Vol.2012-CVIM-183 No.10
2012/9/2

携帯情報端末に適した効率の良い自己位置推定および地図生成アルゴリズム

上田 雄大[1,a)]　新田 直也[1,b)]

概要：近年, スマートフォンなどの携帯情報端末の高機能化, 高性能化に伴い, モバイル環境においても拡張現実感 (AR) を実現することが可能になってきた. とりわけマーカを用いない AR(マーカレス AR) は, カメラ以外にはセンサもマーカも必要としないためその実現が広く期待されている技術であるが, 今のところスマートフォン上では十分な精度と実行速度を同時に達成することが困難な状況にある. 我々研究グループは単眼 SLAM アルゴリズムの高速化を図ることによって, 携帯情報端末上で動作するマーカレス AR システムの実現を目指している. 本稿では, 先行研究で提案した効率の良い単眼 SLAM アルゴリズムの実行速度を維持しつつロバスト性を向上させる改良を行った. その結果, 実環境に近づくように計算機上でランダムノイズを混入させた仮想環境でも十分な推定精度を得れるようアルゴリズムを改良することに成功した. 今後, 実際の実環境においても実用的な精度および実行速度を得られるようアルゴリズムの改善を図る予定である.

1. はじめに

近年, スマートフォンなどの携帯情報端末の高機能化, 高性能化に伴い, モバイル環境においても拡張現実感 (Augmented Reality, AR) を実現することが可能になってきた. 一般に AR においては, 仮想物体をそれがあたかも実在しているかのように現実環境に重ね合わせて表示する必要があるが, 重ね合わせ位置を決定する上で GPS やジャイロセンサなどのセンサを用いる方法と, カメラから得られた動画像を解析して必要な情報を得る方法がある. 動画像に基づく AR はさらにマーカを使用するものと使用しないものに分けられる. センサに基づく AR や, 動画像に基づく AR でもマーカを使用するものについては, スマートフォン上でもすでに実現され広く使われつつある. しかしながら, 一般にマーカを用いない AR(マーカレス AR) は, その実現に比較的高い計算コストを要するため, 今のところスマートフォン上で十分な精度と実行速度を同時に達成することは困難な状況にある.

マーカレス AR では, カメラから得られた動画像に含まれる自然特徴点を追跡することによって, カメラの自己位置・姿勢を実時間で推定し, マーカを使うことなく重ね合わせ位置を決めることができる. これは, ロボット工学分野で研究されてきた単眼 SLAM 問題 (Monocular Simultaneous Localization and Mapping Problem) を解くことに相当する. 単眼 SLAM 問題とは, 単眼カメラを搭載したロボットや移動体などが, カメラから得た情報を元に周囲の未知の環境の地図を生成しながら, 同時に生成された地図内での自己位置・姿勢の推定を行う問題である. ただしロボットとは違い AR では, カメラが人の手によってより自由かつ高速に動かされるため, 一般にロボットにおける SLAM より高速かつロバストな自己位置・姿勢推定が要求される.

マーカレス AR システムとしては PTAM(Parallel Tracking and Mapping) が代表的である. PTAM は, 単眼 SLAM における地図生成処理を自己位置推定処理から分離し, 実時間性は要求されないが計算コストが高い地図生成処理を自己位置推定処理とは別の CPU コアで並列実行させることによって, ロバストかつ高速なマーカレス AR の実現を可能にしている. また, PTAM をスマートフォンでも動作可能なように最適化したバージョンも存在する. しかしながら, PTAM では地図生成処理として, 画像から抽出される特徴点の総数を M, フレーム数を N としたとき時間計算量が $O(N^3)$ または $O(MN^2)$ となるバンドル調整を採用しているため, その実行が頻繁に要求されるような状況では処理が間に合わず, 十分なロバスト性を達成できなくなる傾向がある.

そこで本研究は, 携帯情報端末などの小型計算機でのマーカレス AR の実現を目指し, バンドル調整を使用しない高

1　甲南大学大学院 自然科学研究科
Graduate School of Natural Science, Konan University
a)　mn124001@center.konan-u.ac.jp
b)　n-nitta@konan-u.ac.jp

速かつ十分な精度を有したマーカレス AR アルゴリズムの開発を行っている．先行研究では, 地図生成処理と自己位置推定処理の両方の時間計算量が $O(M)$ であるようなアルゴリズムを考案し, 仮想環境および実環境における精度について評価実験を行った．その結果, 計算機上に構築された仮想環境で十分な推定精度を得ることに成功したが, 実環境では実用的な精度を得ることができなかった．本稿では, 提案アルゴリズムの実行速度を維持しつつロバスト性を向上させる改良を行い, 観測データにランダムノイズを混入させたより実環境に近い仮想環境でも, 十分な推定精度を達成することに成功した．

2. 基本アルゴリズム

提案アルゴリズムは, 取得した動画像のフレームを $(\psi_0, \psi_1, \dots, \psi_n, \dots)$ としたとき, 隣り合う各フレームのペア ψ_{n-1}, ψ_n に対して以下の処理を行って, カメラの自己位置・姿勢および地図生成を行う．

(1) カメラより取得した現在のフレームの画像 (ψ_n) から自然特徴点を取得する．ただし, 提案アルゴリズムは具体的な抽出方法とは独立である．ここで, いずれかのフレームで抽出された特徴点の総数を M 個とし, i 番目 $(0 \leq i \leq M)$ の特徴点が ψ_n で抽出されたか否かを可視性指標 I_i^n で表す．ただし, 第 i 特徴点が ψ_n で抽出されたとき $I_i^n = 1$, されなかったとき $I_i^n = 0$ とする．$I_i^n = 1$ であるような自然特徴点の ψ_n の画像平面上での座標を (u_i^n, v_i^n) とする．

(2) 前回のフレームの画像 (ψ_{n-1}) と現在のフレームの画像 (ψ_n) から自然特徴点の移動を追跡し, その移動量であるオプティカルフローを求める．ここで, 第 i 特徴点について $I_i^{n-1} = I_i^n = 1$ であるとし, その特徴点のオプティカルフローを $(u_i^n, v_i^n, \Delta u_i^n, \Delta v_i^n)$ で表す．ただし, $\Delta u_i^n = u_i^n - u_i^{n-1}$, $\Delta v_i^n = v_i^n - v_i^{n-1}$ である．

(3) 上記 (2) で求めたオプティカルフローと現在までのフレーム $(\psi_1, \dots, \psi_{n-1})$ で推定された各自然特徴点の奥行き情報からカメラ運動の回転移動成分の推定を行う (詳細は 2.1 節で説明).

