考える
  - 真上から見た状態でUV展開



ファサードをベイク

- 画像として保存
  - 解像度：256x256（固定）

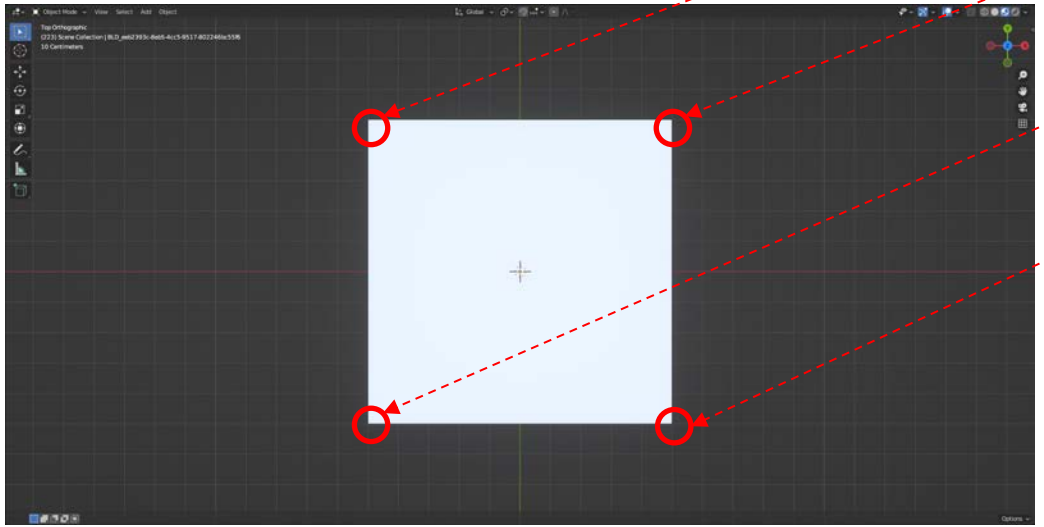


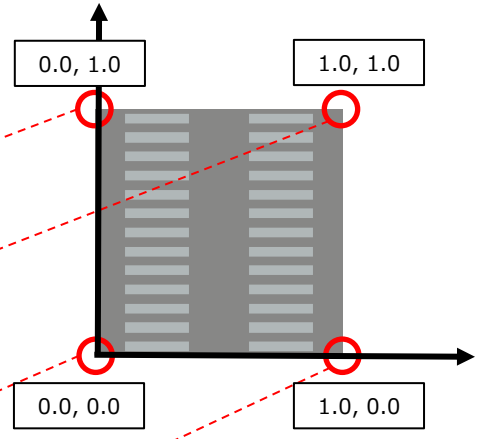
目次

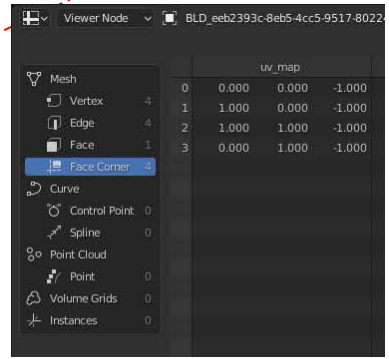
UV座標を設定した平面の作成

- Geometry Nodes内で平面を作成
  - 画像を上から貼り付けるようにUV座標を設定※
  - UV座標はアトリビュートに保存

※フットプリントなどはすべて0に設定



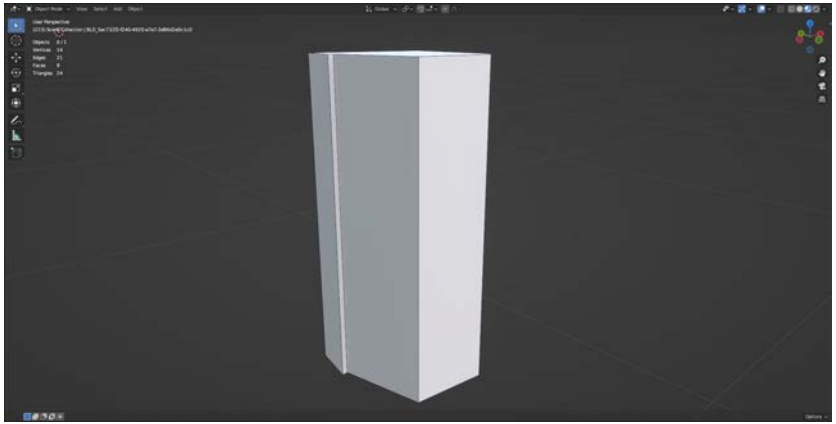




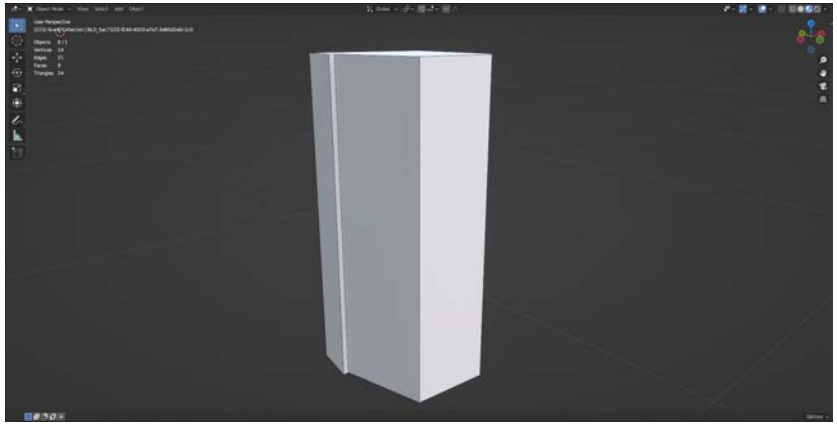
平面のUV座標（アトリビュート）

UV座標付き平面を用いたLOD1の再構成

- UV座標付き平面
  - Procedural PLATEAUのファサード部分に組み込む
