予備審査の質問に対する回答と修正

M2 田川幸汰

1 予備審査の質問

予備審査では，本研究の手法および位置づけに関して，主に以下の点について質問が挙げられた。本章では，それぞれの質問に対する回答を示すとともに，それを踏まえて研究内容に加えた修正点について説明する。

1. 自己位置の補完には，どのようなセンサ情報を用いているのか
2. 画像情報を用いて自己位置推定を行う点で Visual SLAM と類似しているが、本研究手法はどのような点で優れているのか
3. どういう意味で簡易なモデルなのか
4. 簡易モデルにおけるワイヤーフレームを自動生成することは可能か

1.1 自己位置推定の補完に用いるセンサ情報

本研究では、自己位置推定の安定化および精度向上のために、Apple の ARKit が提供する Visual-Inertial Odometry（VIO）を活用している。ARKit の VIO は、加速度計およびジャイロスコープから得られる情報と、カメラ画像から抽出された特徴点のトラッキング情報を統合することで、デバイスの位置および姿勢を推定する手法である。このアルゴリズムは、モーションセンサー情報とシーン画像の解析結果を組み合わせることで、初期位置から相対的な自己位置推定を実現している [1]。

また、本研究で使用する iPad Pro（第 5 世代）には LiDAR スキャナが搭載されており、ARKit は LiDAR スキャナによる深度情報を利用した機能を提供している。LiDAR によって取得される深度情報は、周囲環境の三次元形状の把握や環境理解に利用され、ARKit 内部においてトラッキングの安定化に寄与するとされている [2]。

1.2 Visual SLAM との関係および比較方針

まず、Visual SLAM について簡単に説明する。Visual SLAM は、カメラ画像を用いて自己位置推定を行うと同時に、周辺環境の地図を逐次生成する手法である。未知環境においても自己位置を推定できる点が特徴であり、ロボティクスや AR 分野を中心に広く用いられている [3]。

次に、本研究の手法と Visual SLAM との違いについて説明する。本研究が想定する状況では、あらかじめ簡易的な三次元モデルを構築可能な環境情報が利用できる。このように、すでに環境情報がある程度わかっているかという点で、両者の前提条件は大きく異なる。

Visual SLAM は一般に計算負荷が高く、高精度な地図生成や最適化処理を伴う場合には、モバイル端末上での安定した実行が難しいことがある。そのため，本研究が想定する屋内環境における道案内用途において、幅広いユーザーを対象とした運用を考慮すると、Visual SLAM はオーバースペックとなる可能性が高い。

さらに、自己位置推定の精度を指標とした比較についても検討したが、道案内用途においては必要十分な精度が確保されていれば目的は達成される。そのため、数値的な推定精度のみを用いた直接的な比較は、本研究の目的に対して必ずしも適切な評価方法ではないと判断した。

1.3 簡易モデルについて

本研究において用いる簡易的な 3 次元モデルとは、レーザースキャナによって生成される高密度な点群メッシュモデルや、フォトグラメトリによって生成される、精密なメッシュモデルを指すものではない。代わりに本研究では、建物の床や壁といった空間構造の骨組みを表す幾何形状と、その表面に付与されたテクスチャ情報のみから構成される、軽量で管理しやすい 3 次元モデルを想定している。

このようなモデルは，複雑な形状を高精度に再現することよりも，床や壁、通路といった空間の基本的な構成を簡潔に表現することに重点を置き、屋内の道案内に必要となる自己位置推定を、必要十分な精度で実現できる点に特徴がある (図 1)。

点群　　フォトグラメトリ　　本研究

図 1: 異なる三次元モデル表現の比較

1.4 簡易モデルにおけるワイヤーフレームの自動生成

ワイヤーフレームを自動生成するために、本研究ではフロアマップ画像（もしくはあらかじめ計測済みの 3 次元特徴点ファイル）とカメラの撮影位置情報を入力とし、床および壁の境界を構成する 3 次元頂点群と面情報を生成するプログラムを作成した。本節では，その処理の流れを説明する。