(4) 上記 (3) で求めたカメラ運動の回転移動成分と現在までのフレームで推定された自然特徴点の 3 次元位置情報からカメラ運動の並行移動成分を推定し, カメラの自己位置及び姿勢推定を行う (詳細は 2.2 節で説明).

(5) 現在までのフレームおよび現在のフレームで推定されたカメラの自己位置及び姿勢と各自然特徴点の抽出座標から, 特徴点の 3 次元位置情報を再計算し, 地図の拡張および精製を行う (詳細は 2.3 節で説明).

上記処理をフレーム毎にリアルタイムで実行する．上記処理を繰り返すことで各自然特徴点の奥行き情報及びカメラの運動パラメータの推定精度は反復的に向上していくと考えられる．現実環境の奥行き情報は事前に用意せず, 初回フレームのカメラ運動パラメータの推定に用いる特徴点の奥行き情報については, 一定の値を与える．本稿では画像からの自然特徴点の抽出には Lucas-Kanade 法 [4] を用いた．

2.1 奥行きが既知であると仮定した場合のカメラ運動推定

最初に各自然特徴点の奥行きが既知であるという仮定の下に, それらの特徴点の画像平面上での動きから, カメラ運動の並進成分及び回転成分を推定し, カメラの自己位置及び姿勢を求める方法について考える．以下では, ψ_n におけるカメラ中心座標系を $\mathcal{C}_n$ とおき, 第 i 特徴点の $\mathcal{C}_n$ 上での位置を $\mathbf{p}_{i\mathcal{C}_n} = (x_i, y_i, z_i)$ で表す．なお本節では簡単のため, 座標系を $\mathcal{C}_n$ に固定して考える．また文脈から明らかな場合は, u_i^n, v_i^n, Δu_i^n, Δv_i^n をそれぞれ u_i, v_i, Δu_i, Δv_i と略す．なお, 画像平面上の u 軸と v 軸は常に $\mathcal{C}_n$ の x 軸および y 軸とそれぞれ平行となり, 自然特徴点の奥行き方向が $\mathcal{C}_n$ の z 軸と一致することに注意すること．なお, 以下ではカメラモデルとしてピンホールモデルを採用する．カメラの u 方向および v 方向の焦点距離 f_u, f_v は既知であると仮定し, 画像歪はないとする．また簡単のため, 光軸点を $(0, 0)$ とする．このとき, 透視投影変換は以下のように表される．

$$\begin{pmatrix} u_i \\ v_i \end{pmatrix} = \begin{pmatrix} f_u & 0 \\ 0 & f_v \end{pmatrix} \begin{pmatrix} \frac{x_i}{z_i} \\ \frac{y_i}{z_i} \end{pmatrix} \tag{1}$$

カメラが微小並進運動および微小回転運動した場合の画像平面上での特徴点の見かけの動きを図 1 および図 2 に示す．

並進運動のみの場合:
カメラが x-z 平面上で $(-\Delta t_x, \Delta t_z)$ だけ並進運動して, 図 1 のように, $I_i^{n-1} = I_i^n = 1$ を満たす第 i 特徴点が画像平面上で u 軸方向に u_i から Δu_i^{T} 移動したことが観測されたとすると, 図より以下の式が成り立つ．ただしここでは, 第 i 特徴点の奥行き z_i は既知であるとする．

$$\frac{z_i}{f_u} \cdot \Delta u_i^{\mathrm{T}} = -\Delta t_x + \Delta t_z \cdot \frac{u_i + \Delta u_i^{\mathrm{T}}}{f_u}. \tag{2}$$

$\frac{\Delta u_i^{\mathrm{T}}}{f_u}$ はカメラのフレームレートが高いと仮定すると 0 に近いとみなせるので, この部分を省略して, 並進運動は

$$z_i \cdot \Delta u_i^{\mathrm{T}} \approx -\Delta t_x \cdot f_u + \Delta t_z \cdot u_i \tag{3}$$

で近似することができる．

回転運動のみの場合:
カメラが y 軸を中心に $-\Delta\omega_y$ だけ回転運動して, 図 2 のように, $I_i^{n-1} = I_i^n = 1$ を満たす第 i 特徴点が画像平面上で u 軸方向に u_i から Δu_i^{R} 移動したことが観測されたとすると,

$$\Delta u_i^{\mathrm{R}} = -\sin\Delta\omega_y \cdot \sqrt{f_u^2 + u_i^2} \cdot \frac{\sqrt{f_u^2 + (u_i + \Delta u_i^{\mathrm{R}})^2}}{f_u}. \tag{4}$$

