

## RDBMS による 3D TIN データベース実装手法

田中 玲吏<sup>††</sup> 杉浦 健人<sup>†</sup> 石川 佳治<sup>†</sup>

<sup>†</sup> 名古屋大学大学院情報学研究科 <sup>††</sup> 名古屋大学工学部電気電子・情報工学科

### 1 はじめに

3D モデルを柔軟に表すためのデータモデルとして、3D TIN (triangulated irregular network) データが広く用いられている。TIN データは頂点と辺の集合からなるグラフ (ネットワーク) データの一種であり、特徴として全ての閉路が三角形 (3 つの頂点及び辺) で構成されている。矩形などの単純な図形を表す際には全ての閉路を三角形とするために冗長な辺を加える必要があるが、一方でどのような複雑な物体であっても同じデータ構造で表せるという利点がある。特に、オブジェクトに対して概念的な意味を求めず、その表面のみを表したい際に有用である。

3D TIN を 3D モデルのために用いる利点として、TIN の簡単化 (低解像度化) [1, 2] が容易に行えるという点が挙げられる。矩形など他の図形を組み合わせる 3D モデルを構成する場合と異なり、TIN を構成する三角形は物体の表面を近似するものであり、概念的な意味を持たない。そのため、削除した際に影響を受ける三角形の角度の変化が少ない点を優先的に削除するなどの方針によって、TIN を構成する頂点・辺の数を削減できる。

TIN の簡単化は、特に広域の 3D モデルを効率的に可視化したい際に有用である。可視化においては LOD (level of detail) [3, 4] に基づく効率化が行われており、広い範囲を可視化する際は低解像度のデータを、狭い範囲を可視化する際は高解像度のデータを用いる。つまり、TIN データを用いる場合 LOD に基づく可視化のための低解像度なデータを容易に生成でき、可視化アプリケーションとの相性がよい。近年では写真画像から LOD を考慮した 3D TIN を生成するための OSS (open source software) として OpenMVG (multiple view geometry) [5] 及び OpenMVS (multi-view stereo) [6] などのライブラリが開発されており、今後様々な場面で 3D TIN を用いたアプリケーションの開発が予想される。

しかし、3D TIN モデルの管理は主にファイルベースで行われており、関係モデルを用いた別のデータとの結合が難しいという課題がある。3D モデルのためのファイルフォーマットはいくつか存在するが、その中でも LOD を考慮した構造を持つフォーマットとして gITF [7] が挙げられる。gITF では低解像度から高解像度までのデータを四分木などの木構造で表しており、各ファイルが子ノードへのパスや URI を持つことで階

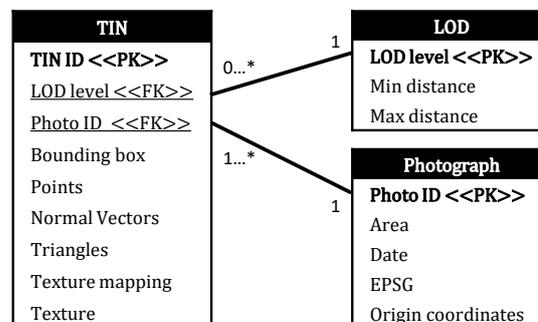


図 1 3D TIN 格納のための論理スキーマ

層構造を実現している。しかし、こうしたファイルフォーマットは可視化に焦点を当てたものであり、PostGIS [8] など、既存の GIS ツールを用いて管理された 2 次元地図の情報を直接活用できない。

そこで本稿では、3D TIN のデータベースへの格納、及びデータベースの機能を用いた 3D TIN データに対する処理の補助を検討する。特に、LOD を考慮した可視化のために実体化ビューを用いた TIN のグループ化と索引の作成を行い、データベースからの効率的な TIN の抽出を目指す。

### 2 3D TIN 格納のための論理スキーマ

3D TIN をデータベースへ格納するための論理スキーマを図 1 に示す。本研究では、可視化のための TIN データを TIN・LOD・Photograph の 3 つのテーブルで管理する。以下、各テーブルの役割をそれぞれ述べる。

TIN テーブルでは、TIN を構成するデータ自体を管理する。TIN データは 5 つの属性 (points, normal vectors, triangles, texture mapping, texture) で表現でき、それぞれ頂点群、各頂点の法線ベクトル、頂点の組合せによって構成される三角形、各頂点に対するテクスチャのマッピング情報、テクスチャ画像を表す。Points, normal vectors, triangles は TIN を構成するために必須となるが、texture mapping 及び texture は TIN 表面の描画にテクスチャ画像を用いない場合は省略できる。なお、points 中では各頂点の 3 次元座標を保持するが、TIN テーブルでは各頂点がいずれの座標系に属するかは管理せず、後述する Photograph テーブルで座標系に関する情報を管理する。また、TIN テーブルではストレージ利用の効率化のため、各撮影対象の原点座標からの相対座標で頂点位置を表す。これにより、要素数の多い頂点群を各座標 4 byte で表現でき、効率よくストレージに格納できる。最後に、bounding box 属性は各ダブルで保持される TIN データの空間的な範囲を示す 3 次元矩形を表

Implimentation of a 3D TIN database based on an RDMBS  
Reiji Tanaka<sup>††</sup>, Kento Sugiura<sup>†</sup>, and Yoshiharu Ishikawa<sup>†</sup>

<sup>†</sup> Graduate School of Informatics, Nagoya University

<sup>††</sup> Department of Information Engineering, School of Engineering, Nagoya University

し、空間索引の構築に利用する。ただし、TIN テーブルに直接空間索引を構築する場合、LOD レベルや撮影対象の混在により効率が悪化するという問題がある。この点については、後の3章で詳細を述べる。