  - オリジナルのLOD1と同等のLOD1を作成



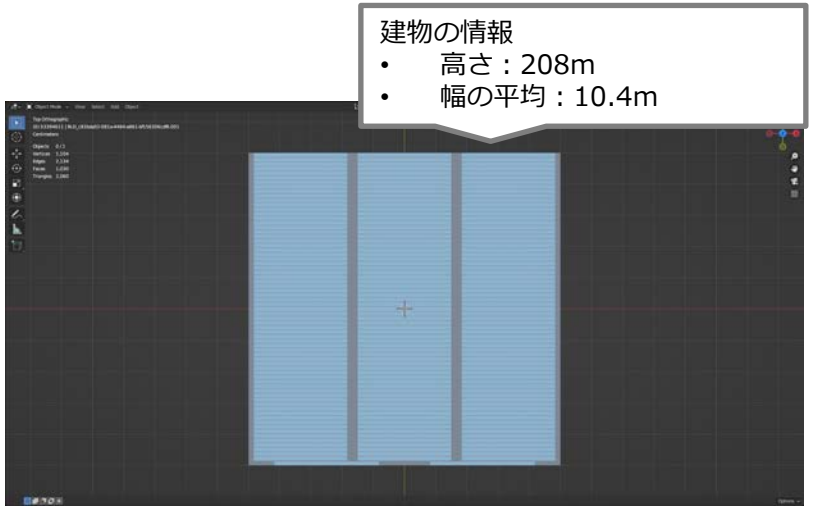
LOD1
（オリジナル）



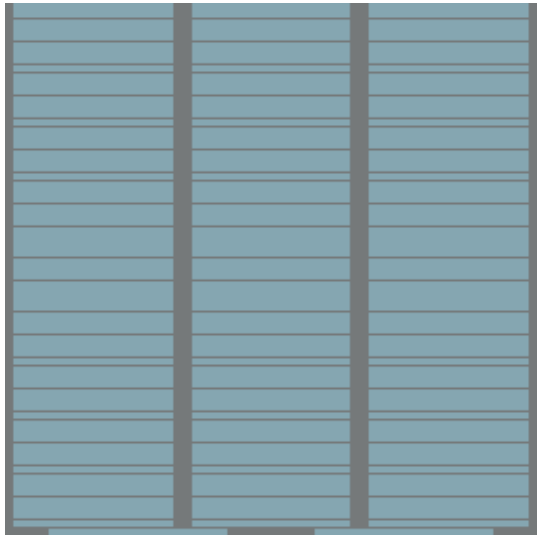
再構成したLOD1
（UV座標付き）

テクスチャの解像度問題

- 高い建物では窓の領域が潰れる
  - 解像度を上げれば改善できるが、良い方法ではない



ファサード

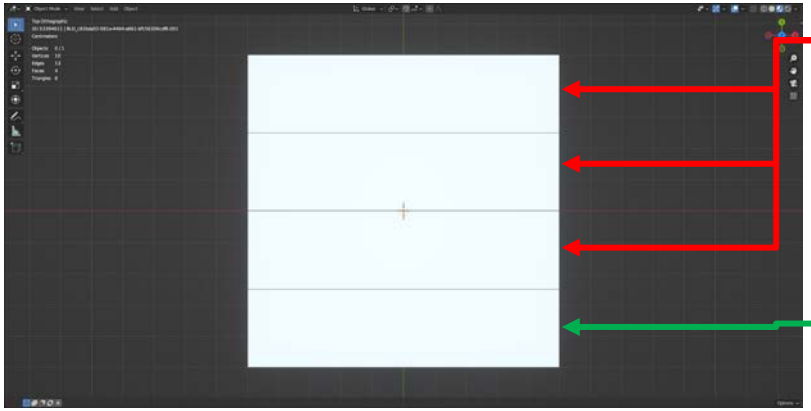


ベイクされたテクスチャ
（解像度：256x256）

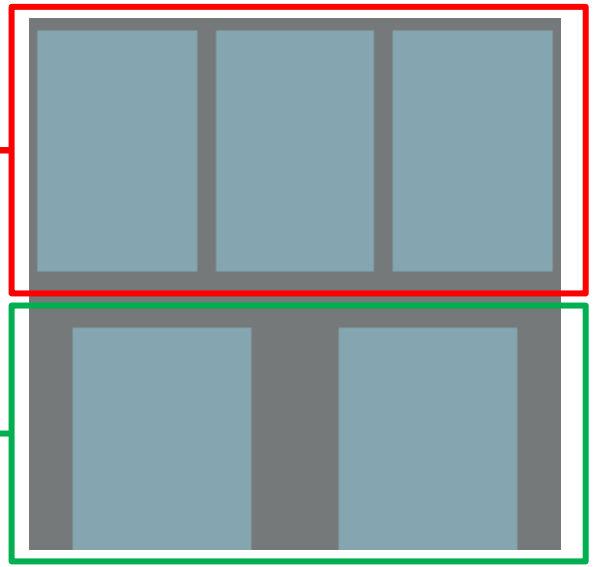
テクスチャの解像度問題のアプローチ（イメージ）

Silicon Studio
Ideas × Art × Technology

- 各フロアに頂点を持たせる
  - 1フロアあたりの窓とドアのテクスチャを作る
  - それぞれの領域のUV座標を各頂点に設定
  - テクスチャ解像度の固定維持