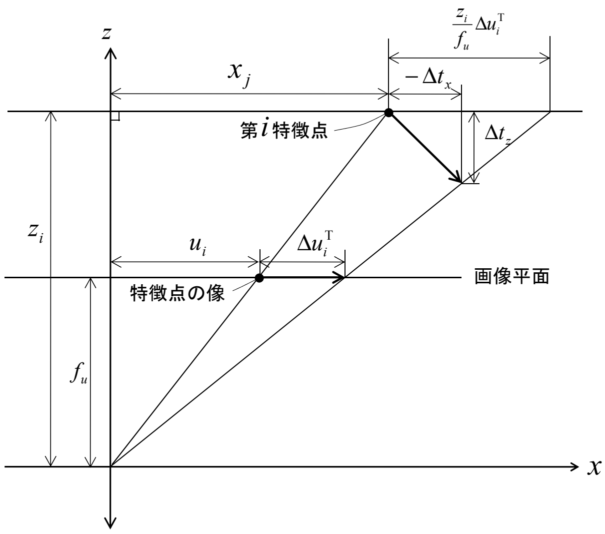


図 1　並進運動の場合のオプティカルフロー

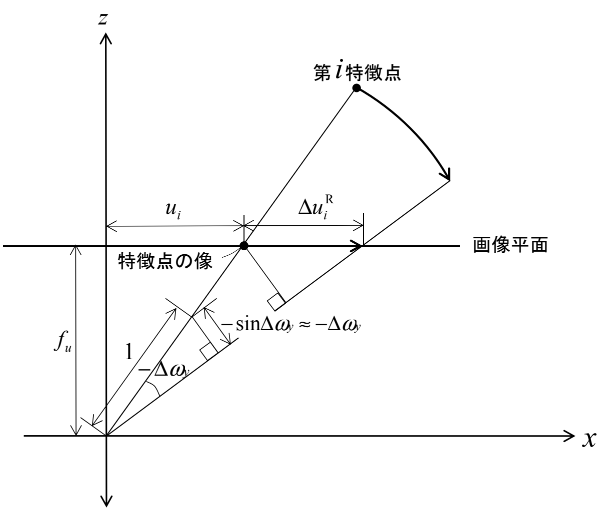


図 2　回転運動の場合のオプティカルフロー

また, カメラのフレームレートが高いと仮定すると $\Delta\omega_y$ が 0 に近いとみなすことができるので $\sin\Delta\omega_y \approx \Delta\omega_y$ とみなせる. 同様に, $(u_i + \Delta u_i^{\mathrm{R}}) \approx u_i$ とみなせることから, 回転運動は

$$\Delta u_i^{\mathrm{R}} \approx -\Delta\omega_y \cdot \frac{f_u^2 + u_i^2}{f_u} = -\Delta\omega_y \cdot f_u - \frac{\Delta\omega_y}{f_u} \cdot u_i^2 \quad (5)$$

で近似することができる.

一般の運動の場合:
一般の運動の場合における $I_i^{n-1} = I_i^n = 1$ を満たす第 i 特徴点の画像平面上での u 軸方向の移動量を Δu_i とすると,

$$\Delta u_i \approx \Delta u_i^{\mathrm{T}} + \Delta u_i^{\mathrm{R}} \quad (6)$$

となる. よって, 式 (3), (5) より

$$\begin{aligned}\Delta u_i &\approx \Delta u_i^{\mathrm{T}} + \Delta u_i^{\mathrm{R}} \\ &\approx -\Delta\omega_y \cdot f_u - \frac{\Delta t_x \cdot f_u}{z_i} + \frac{\Delta t_z}{z_i} \cdot u_i - \frac{\Delta\omega_y}{f_u} \cdot u_i^2 \\ &= f_0(u_i)\end{aligned} \quad (7)$$

が成り立つ.

(Step1) まず式 (7) の右辺を引数 u_i に関する 2 次の関数 f_0 とみなし, 各項の係数を重み付き最小二乗法を用いて求める. ここで 2 次の係数から $\Delta\omega_y$ を推定し, その値を $\hat{\Delta\omega}_y^{\,0}$ とおく.

(Step2) 次に式 (7) の Δt_x および Δt_z をそれぞれ $\hat{\Delta t}_x^{\,k+1}$, $\hat{\Delta t}_z^{\,k+1}$ に置き換え, $\Delta\omega_y$ の 2 箇所の出現をそれぞれ $\hat{\Delta\omega}_y^{\,k}$ と $\hat{\Delta\omega}_y^{\,k+1}(k \geq 0)$ に置き換えて, 以下のように変形する.

$$\begin{aligned}&z_i(\Delta u_i + \frac{\hat{\Delta\omega}_y^{\,k}}{f_u} \cdot u_i^2) \\ &\approx -\hat{\Delta t}_x^{\,k+1} \cdot f_u + \hat{\Delta t}_z^{\,k+1} \cdot u_i - \hat{\Delta\omega}_y^{\,k+1} \cdot f_u \cdot z_i \\ &= f_1(u_i, z_i).\end{aligned} \quad (8)$$

このとき, 式 (8) は $(u_i,\ z_i,\ z_i(\Delta u_i + \frac{\hat{\Delta\omega}_y^{\,k}}{f_u} \cdot u_i^2))$ の 3 変数による平面の方程式とみなすことができる. そこで, 3 次元の重み付き最小二乗法を用いて, 平面を推定しカメラの運動パラメータ $(\hat{\Delta t}_x^{\,k+1}, \hat{\Delta t}_z^{\,k+1}, \hat{\Delta\omega}_y^{\,k+1})$ を求める. この計算を K_1 回繰り返して Δt_x, Δt_z および $\Delta\omega_y$ の推定精度の向上を図る.

(Step3) 最後に, 式 (2) および (4) について再度考察する. ここでは, これらの式から式 (3) および (5) を得る際に無視した $\frac{\Delta u_i^{\mathrm{T}}}{f_u}$ および $\frac{\Delta u_i^{\mathrm{R}}}{f_u}$ を考慮に入れ, 以下のように式を詳細化する.

$$\Delta u_i{}^{\mathrm{T}} \approx -\hat{\Delta t}_x^{\,k+1} \cdot \frac{f_u}{z_i} + \hat{\Delta t}_z^{\,k+1} \cdot \frac{u_i + \widetilde{\Delta u_i{}^{\mathrm{T}}}}{z_i}, \quad (9)$$

ただし,

$$\widetilde{\Delta u_i{}^{\mathrm{T}}} = -\hat{\Delta t}_x^{\,k} \cdot \frac{f_u}{z_i} + \hat{\Delta t}_z^{\,k} \cdot \frac{u_i}{z_i} \quad (10)$$

とする. また,

$$\Delta u_i{}^{\mathrm{R}} \approx -\hat{\Delta\omega}_y^{\,k+1} \cdot \sqrt{f_u^2 + u_i^2} \cdot \frac{\sqrt{f_u^2 + (u_i + \widetilde{\Delta u_i{}^{\mathrm{R}}})^2}}{f_u}, \quad (11)$$

ただし,

$$\widetilde{\Delta u_i{}^{\mathrm{R}}} = -\hat{\Delta\omega}_y^{\,k} \cdot f_u - \frac{\hat{\Delta\omega}_y^{\,k}}{f_u} \cdot u_i^2 \quad (12)$$

とする. ここで, 式 (10) および (12) は, 式 (3) および (5) にそれぞれ基づいていることに注意されたい. これらの式より以下の式を得ることができる.