LOD テーブルでは、可視化のための LOD に関する情報を管理する。各 LOD は描画のための最小距離 (min distance) と最大距離 (max distance) を持ち、可視化時の視点位置からの距離に応じて LOD レベルが決定する状況を想定する。つまり、可視化の際ビューワは LOD テーブルから距離に応じた各 LOD レベルの情報を読み取り、その値を選択条件とした問合せを与えると想定している。

Photograph テーブルでは、TIN データの生成に使用した写真に関する情報を管理する。Area 属性は撮影対象を表すためのテキスト情報であり、本研究では市や区などのある程度広域な範囲を想定する。また、撮影を定期的に行うことで撮影対象の差分解析などの実施も想定し、Date 属性によって撮影日に関する情報を保持する。EPSG 属性では各撮影対象が属する座標系に関する情報を保持する。なお、座標系として平面直角座標系 [9] を用いるとき県境で異なる EPSG の値を持つ場合があるが、本研究では撮影対象は一つの EPSG にのみ所属するとし、異なる EPSG に属するオブジェクトの管理は考慮しない。また、前述したとおり本研究では TIN の各頂点を相対座標によって表すため、各撮影対象毎に基準となる原点座標を origin coordinates 属性として保持する。

### 3 実体化ビューによる TIN の分割管理

前章で述べた TIN 管理では、全ての LOD レベル・撮影対象の TIN が同一テーブルで保持されており、効率的な TIN の取り出しが難しい。LOD を用いた可視化における特徴の一つとして、一般に低解像度の LOD レベルのオブジェクトは高解像度なものよりも広域の情報を表すことが挙げられる。つまり、最も低解像度な TIN データと高解像度なものとで、各テーブルで保持されるオブジェクトの粒度が全く異なる。そのため、単に TIN テーブルの bounding box 属性に対して R 木や八分木などの空間索引 [10] を作成しても、粒度の小さい高解像度なオブジェクトに対して効果的な索引が生成できないことが予想される。また、前述したとおり各 TIN は撮影対象からの相対座標で管理されるため、実際の位置と bounding box 属性で保持される位置とが異なる点も問題となる。

そこで、本稿では各 LOD レベル・撮影対象を分割するための実体化ビューを定義し、各実体化ビューで空間索引を構築する。以下に、実体化ビューを定義するための問合せ例を示す。

```
CREATE MATERIALIZED VIEW <view_name> AS
SELECT tin_id,
       ST_SetSRID(ST_Translate(
         bounding_box, origin_x, origin_y), epsg),
       points, normal_vectors, triangles,
       texture_mapping, texture
FROM tin INNER JOIN
       photograph USING (photo_id)
```

```
WHERE photo_id = <photo_id>
AND lod_level = <lod_level>;
```

このような実体化ビューを各撮影対象・LOD レベル毎に生成することで、各ビューでは同一の座標軸・LOD レベルで統一され、バウンディングボックスに対して効果的に索引を構築できる。なお、ST\_SetSRID 及び ST\_Translate は PostGIS [8] における空間処理のための関数であり、それぞれ座標空間の設定と平行移動を行う。

### 4 おわりに

本稿では、3D TIN データのデータベースへの格納について検討した。3D TIN を格納するための論理スキーマについて提案し、実体化ビューを用いた効果的な索引の構築について議論した。

今後の課題としては、試作の実装及び LOD レベルを考慮した空間索引の検討が挙げられる。試作には PostgreSQL/PostGIS を使用し、実体化ビューによる分割を行う場合と行わない場合との性能評価を行う予定である。また、実体化ビューを利用した実装は効率的な反面、冗長なストレージの使用が多く発生する。そのため、TIN テーブルの bounding box 属性に直接構築可能な空間索引を議論し、冗長なストレージの使用を避けつつ効率的な TIN 抽出の実現について検討する。

### 謝辞

本研究は、JSPS 科研費 (JP16H01722, JP19K21530) の助成、および、国立研究開発法人新エネルギー・産業技術総合開発機構 (NEDO) の委託業務による。

### 参考文献

- [1] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen, "Decimation of triangle meshes," in *Proc. Annual Conference on Computer Graphics and Interactive Techniques*, pp. 65–70, 1992.
- [2] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *Proc. Annual Conference on Computer Graphics and Interactive Techniques*, pp. 209–216, 1997.
- [3] J. H. Clark, "Hierarchical geometric models for visible surface algorithms," *Commun. ACM*, vol. 19, no. 10, pp. 547–554, 1976.
- [4] F. K. Musgrave, C. E. Kolb, and R. S. Mace, "The synthesis and rendering of eroded fractal terrains," *SIGGRAPH Comput. Graph.*, vol. 23, no. 3, pp. 41–50, 1989.
- [5] openMVG: "open Multiple View Geometry": <http://imagine.enpc.fr/~moulonp/openMVG/> (accessed Jan 10, 2020.).
- [6] OpenMVS by cdcseacave: <http://cdcseacave.github.io/openMVS/> (accessed Jan 10, 2020.).
- [7] glTF Overview - The Khronos Group Inc: <https://www.khronos.org/glTF/> (accessed Jan 10, 2020.).
- [8] PostGIS — Spatial and Geographic Objects for PostgreSQL: <https://postgis.net/> (accessed Jan 10, 2020.).
- [9] 平面直角座標系 (平成十四年国土交通省告示第九号) | 国土地理院: <https://www.gsi.go.jp/LAW/heimencho.html> (accessed Jan 10, 2020.).
- [10] P. Rigaux, M. Scholl, and A. Voisard, *Spatial Databases with Application to GIS*. Morgan Kaufmann Publishers Inc., 2001.