



三次元モデルのテクスチャの作成

M1 田川幸汰

1 概要

カメラの内部パラメータ、外部パラメータを用いて3次元モデルにテクスチャを張り付ける方法の解説と、実際に取得したテクスチャ画像の考察について示す。先週までに求められているカメラの内部、外部パラメータを下に示す。

2 テクスチャ画像の取得方法

3 テクスチャ画像取得結果

- ショッピングモールのフロアマップから3次元モデルを生成する。
- ショッピングモール内を全方位カメラで撮影し、そこから透視投影画像を生成する。
- 透視投影画像から得られる点、線特徴量からカメラ姿勢を推定する。
- 3次元モデルに対して正しい大きさ、位置、方向でテクスチャ画像を割り当てる。

このうち、本研究では主に2,4の手順について担当する。特に、全方位画像から3次元モデルのテクスチャを生成する手法を用いることで、高精度なショッピングモールの3次元モデルを自動的かつ効率的に生成することを目指す。今回の進捗報告では、研究の準備として行った全方位画像から透視投影画像の生成と3次元モデルと透視投影画像の対応からのカメラ位置姿勢推定の結果について示す。

4 全方位画像から透視投影画像の生成

全方位画像から任意視線方向の透視投影画像を生成する。今回は透視投影画像を生成する際にOpenCVのremap関数を用いた。OpenCvのremap関数を用いるメリットとしてmapを使い回すことで幾何変換を行う回数を減らせることができる点がある。特に複数の画像に対して同じ幾何変換を適用する場合に、処理速度が大きく向上する。透視投影画像の生成手法については、remap関数を利用した部分について説明する。

4.1 remap関数を利用した透視投影画像生成

remap関数は変換前の画像とX座標のマップ、Y座標のマップ、画像の補完手法を引数として、変換後の画像を返す。ここで、マップとは出力先の各軸の座標が入っている。例として無変換の場合のX座標は $[0, 1, 2, 3, \dots]$ となり、X軸方向に2倍に拡大したい場合のX座標は $[0, 0.5, 1.0, 1.5, \dots]$ となる。 X 軸方向の出力画像座標データ \mathbf{U} 、 Y 軸方向の出力画像座標データ \mathbf{V} を以下のように定義する。

$$\mathbf{U}(W_e \times H_e) = \begin{pmatrix} 0 & 1 & 2 & \dots & W_p \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 1 & 2 & \dots & W_p \end{pmatrix}, \mathbf{V}(W_p \times H_p) = \begin{pmatrix} 0 & \dots & 0 \\ 1 & \dots & 1 \\ 2 & \dots & 2 \\ \vdots & \ddots & \vdots \\ H_p & \dots & H_p \end{pmatrix} \quad (1)$$

\mathbf{U} 、 \mathbf{V} から求めた視線ベクトル (x, y, z) を回転行列で回転する。視線ベクトルを角度に変換し、これを用いて透視投影画像の座標データ \mathbf{U} 、 \mathbf{V} が求められる。



4.2 出力結果

入力した全方位画像図1(a)、出力された透視投影画像を図1(b)に示す。また、画像出力に用いたパラメータを下に示す。

- 透視投影画像サイズ (600×800)
- スケール 4.0
- 視線角度 $\theta = 0^\circ$
- 視線角度 $\phi = 0^\circ$



(a) 全方位画像



(b) 透視投影画像

図 1: 出力結果

5 直行射影の共線性と共面性を用いたカメラ姿勢推定

直行射影の共線性と共面性を用いたカメラ姿勢推定について簡潔に説明する。



直行射影の共線性とは直行射影された複数の点が同一直線状に並ぶ性質のことである。Lu らの論文では、カメラ姿勢の反復的最適化を行う際に共線性誤差を検証することで、カメラ姿勢推定の精度と効率を向上させている。

直行射影の共面性とは射影された複数の点や線が同一平面上に並ぶ性質のことである。Zhang らの論文では、カメラ姿勢の反復的最適化を行う際に共面性誤差を検証することで、カメラ姿勢推定の精度と効率を向上させている。