$$z_i \cdot \Delta u_i$$
$$\approx z_i \cdot \Delta {u_i}^{\mathrm{T}} + z_i \cdot \Delta {u_i}^{\mathrm{R}}$$
$$= f_2(u_i + \widetilde{\Delta u_i}^{\mathrm{T}}, z_i \cdot \sqrt{f_u^2 + u_i^2} \cdot \frac{\sqrt{f_u^2 + (u_i + \widetilde{\Delta u_i}^{\mathrm{R}})^2}}{f_u}). \tag{13}$$

ここで式 (13) は, 3 変数 ($z_i \cdot \Delta u_i$, $u_i + \widetilde{\Delta u_i}^{\mathrm{T}}$, $z_i \cdot \sqrt{f_u^2 + u_i^2} \cdot \frac{\sqrt{f_u^2 + (u_i + \widetilde{\Delta u_i}^{\mathrm{R}})^2}}{f_u}$) の間の平面の方程式とみなすことができる．したがって, 3 次元の重み付き最小二乗法を用いることによって, 事前に求められた $\hat{\Delta t_x}^k$, $\hat{\Delta t_z}^k$ および $\hat{\Delta \omega_y}^k$ の値から, $\hat{\Delta t_x}^{k+1}$, $\hat{\Delta t_z}^{k+1}$ および $\hat{\Delta \omega_y}^{k+1}$ の値を繰り返し推定することができる．(Step2) の後にこの計算を K_2 回繰り返して Δt_x, Δt_z および $\Delta \omega_y$ の推定精度の向上を図る．最終的に $\hat{\Delta t_x}^{K_1+K_2}$, $\hat{\Delta t_z}^{K_1+K_2}$ および $\hat{\Delta \omega_y}^{K_1+K_2}$ をそれぞれ Δt_x, Δt_z, $\Delta \omega_y$ の推定値とする．また, $\hat{\Delta t_y}^{K_1+K_2}$ および $\hat{\Delta \omega_x}^{K_1+K_2}$ も同様に求める．最適な K_1 および K_2 の値は後で議論する．

2.2 カメラ位置および姿勢の更新

以下では世界座標系を $\mathcal{W}$, 第 i 特徴点の $\mathcal{W}$ における位置を $\mathbf{p}_{i\mathcal{W}}$, フレーム $\psi_n (n \geq 1)$ 時点でのカメラの $\mathcal{W}$ 上の位置を $\mathbf{t}^n_{\mathcal{W}}$, 同時点でのカメラ向きを表す $\mathcal{W}$ 上の回転行列を $\mathbf{R}^n_{\mathcal{W}}$ とおく．また, $\hat{\mathbf{p}^n_{i\mathcal{W}}}$, $\hat{\mathbf{t}^n_{\mathcal{W}}}$ および $\hat{\mathbf{R}^n_{\mathcal{W}}}$ を, それぞれ ψ_n フレームにおける $\mathbf{p}_{i\mathcal{W}}$, $\mathbf{t}^n_{\mathcal{W}}$, $\mathbf{R}^n_{\mathcal{W}}$ の推定値とする．このとき ψ_n フレームにおけるカメラ位置および姿勢の更新は, $\hat{\mathbf{t}^n_{\mathcal{W}}}$, $\hat{\mathbf{R}^n_{\mathcal{W}}}$ を求めることに相当する．

$\hat{\mathbf{t}^n_{\mathcal{W}}}$ および $\hat{\mathbf{R}^n_{\mathcal{W}}}$ は, 2.1 節のアルゴリズムで推定されたカメラ運動パラメータから求めることができるが, ここでは, $\hat{\mathbf{R}^n_{\mathcal{W}}}$ のみを推定カメラ運動パラメータの回転成分 (すなわち, $\hat{\Delta \omega_y}^{K_1+K_2}$ および $\hat{\Delta \omega_x}^{K_1+K_2}$) から求め, $\hat{\mathbf{t}^n_{\mathcal{W}}}$ は別の方法で求める (以下では, **特徴点位置に基づいた自己位置の校正**と呼ぶ). その理由は, 2.1 節のアルゴリズムで推定されたカメラ運動パラメータの並進成分 (すなわち, $\hat{\Delta t_x}^{K_1+K_2}$, $\hat{\Delta t_y}^{K_1+K_2}$, $\hat{\Delta t_z}^{K_1+K_2}$) がスケール変動の影響をより強く受けるためである．そこで, 隣り合うフレーム間でスケールファクタがほぼ保存されるように $\hat{\mathbf{t}^n_{\mathcal{W}}}$ を求めることを考える．まず第 i 特徴点について, その特徴点をフレーム ψ_n のカメラ位置から見た場合の方向ベクトル $\mathbf{v}^n_{i\,\mathcal{C}}$ は, ψ_n におけるカメラ中心座標系では

$$\mathbf{v}^n_{i\,\mathcal{C}} = \frac{1}{\sqrt{(u^n_i)^2 + (v^n_i)^2 + f_u^2}} (u^n_i, v^n_i, f_u)^T \tag{14}$$

と表現される．さらに世界座標系では,

$$\mathbf{v}^n_{i\,\mathcal{W}} = (\mathbf{R}^n_{\mathcal{W}})^{-1} \mathbf{v}^n_{i\,\mathcal{C}} \tag{15}$$

と表現することができる．理想的には $\mathbf{t}^n_{\mathcal{W}}$ は, $I^n_i = 1$ を満たす任意の i について, 点 $\mathbf{p}_{i\mathcal{W}}$ を通り方向ベクトル $\mathbf{v}^n_{i\,\mathcal{W}}$ を持つ直線上を通るが, 実際には $\mathbf{R}^n_{\mathcal{W}}$ の推定値 $\hat{\mathbf{R}^n_{\mathcal{W}}}$ および $\mathbf{p}_{i\mathcal{W}}$ の推定値 $\hat{\mathbf{p}^{n-1}_{i\mathcal{W}}}$ には推定誤差が含まれているため, $\mathbf{t}^n_{\mathcal{W}}$ を以下の $D(\mathbf{t}^n)$ を最小化する $\mathbf{t}^n$ として推定する．

$$D(\mathbf{t}^n) = \sum_{i=1}^{M} I^n_i \cdot \{w_i \cdot dist(\mathbf{t}^n, \hat{\mathbf{p}^{n-1}_{i\mathcal{W}}}, \hat{\mathbf{v}^n_{i\,\mathcal{W}}})\}^2, \tag{16}$$