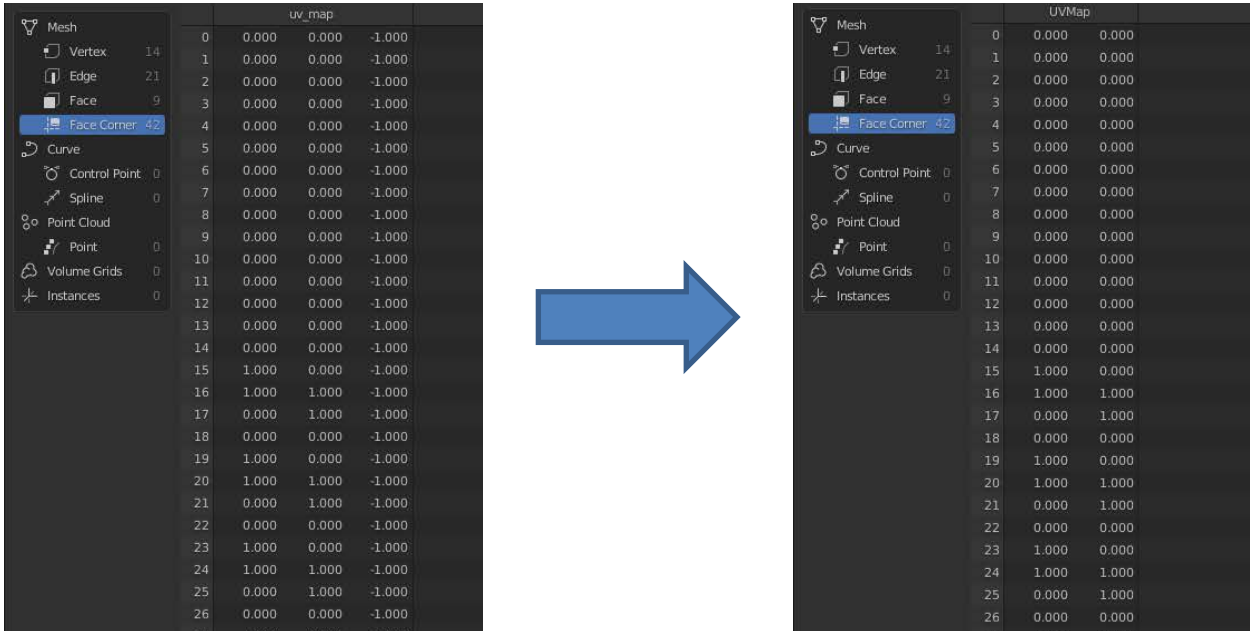
フロアごとに分割したファサードに割り当て



1フロアあたりの窓とドアのテクスチャ
（解像度：256x256）

UV情報の変換

- UV情報
  - アトリビュートのUV（Vector3）をメッシュのUV（Vector2）に変換



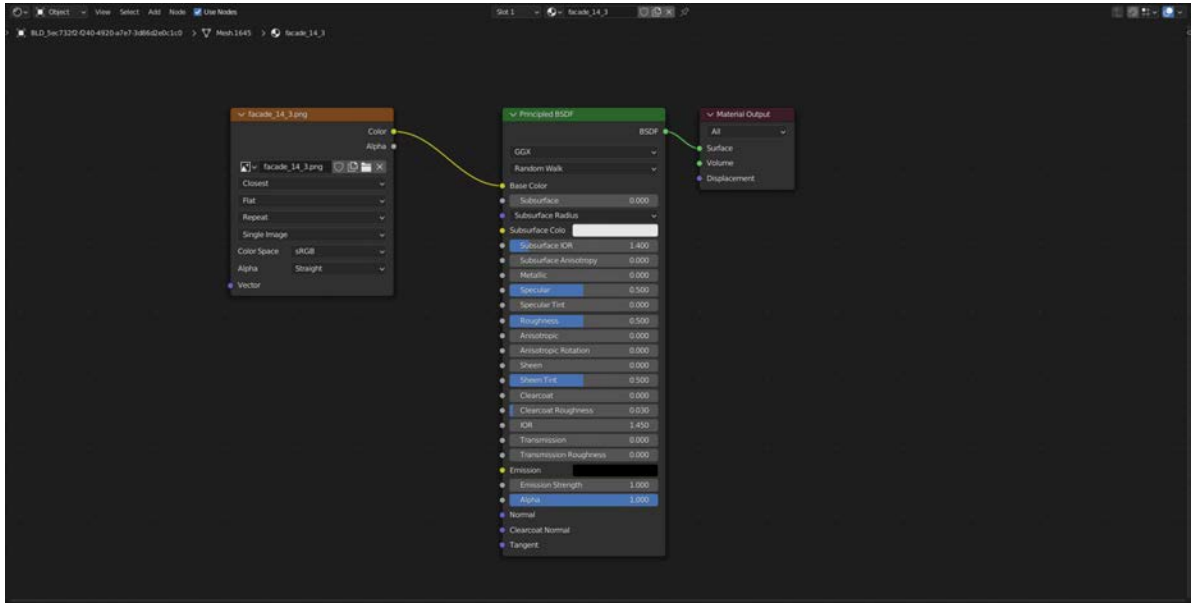
Geometry NodesのUV情報（Vector3）

メッシュのUV情報（Vector2）

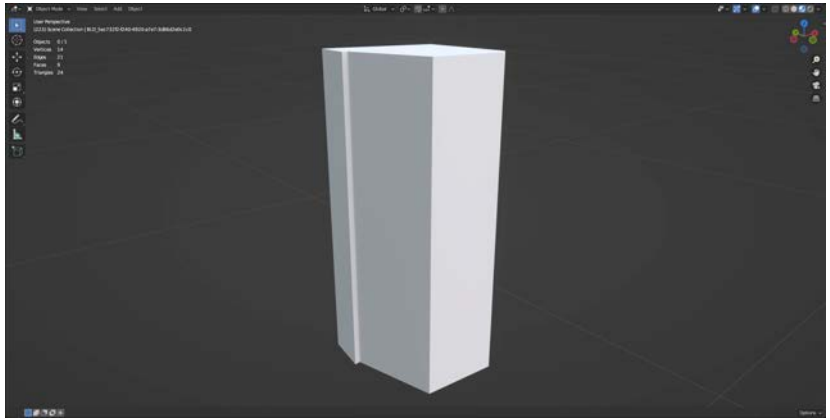
マテリアルの作成

Silicon Studio
Ideas × Art × Technology

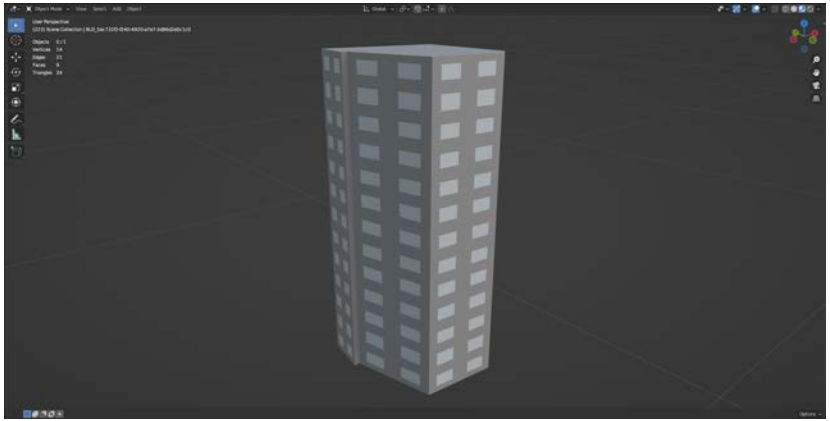
- Principled BSDF
  - 保存したテクスチャをBase Colorに接続
  - 補間はClosestに設定



再構成したLOD1にマテリアル付与



再構成したLOD1
（UV座標付き）

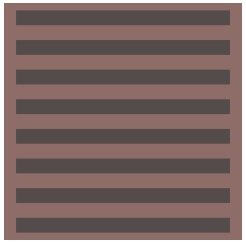


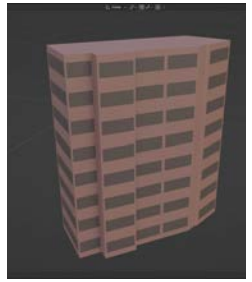
テクスチャ付きLOD1

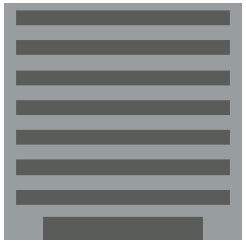
目次

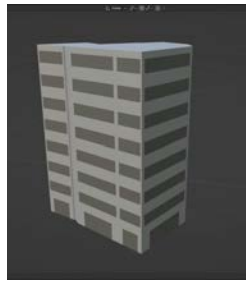
ゲームエンジンへ向けての課題

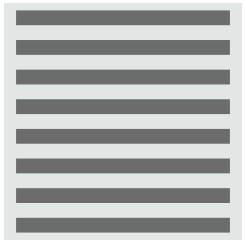
- テクスチャの枚数
  - 独立したファサードを持つ場合
    - 建物の数だけテクスチャが必要
      →1km四方だと数千単位
    - テクスチャメモリ量を圧迫