直行射影の共線性を用いた点対応からカメラ姿勢を推定する方法と、直行射影の共面性を用いた線対応からカメラ姿勢を推定する方法は、直行射影行列が異なるだけである。また、直線の方向ベクトルについても同様に扱うことができるため、カメラ姿勢を推定するための目的関数を、これらをすべて足すことで次の式のように定義する。

$$E(\mathbf{R}, \mathbf{t}) = \sum_{a=1}^N \|(\mathbf{I} - \mathbf{V}_a)(\mathbf{R}\mathbf{p}_a + \mathbf{t})\|^2 + \sum_{a=1}^M \|(\mathbf{I} - \mathbf{K}_a)(\mathbf{R}\mathbf{r}_a + \mathbf{t})\|^2 + \sum_{a=1}^M \|(\mathbf{I} - \mathbf{K}_a)\mathbf{R}\mathbf{d}_a\|^2 \quad (2)$$

目的関数 $E(\mathbf{R}, \mathbf{t})$ から、並進ベクトル \mathbf{t} 、回転行列 $\mathbf{R}^k, \mathbf{R}^{k+1}$ を求め、回転行列のノルムの差が収束するまで計算を行うことで、最適な並進ベクトル \mathbf{t} と回転行列 \mathbf{R}^k を求める。

5.1 全方位カメラのキャリブレーション

カメラ位置姿勢推定を行う際に全方位カメラの内部パラメータである焦点距離を使用する。そのため事前準備として全方位カメラのキャリブレーションを行った。

全方位カメラのキャリブレーションは以下の手順で行う。

- 全方位カメラでチェッカーボードを異なる角度から 10 枚以上撮影
- 全方位画像から、チェッカーボードの全体が写る視点で透視投影画像を生成
- キャリブレーションを行うプログラムを実行

注意点として透視投影画像を生成する際、解像度は変更しないようにする。また、キャリブレーションを行うプログラムは OpenCV のチェッカーボードの交点の座標を調べる関数 `findChessboardCorners`、3 次元座標と 2 次元座標の対応から内部パラメータを計算する関数 `calibrateCamera` などが用いられている。

全方位画像から生成された透視投影画像のキャリブレーション結果の一部を図 2 に示す。また、出力されたカメラの



図 2: キャリブレーション結果



内部パラメータ行列 M を以下に示す。

$$M = \begin{pmatrix} 511.69918760786322 & 0.0000000000000000 & 784.54139306531715 \\ 0.0000000000000000 & 499.73234060481144 & 536.82834728101511 \\ 0.0000000000000000 & 0.0000000000000000 & 1.0000000000000000 \end{pmatrix} \quad (3)$$

これより、チェッカーボードの交点を検出し、カメラの内部パラメータが推測されていることがわかる。また焦点距離 $f_x = 511.69918760786322$ 、 $f_y = 499.73234060481144$ となったが、今回焦点距離は各軸共通なので f_x と f_y の平均を用いる。

6 カメラ位置姿勢推定の結果

6.1 入力

直行射影の共線性と共面性を利用したカメラ姿勢の推定の際に必要な入力を以下に示す。

- p_a : 世界座標系で表現された空間点の座標
- v_a : p_a に対応する画像上の特徴点座標
- d_a : 世界座標系で表現された直線 L_a の方向ベクトル
- r_a : 世界座標系で表現された直線 L_a 上の点の座標
- n_a : L_a に対応する画像上の直線のパラメータベクトル
- R^0 : 回転行列の初期値
- thd : 収束に用いるしきい値

p_a は3次元モデルの頂点の座標の一部を用いる。ただし透視投影画像に映らない点については、空間点の座標として用いない。3次元モデルについては C 棟エレベータ前の廊下のモデルを用いる。3次元モデルを図 3 に示す。