ただし, $\hat{\mathbf{v}^n_{i\,\mathcal{W}}} = (\hat{\mathbf{R}^n_{\mathcal{W}}})^{-1} \mathbf{v}^n_{i\,\mathcal{C}}$ とし, w_i は i 番目の特徴点の適当な重み (**特徴点重み**と呼ぶ. 詳細については後述する.) とし, また $dist(\mathbf{p}_1, \mathbf{p}_2, \mathbf{v}_2)$ は, 方向ベクトル $\mathbf{v}_2$ を持ち点 $\mathbf{p}_2$ を通る直線と点 $\mathbf{p}_1$ の間の距離とする．このような $\mathbf{t}^n$ の値は重み付き最小二乗法によって求めることができる．

2.3 地図の拡張および精製

式 (7), (8), (13) に出現するパラメータのうち, z_i 以外のものは観測によって直接測定されるか, もしくは統計的に推定することができる．一方 z_i は, $\mathbf{p}_{i\mathcal{W}}$, $\mathbf{t}^n_{\mathcal{W}}$ および $\mathbf{R}^n_{\mathcal{W}}$ から求められるが, $\mathbf{p}_{i\mathcal{W}}$ の値は本アルゴリズムによってフレーム ψ_n 毎に $\hat{\mathbf{p}^n_{i\,\mathcal{W}}}$ として推定されるため, この値の精度を反復的に向上させていくことを考える．理想的には $\mathbf{p}_{i\mathcal{W}}$ は, 過去のカメラ位置 $\mathbf{t}^m_{\mathcal{W}}$(ただし, $1 \leq m < n$) を通り, 方向ベクトル $\mathbf{v}^m_{i\,\mathcal{W}}$ を持つ直線群の交点と一致する．各 $m(1 \leq m < n)$ について, $\mathbf{t}^m_{\mathcal{W}}$ の推定値 $\hat{\mathbf{t}^m_{\mathcal{W}}}$ および $\mathbf{v}^m_{i\,\mathcal{W}}$ の推定値 $\hat{\mathbf{v}^m_{i\,\mathcal{W}}}$ を ψ_n の開始以前に求めることができるが, それらの値には一般に推定誤差が含まれているため, $\mathbf{p}_{i\mathcal{W}}$ を以下の $D(\mathbf{p}_i)$ を最小化する $\mathbf{p}_i$ として推定する．

$$D(\mathbf{p}_i) = \sum_{m=1}^{n} I^m_i \cdot \{w_m \cdot dist(\mathbf{p}_i, \hat{\mathbf{t}^m_{\mathcal{W}}}, \hat{\mathbf{v}^m_{i\,\mathcal{W}}})\}^2, \tag{17}$$

ただし, $dist$ は前節で定義したものと同様であり, w_m は m 番目の直線の適当な重み (**光線重み**と呼ぶ. 詳細については後述する.) とする．また, 前節と同様にこのような $\mathbf{p}_i$ の値は重み付き最小二乗法を用いて求めることができる．なおここでの重み付き最小二乗法は, 計算時間が現在までのフレーム数 N に依存しないようフレーム毎にインクリメンタルに求められる．

式 (8) および (13) に関する重み付き最小二乗法における第 i 特徴点の奥行き z_i および重み w_i は以下のように計算する．第 i 特徴点がフレーム ψ_l からフレーム $\psi_m (m \geq l \geq 0)$ までカメラの視野内で観測されたとする (すなわち, $I^l_i = \cdots = I^m_i = 1$). このとき, この特徴点の推定三次元位置 $\hat{\mathbf{p}^n_{i\,\mathcal{W}}}$ は, ψ_l から $\psi_m (m \geq l \geq 0)$ までの各フレームで (16) 式を用いて再計算される．

- もし $n = 1$ ならば, $z_i = \mathrm{c}_{\mathrm{depth}}$ とする．ただし, $\mathrm{c}_{\mathrm{depth}}$ は奥行きが未知である場合の初期値である．(ψ_1 ではどの特徴点の三次元位置も推定されていないため.)
- もし $I^{n-1}_i = 0$ または $I^n_i = 0$ ならば, z_i を求める必要はない. (第 i 特徴点の情報が重み付き最小二乗法で使用されないため.)
- もし $I^{n-1}_i = 1$ かつ $I^n_i = 1$ ならば, z_i は以下のように

計算される．

$$z_i = (\mathbf{p}_{i\mathcal{W}}^{\hat{n-1}} - \mathbf{t}_{\mathcal{W}}^{\hat{n-1}}) \cdot (\mathbf{R}_{\mathcal{W}}^{\hat{n-1}}(0,0,1)^T).$$

また, w_i は以下のように求められる．

$$w_i = n - l - 1.$$

直観的に, w_i は第 i 特徴点の年齢を表す．ここで, $n = 1$ のときにすべての特徴点の奥行きが初期値と一致することに注意されたい．これは, 我々のアルゴリズムが環境マップに関する事前知識を一切使用せず, また初期化ステージも持たないことを意味している．式 (16) から明らかなように, 地図の生成処理はカメラ位置および姿勢の推定結果に大きく依存している．

3. 仮想環境における実験

3.1 実験方法

提案アルゴリズムの精度評価実験を行うために, Java を用いて提案アルゴリズムの実装を行った．本節の実験では計算機上に仮想環境を構築し, その中でカメラを仮想的に動かして推定精度の評価を行う．SLAM アルゴリズムではカメラの自己位置及び姿勢の推定と地図の生成が同時に行われるため, 評価指標について以下の 3 点を考える．