まず、フロアマップ画像を読み込み、床領域の境界を構成するコーナー点を取得する。コーナー点は，あらかじめ用意された特徴点ファイルから読み込むか，OpenCV を用いた簡易的な GUI を通じて手動で指定する。手動指定の場合には、フロアマップ上の 2 次元座標と床面上の 3 次元座標（$Z = 0$ 平面）との対応点を入力し、これらから 2 次元座標から 3 次元座標への変換行列を推定する。

次に、取得した床境界のコーナー点間をサンプリングし、床境界を構成する頂点群を生成する。この際、カメラ位置と床境界との距離関係を考慮し、サンプリング点間の間隔が過度に大きくならないように調整を行うことで、十分な密度の境界点を確保する。また、生成された床境界点に対して、一定の高さを付与した対応する天井境界点を作成する。

床面については、床境界点を用いて制約付きドロネー三角形分割を行い、床領域内部を複数の三角形面に分割する。また、床境界点と天井境界点の対応関係から，各辺に対して四角形の壁面を生成することで、床と壁から構成される簡易的なワイヤーフレーム構造を形成する。

最後に、生成した頂点情報および面情報をもとに、Metasequoia 形式（MQO）の 3 次元モデルとして出力する。このようにして出力された、テクスチャ情報付与前の 3 次元モデルを図 2 に示す。

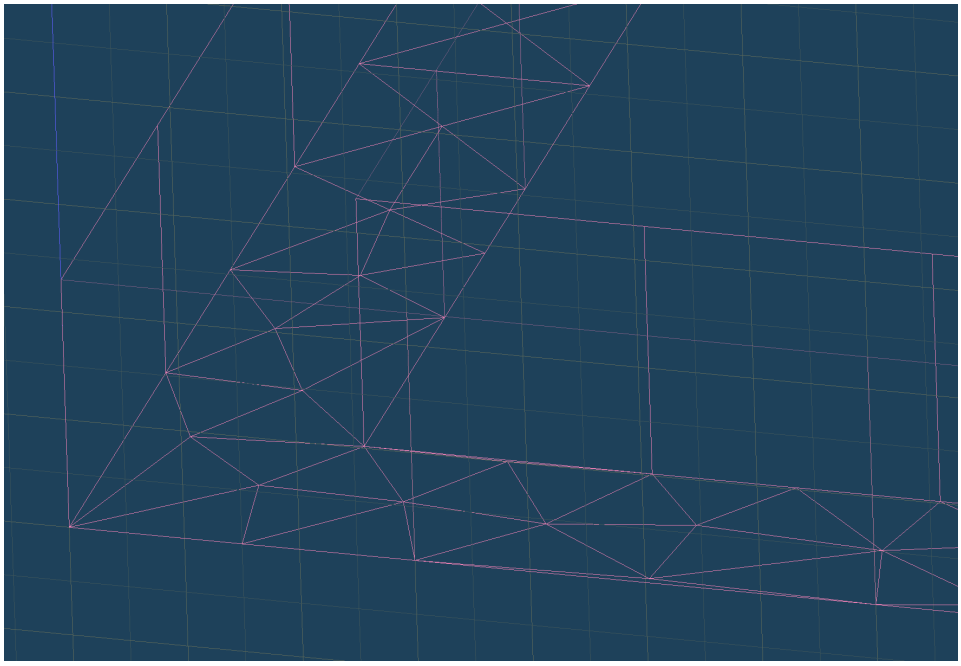

図 2: ワイヤーフレームモデル

参考文献

[1] A. Inc, “Managing session life cycle and tracking quality.” https://developer.apple.com/jp/documentation/arkit/managing_session_lifecycle_and_tracking_quality/, 2026. Accessed: 2026-01-16.

[2] A. Documentation, “More to explore with arkit 6, ”depth api”.” https://developer.apple.com/augmented-reality/arkit/, 2026. Accessed: 2026-01-16.

[3] T. Taketomi, H. Uchiyama, and S. Ikeda, “Visual slam algorithms: a survey from 2010 to 2016,” *IPSJ Transactions on Computer Vision and Applications*, pp. 1–15, 2017.