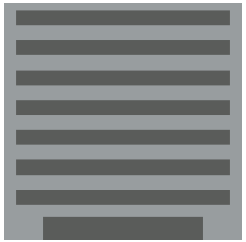


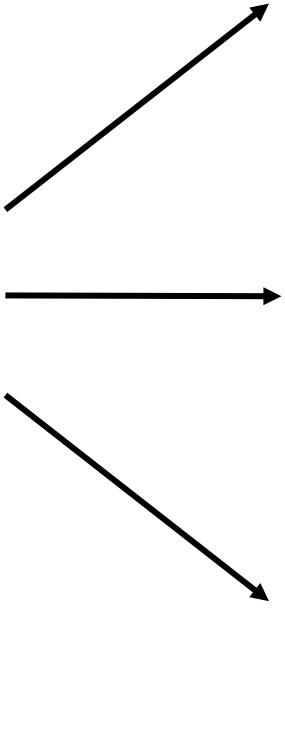




ゲームエンジンへ向けての課題（アプローチ）

- テクスチャの枚数
  - 共通化したファサードを持つ場合
    - 1つのテクスチャを形状が似た建物に割り当て
      →全体のテクスチャ数を抑制できる
    - テクスチャメモリ量を軽減





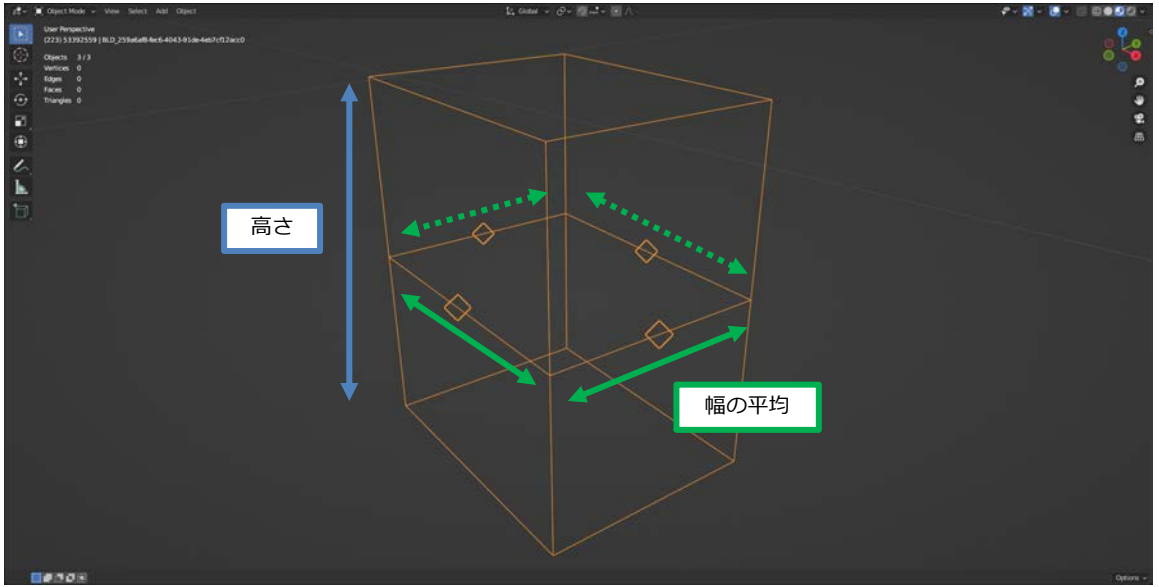
ゲームエンジンへ向けての課題（アプローチ）

Silicon Studio
Ideas × Art × Technology

- 具体的なアプローチ
  - ファサード共通化
  - テクスチャアトラス化

ファサード共通化

- 共通化に利用する形状情報
  - 高さ
  - （側面の）幅の平均

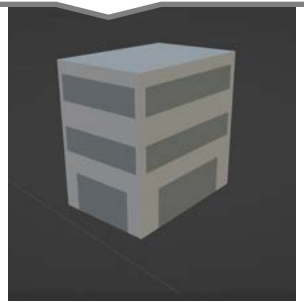


建物の情報

ファサード共通化（高さ）

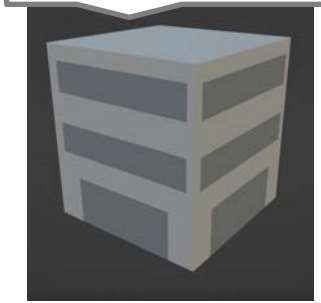
- 高さ
  - フロアの数ごとにグループ化
    - フロアの数 = 高さ/3m（小数点以下切り捨て&1未満切り上げ）