- 地図の推定精度
- カメラ位置の推定精度
- カメラ姿勢の推定精度

実験を行う際に, 2 種類のカメラの運動を考える．構築する仮想環境とカメラの運動は以下の通りである．

- 仮想環境: x, y, z 方向に原点を中心としてそれぞれ 10 個ずつ，計 1000 個の格子点を計算し, それらを特徴点とする．特徴点は x 軸, y 軸, z 軸に対し 10 ずつ等間隔で配置する．
- カメラ運動 1: 原点を中心とする x-z 平面上の半径 10 の円周に沿ってカメラが常に z 軸方向を向くように固定して並進運動をさせる．円周上を全部で 10 周させる．
- カメラ運動 2: 原点を中心とする x-z 平面上の半径 10 の円周に沿って, カメラが常に進行方向を向くように回転運動と並進運動の合成運動をさせる．円周上を全部で 10 周させる．

なお仮想環境における実験では, 特徴点の画像平面上の座標はカメラモデルを使って厳密に算出され, またすべての特徴点が視野内で正しく追跡されると仮定する．

3.2 実験の結果と評価

3.2.1 特徴点位置に基づいた校正の評価

2.2 節で述べたように, 2.1 節のアルゴリズムで推定されたカメラ運動パラメータをそのまま用いてもカメラの自己位置推定は可能である．本節では, 2.1 節のアルゴリズムで推定された結果をそのまま用いた場合と, 2.1 節で説明した特徴点位置に基づいたカメラ位置の校正を行った場合の推定精度の比較を行う．特徴点位置に基づいた校正を行わなかった場合のカメラ運動 1, 2 の自己位置の推定結果それぞれ図 3(a), (b) に示す．カメラ運動 1 は並進運動のみで構成されるため, 特徴点が比較的長時間視野内に滞在する傾向があることから, 自己位置推定は比較的高い精度でできると考えられる．実際図 3(a) の結果を見ると, 特徴点位置に基づいた校正を行わなくても十分な精度が得られている．したがって, 以降はカメラ運動 2 のみを対象に精度評価を行っていく．特徴点位置に基づいた校正を行った場合のカメラ運動 2 の自己位置の推定結果は図 3(c) に示されている．カメラ運動に回転要素が加わった場合, 特徴点位置に基づいた校正を行わないとスケールの変動が発生するが, 特徴点位置に基づいた校正を行うことでスケールが安定することがわかる．

3.2.2 抽出した特徴点の画像座標に誤差が含まれている場合の評価

一般に, 実環境でカメラから得た画像から自然特徴点を抽出すると, その特徴点の抽出座標には誤差が含まれる．これを考慮し, 仮想環境上で算出した特徴点の画像平面上の座標に ±0.5 画素以内のランダムなノイズを加えて自己位置推定の評価実験を行った．特徴点位置に基づいた校正を行った場合のカメラ運動 2 の自己位置の推定結果を図 4(a) に示す．図からわかるように, 自己位置推定の精度が極めて低い結果となった．この結果を踏まえて, 誤差を含む場合でも精度のよい自己位置推定ができるように, アルゴリズムの拡張を行う．拡張の詳細は次節で説明する．

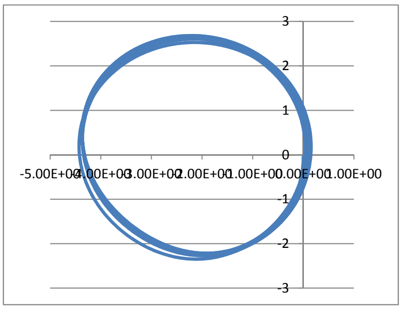
4. アルゴリズムの拡張

4.1 抽出座標に含まれる誤差に対する対処

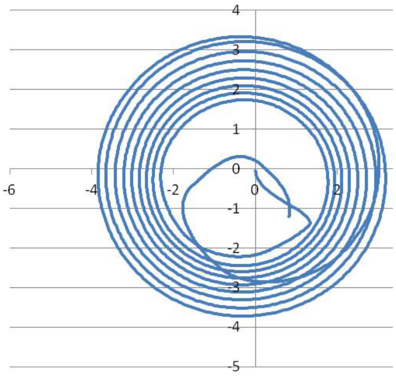
上記仮想実験の結果から, 提案アルゴリズムにおいては抽出した特徴点の画像平面上の座標に含まれる誤差が, 推定精度に大きな影響を及ぼしていることがわかる．画像から抽出した特徴点 p_1, p_2 について, それぞれの視点からの距離 d_1, d_2 が $d_1 > d_2$ の関係にあった場合, p_1 の推定座標の方が画像平面上の座標に含まれる誤差の影響を強く受けると考えられる．これは画像平面上の観測誤差が同じであっても, 観測地点から遠い特徴点ほど推定位置の誤差が大きくなるためである．そこで, 式 (16) および (17) に関する最小二乗法について, 特徴点までの距離が大きいほど軽くなるよう重みをつけることを考える．

4.2 スケールドリフト問題への対処

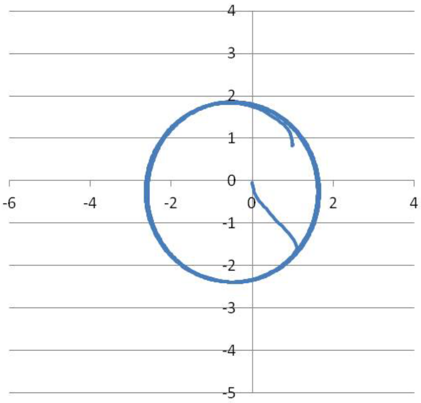
単眼 SLAM 問題では観測情報は実空間のスケールの影響を受けない．したがって, 観測情報に基づいて推定された空間のスケールは実空間のスケールとは独立しており, そのため一般に推定空間の実空間に対するスケール比は安



(a) 特徴点位置に基づいた校正を無効
(カメラ運動 1)



(b) 特徴点位置に基づいた校正を無効
(カメラ運動 2)



(c) 特徴点位置に基づいた校正を有効
(カメラ運動 2)

図 3 仮想環境における自己位置推定の特徴点位置に基づいた校正の有無による違い.

定しない傾向にある. この問題はスケールドリフト問題 [7] と呼ばれる. 特に本アルゴリズムでは地図生成と自己位置推定が相互に依存するため, アルゴリズム固有の性質によりスケール比が単調増加もしくは単調減少する傾向がみられる. この問題に対処するため, (16) 式に関する最小二乗法を用いてカメラ位置の校正を行う際に, 特徴点毎に, 遠くに推定された特徴点か近くに推定された特徴点のいずれを重視するかを示す距離に依存した重みをつける.

4.3 特徴点重みおよび光線重みの算出

4.1 節および 4.2 節で述べた重みは, 式 (16) および式 (17) に関する最小二乗法で用いられる. それぞれの重みについて具体的な計算方法を示す. 第 i 特徴点がフレーム ψ_m で観測されたとき, (17) 式で用いる光線重み w_m を,

$$w_m = \frac{1}{|\mathbf{p}_{i\mathcal{W}}^{\hat{m}} - \mathbf{t}_{\mathcal{W}}^{\hat{m}}|}.$$

で与える. これは, 4.1 節で述べた対処である.

一方, (16) 式で用いる特徴点重みは 4.1 節で述べた問題と 4.2 節で述べた問題の両方に対処しなければならない. そこで, フレーム ψ_n で観測された第 i 特徴点の特徴点重み w_i を以下のように与える.

$$w_i = \frac{1}{|\mathbf{p}_{i\mathcal{W}}^{\widehat{n-1}} - \mathbf{t}_{\mathcal{W}}^{\hat{n}}|} \times |\mathbf{p}_{i\mathcal{W}}^{\widehat{n-1}} - \mathbf{t}_{\mathcal{W}}^{\hat{n}}|^{\alpha} = |\mathbf{p}_{i\mathcal{W}}^{\widehat{n-1}} - \mathbf{t}_{\mathcal{W}}^{\hat{n}}|^{\alpha-1}.$$

ここで, α はスケールドリフトが推定に与える影響を小さくための係数である. なお, $|\mathbf{p}_{i\mathcal{W}}^{\widehat{n-1}}$ - $\mathbf{t}_{\mathcal{W}}^{\hat{n}}|^{\alpha}$ が 4.2 節で述べた対処に相当し, $\frac{1}{|\mathbf{p}_{i\mathcal{W}}^{\widehat{n-1}} - \mathbf{t}_{\mathcal{W}}^{\hat{n}}|}$ が 4.1 節で述べた対処に相当する.

5. 拡張したアルゴリズムの評価実験

5.1 抽出した特徴点の画像座標に誤差が含まれている場合の評価

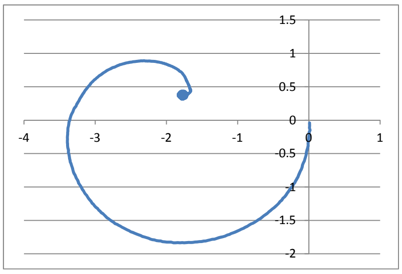
上記で用意した画像平面上の座標にランダムノイズを加えた仮想実験を, 拡張したアルゴリズムで行った. 特徴点重みのみを有効に (光線重みを定数に設定) した場合のカメラ運動 2 の自己位置の推定結果を図 4(b) に示す. 光線重みと特徴点重みを用いた (いずれも定数に設定しない) 場合のカメラ運動 2 の自己位置の推定結果を図 4(c) に示す. 特徴点重みの係数 α は最も推定結果が良かった場合の値である 1.75 とした. 図 4 より, 光線重みと特徴点重みの両方が自己位置の推定精度の向上に貢献していることがわかる.

5.2 速度評価

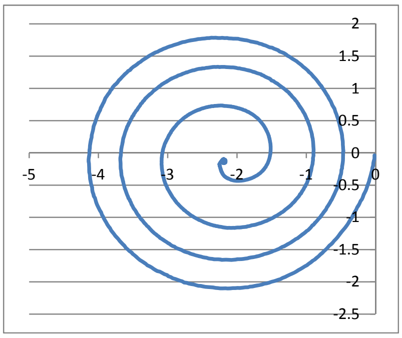
図 4(c) の計算機実験において, 提案アルゴリズムの実行速度の測定を行った. 実行は, Intel Core2 Duo CPU 2.80GHz, 4.00GB RAM, 32 ビット版 Windows7 のノート PC 環境で行った. 全 3600 フレームを実行した結果, 平均追跡特徴点数は 52, フレームあたりの平均実行速度は 0.82ms であった. 計算環境が通常のスマートフォンよりは高性能であり, また実環境での実行と比較して追跡すべき特徴点数が少ない点でも有利であるといえるが, その点を考慮に入れたとしてもアルゴリズム単体で見ると十分高速であり, アルゴリズムの時間計算量の低さが効果を発揮しているといえる.

5.3 実環境における実験

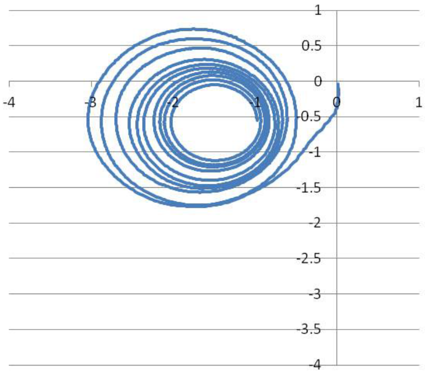
図 5(a) に示した屋内環境で, 実際にカメラを移動させて自己位置推定を行った. 実験に使用した実装では, 画像からの自然特徴点の抽出として OpenCV 2.0 に実装されている Lucas-Kanade 法 [4] を用いた. なお, 提案アルゴリズムを実装している Java プログラムから OpenCV 2.0 への呼び出しには, JNA(Java Native Access) を使用した. また, 特徴点重みの係数 α は実環境で最も推定結果が良かった値である 1.6 を用いた. カメラは, Logicool 社製 HD Pro Webcam C910 (640 × 480 ピクセル, 30FPS, 視野角 78 度) を用いた. 実験では, カメラを固定したノート PC を台車に



(a) 光線重みと特徴点重みのいずれも無効
(カメラ運動 2)



(b) 特徴点重みのみ有効
(カメラ運動 2)



(c) 光線重みと特徴点重みのいずれも有効
(カメラ運動 2)

図 4 特徴点の画像平面上座標にランダムノイズを加えた場合の自己位置推定の比較 (いずれも特徴点位置に基づいた校正を行っている).