建物の情報
- 高さ：9.26m



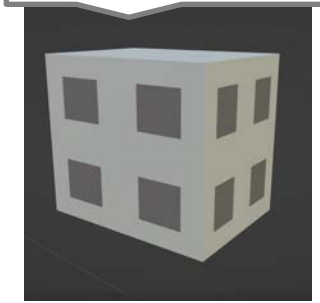
フロアの数：3

建物の情報
- 高さ：9.16m



フロアの数：3

建物の情報
- 高さ：7.16m



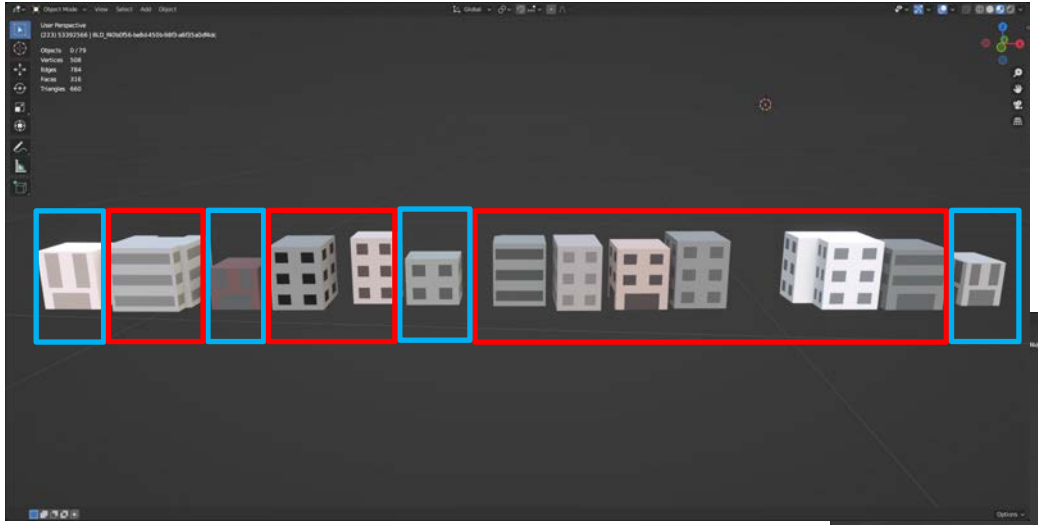
フロアの数：2

ファサード共通化手順（高さ）

共通化なし

共通化あり（高さのみ）

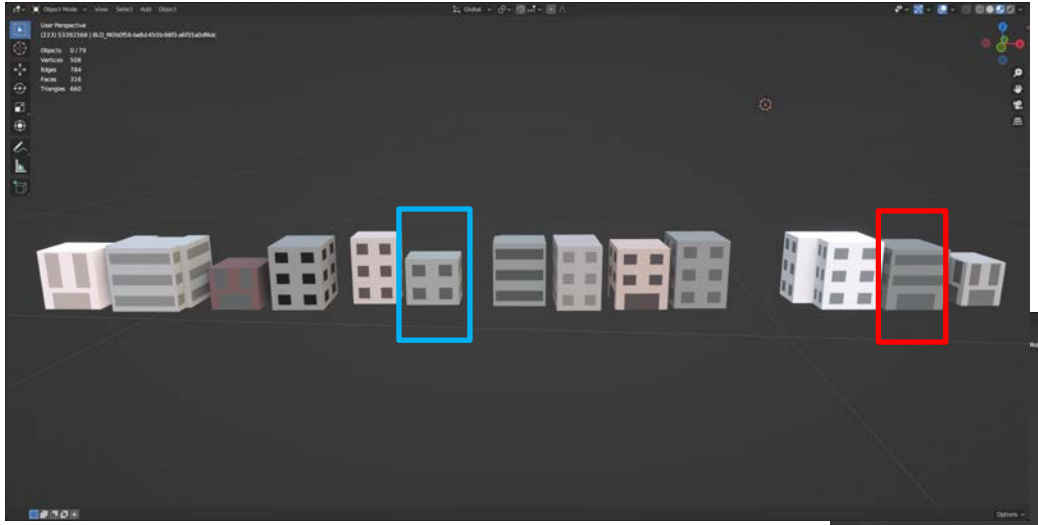
ファサード共通化手順（高さ）



共通化なし

共通化あり（高さのみ）

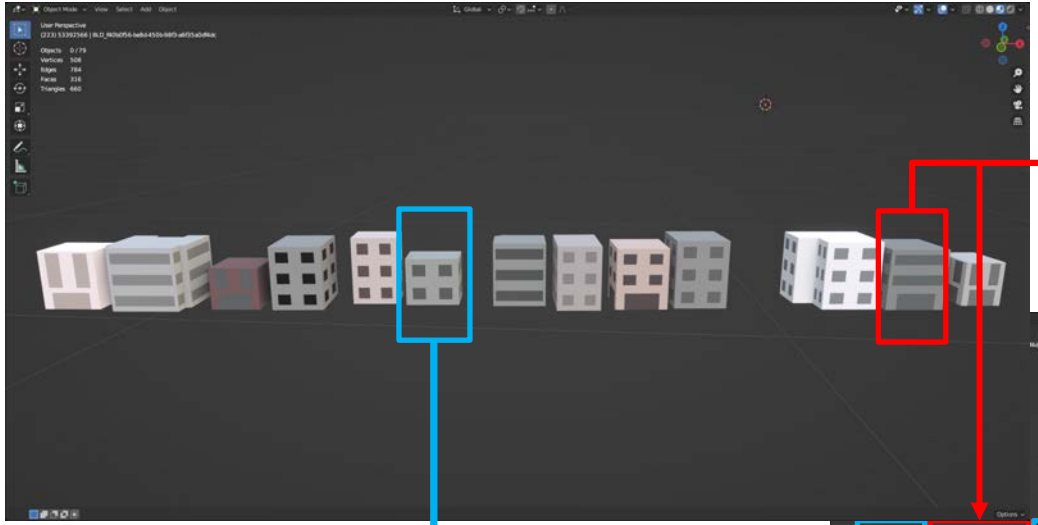
ファサード共通化手順（高さ）



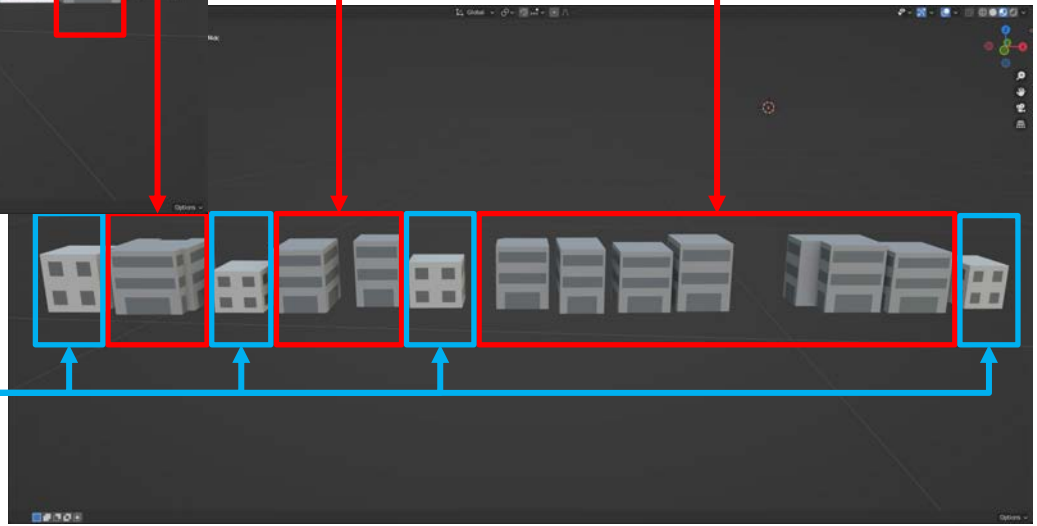
共通化なし

共通化あり（高さのみ）

ファサード共通化手順（高さ）



共通化なし



共通化あり（高さのみ）

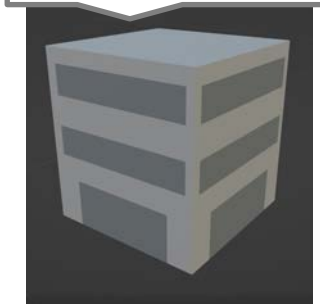
ファサード共通化（高さ+幅の平均）

- 幅の平均
  - （1フロアあたりの）窓の数ごとにグループ化
    - 窓の数 = 幅の平均/3m（小数点以下切り捨て&1未満切り上げ）

建物の情報
- 高さ：9.26m
- 幅の平均：7.80m

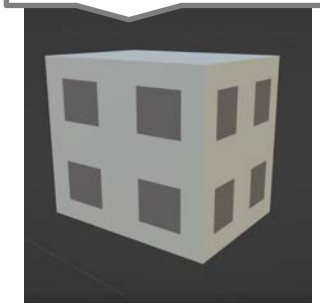
フロアの数：3
窓の数：2

建物の情報
- 高さ：9.16m
- 幅の平均：5.69m



フロアの数：3
窓の数：1

建物の情報
- 高さ：7.16m
- 幅の平均：6.93m



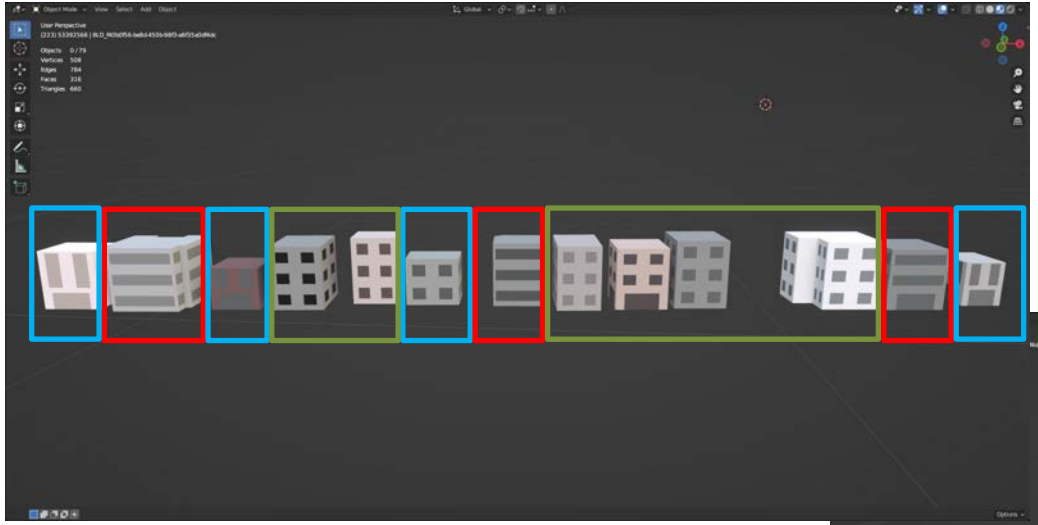
フロアの数：2
窓の数：2

ファサード共通化手順（高さ+幅の平均）

共通化なし

共通化あり（高さ+幅の平均）

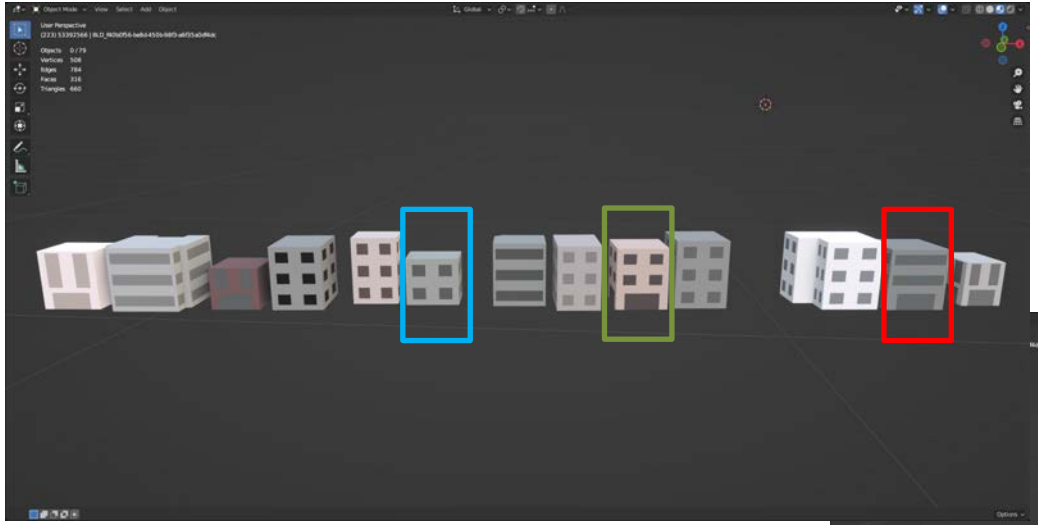
ファサード共通化手順（高さ+幅の平均）



共通化なし

共通化あり（高さ+幅の平均）

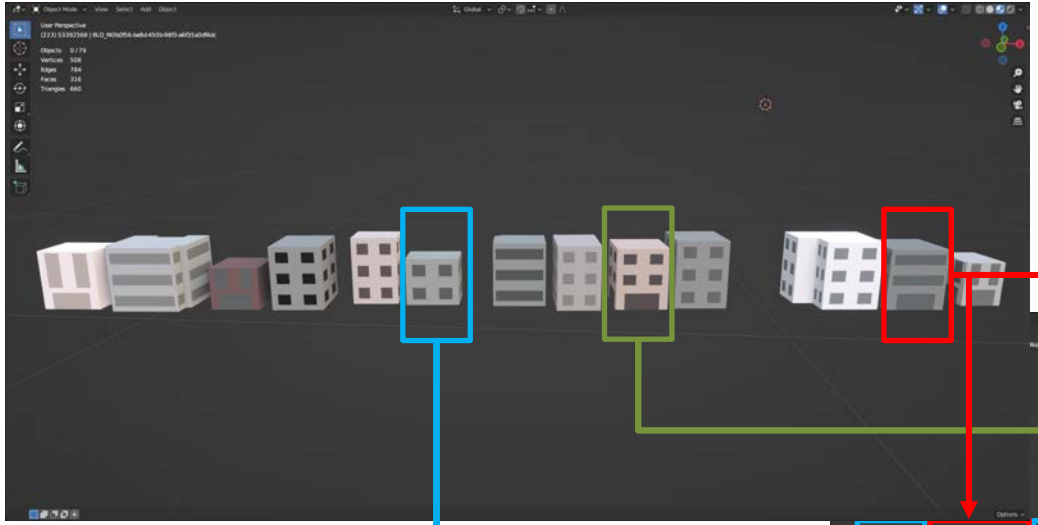
ファサード共通化手順（高さ+幅の平均）



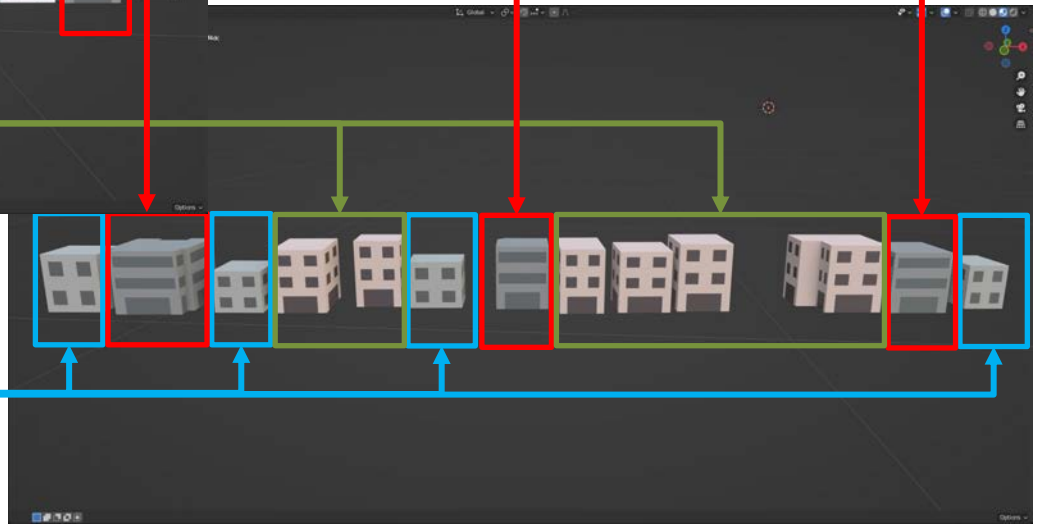
共通化なし

共通化あり（高さ+幅の平均）

ファサード共通化手順（高さ+幅の平均）



共通化なし



共通化あり（高さ+幅の平均）

ファサード共通化におけるテクスチャ数の比較

Silicon Studio
Ideas × Art × Technology

- どの程度テクスチャ数が抑制できているか
  - 対象エリア
    - 東京駅周辺3km四方
      - LOD1モデルを使用
      - 建物数：12,557棟

	共通化なし	共通化あり（高さのみ）	共通化あり（高さ+幅の平均）
テクスチャ数	12,557※	**62**	**350**

※建物数=テクスチャ数と仮定

アトラス化

Silicon Studio
Ideas × Art × Technology

- material-combiner-addon
  - Blenderのアドオン
  - 複数のマテリアルを結合＆アトラス化
  - MITライセンス