載せて, 4m 前進 → 4m 後退 → 2m 左移動 → 4m 右移動の順で並行移動させた. 推定されたカメラの軌道を図 5(b) に示す. 全部で 1776 フレーム実行し, 平均追跡特徴点数は 482 であった. 図からわかるように, 軌道の推定が安定するまでに多少の時間を要するものの, 安定した後は概ね推定ができているといえる. なお, 本稿では紙面の都合上詳細は省略するが, 実環境でより自由にカメラを移動させた場合 (特に回転運動を含む場合) は, 途中で追跡に失敗したりスケールドリフトが発生するケースも見られた.

6. 考察と今後の課題

提案アルゴリズムの仮想環境における実験を行い, 自己位置推定については十分な精度と実行速度を得ることができた. しかしながら実環境での実験では, ある程度の推定精度が得られた一方で, 実用レベルのロバスト性を達成することはできなかった. 今後, 時間計算量を増やさない範囲で, より高いロバスト性が得られるよう拡張を行っていきたい. また, 今のところスマートフォンへの移植作業が完了していないため, 実際にスマートフォン上で動作させた場合の実験ができていない. 今後スマートフォン上で実験できるように, 移植作業についても進めていきたい.

7. 関連研究

SLAM 問題はロボット工学分野で主に研究されてきた. 代表的なものとして, 拡張カルマンフィルターによるもの (EKF-SLAM[5]) と, パーティクルフィルタによるもの (FastSLAM[6]) が知られている. これらの技術の単眼 SLAM 問題への適用もいくつか提案されている [1] が計算コストが高く, またカメラが人の手によってより自由かつ高速に動かされるマーカレス AR 環境ではロバスト性を維持しにくいという問題がある.

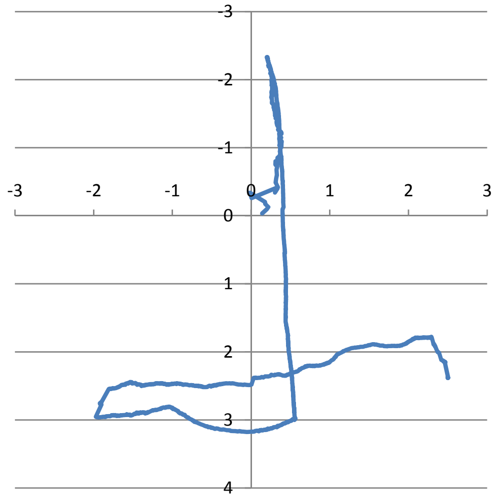
代表的なマーカレス AR システムとしては Klein らが開発した PTAM(Parallel Tracking and Mapping)[2] が挙げられる. PTAM は, 計算コストが高いバンドル調整に基づいた地図生成処理を自己位置・姿勢推定処理から分離し, 異なる CPU コアで並列実行させることによって, ロバスト性の高い AR トラッキングをノート PC 上でも実行可能にしている. さらに Klein らは PTAM のアルゴリズムを最適化し, スマートフォン上でも動作可能にした [3]. しかしながら, 本質的に AR トラッキングの精度がバンドル調整の実行に依存しているため, PC 上で動作させる場合と同等の精度およびロバスト性を実現するには至っていない.

本研究では計算コストが高いバンドル調整を使用せずに単眼 SLAM 問題を解くアプローチを採用している. 特徴点の総数を M フレーム数を N としたときに, バンドル調整の時間計算量が $O(N^3)$ または $O(MN^2)$ になるのに対し, 提案アルゴリズムの時間計算量は $O(M)$ となり, N には依存しない. したがって, 提案アルゴリズムは PC よりも計算能力の劣る携帯情報端末などにより適していると考えられる. しかしながら現状では, バンドル調整を用いるアルゴリズムと同等の精度を達成するには至っていない. また, AR トラッキングに失敗したときに局所最適解に陥るケースも見つかっている. ただし, PTAM のように AR トラッキングを地図の初期化処理やキーフレーム毎に実行される地図の拡張処理に依存していないため, 仮に地図の拡張が頻繁に要求されるような状況でも大きくトラッキングを失敗しないという特長もある. したがって, ロバスト性については一概に比較することはできない.

8. おわりに

スマートフォンなどの携帯情報端末上で動作する実用的なマーカレス AR を実現することを目指して, 先行研究で開発した単眼 SLAM アルゴリズムのロバスト性を向上させる改良を行った. その結果, 改良前には十分な推定精度が得られなかったランダムノイズを混入させた仮想環境でも, 十分な実行速度と推定精度を得ることに成功した. 一

(a) 実験で使用した屋内環境



(b) 自己位置の推定結果

図 5 屋内環境における実験.

方, 実環境で実験を行ったところ精度およびロバスト性の点で十分な結果を得ることはできなかった.

今後, 実行速度の低下を最小限に留めつつ推定精度の向上を図ることによって, 実環境での使用に耐え得るようアルゴリズムの改良を行っていく予定である.

参考文献

[1] Ethan Eade and Tom Drummond: Scalable monocular SLAM, IEEE Conference on Computer Vision and Pattern Recognition (CVPR'06), 2006.
[2] Georg Klein and David Murray: Parallel tracking and mapping for small AR workspaces, In Proc. of IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07), 2007.
[3] Georg Klein and David Murray: Parallel tracking and mapping on a camera phone, In Proc. of IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'09), 2009.
[4] Bruce D. Lucas and Takeo Kanade: An iterative image registration technique with an application to stereo vision, In Proc. of Imaging understanding workshop, pp 121–130, 1981.
[5] Randall C. Smith and Peter Cheeseman: On the representation and estimation of spatial uncertainty, International Journal of Robotics Research, Vol. 5, No. 4, pp. 56–68, 1986.
[6] Michael Montemerlo, Sebastian Thrun, Dapjne Koller and Ben Wegbreit: FastSLAM 2.0: An improved particle filtering algorithm for Simultaneous localization and mapping that provably converges, In Proc. of International Joint Conference on Artificial Intelligence, pp. 1151–1156, 2003.
[7] Hauke Strasdat, J. M. M. Montiel and Andrew Davison: Scale drift-aware large scale monocular SLAM, In Proc. of 2010 Robotics: Science and Systems Conference (RSS'10), 2010.