  - Code: https://github.com/Grim-es/material-combiner-addon

アトラス化

- アトラステクスチャ
  - 東京駅周辺3km四方
    - 共通化：高さ+幅の平均
    - テクスチャ数：350枚
    - 解像度：8192x8192
    - 各テクスチャ解像度：256x256
    - テクスチャ間の幅：4 pixel

Unreal Engine 5上でのパフォーマンス比較

Silicon Studio
Ideas × Art × Technology

- オリジナル（LOD1）とテクスチャ付きLOD1の比較
  - どの程度パフォーマンスに影響があるかを調査
- Unreal Engine 5(UE5)
  - バージョン：5.1.1
  - テンプレート：blank
  - 比較項目
    - ポリゴン数
    - テクスチャメモリ量
    - ドローコール（Basepass）
    - シーン全体の負荷

Unreal Engine 5上でのパフォーマンス比較

Silicon Studio
Ideas × Art × Technology

- 対象エリア
  - 東京駅周辺3km四方
    - 建物
      - 12,557棟（LOD1モデル）
      - インポート時にメッシュをマージ

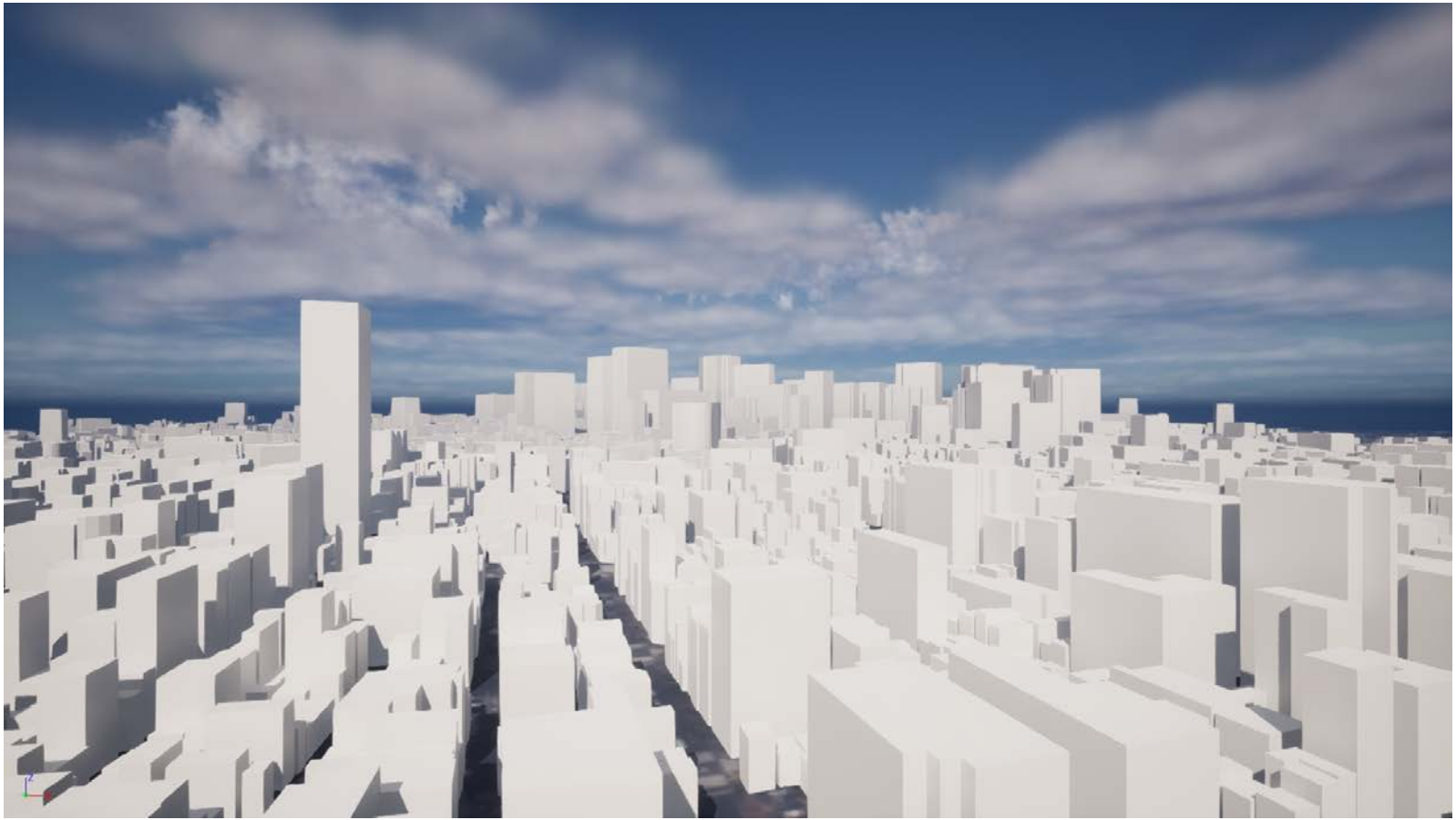
	LOD1（オリジナル）	LOD1（テクスチャ付き）
ポリゴン数	356,918	692,819
テクスチャメモリ量	0 MB	42.7 MB
ドローコール（Basepass）	1	2
シーン全体の負荷	6.98 ms	7.29 ms

※1920x1080 NVIDIA RTX 3070で計測

UE5上での比較1（オリジナル）

オリジナルに白のマテリアル付与
地形データに航空写真を貼りつけ

Silicon Studio
Ideas × Art × Technology



UE5上での比較1（テクスチャ付き）

オリジナルに白のマテリアル付与
地形データに航空写真を貼りつけ

Silicon Studio
Ideas × Art × Technology

UE5上での比較2（オリジナル）

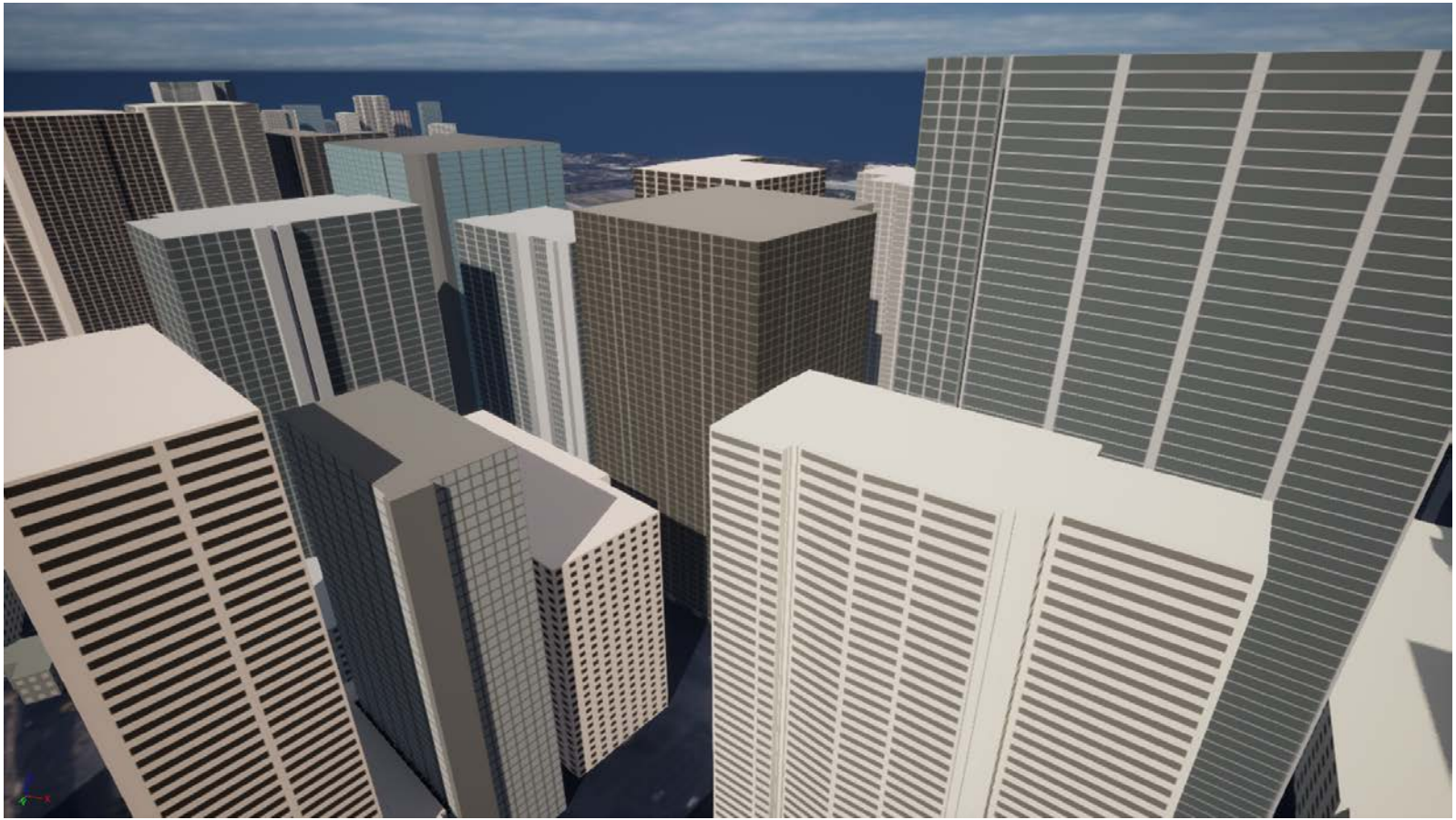
オリジナルに白のマテリアル付与
地形データに航空写真を貼りつけ

Silicon Studio
Ideas × Art × Technology

UE5上での比較2（テクスチャ付き）

オリジナルに白のマテリアル付与
地形データに航空写真を貼りつけ

Silicon Studio
Ideas × Art × Technology



目次

まとめ

- PLATEAUの簡単な紹介と課題を説明
  - 日本全国の約130都市の3D都市モデル（オープンデータ）
  - テクスチャがない建物が大多数
- Procedural PLATEAU
  - Good
    - 任意のLOD1に対して品質（位置情報）を崩さず、ファサードの自動生成
    - 軽量なテクスチャ付き3Dモデルの自動生成（＋アドオン）
  - Bad
    - 現状はビルのファサードのみ
    - テクスチャの解像度は低め（256x256固定）

今後の課題

- PLATEAUの属性情報に基づいたファサード作り
  - 属性情報をパラメータとして考える
    - その建物が住宅なのか、商業施設なのか
    - 古い建物なのか、新しい建物なのか
- LOD1のプロシージャル屋根生成（≒LOD2）
  - [Zhang 2022]などを参考に
  - 住宅などは屋根の有無が重要
- Cesium for Unrealへの対応
  - 3D地理空間上での直感的な理解に期待

[Zhang 2022] X. Zhang et al.: Procedural Roof Generation From a Single Satellite Image. Computer Graphics Forum (CGF), Eurographics, 2022

Ideas × Art × Technology

技術力・表現力・発想力を兼ね備えたCGソリューションプロバイダー