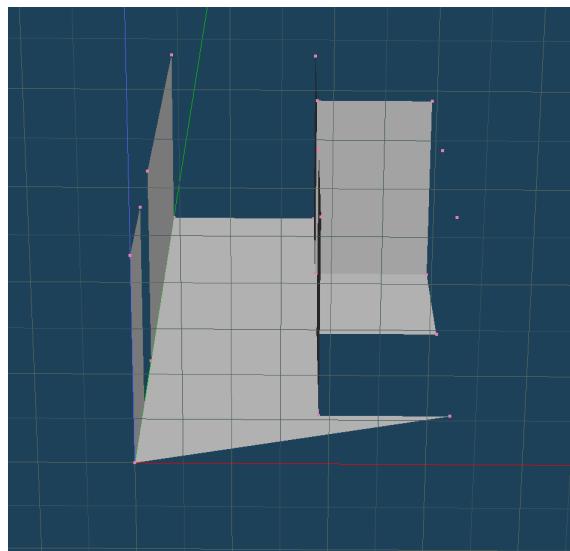


図 3: 3d モデル

v_a は透視投影画像を表示し p_a に対応する画像の座標を選択して求める。(図 4)。また $v_a = (x_a, y_a, f)$ で、 (x_a, y_a) は光軸点を原点とした画像点の座標であるため、以下の式で平行移動する。

$$x_a = x - \frac{W}{2}, y_a = y - \frac{H}{2} \quad (4)$$

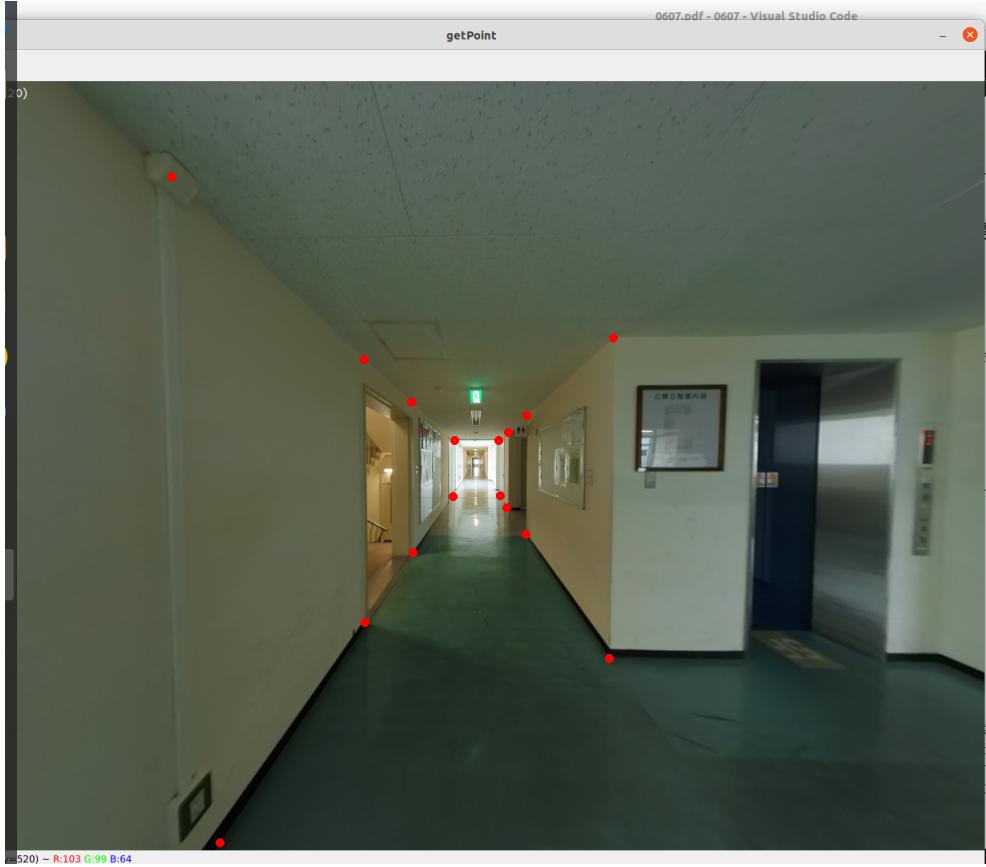


図 4: 点特徴を選択

さらに、焦点距離 f を座標の後ろに追加する必要がある。焦点距離 f については、全方位カメラのキャリブレーションを行い計算した。

$\mathbf{d}_a = \mathbf{d}_1, \dots, \mathbf{d}_i, \dots$ については3次元モデルの頂点の座標の一部 (\mathbf{p}_a で使用していない点) を用いて以下の式で求める。

$$\mathbf{d}_i = \frac{\mathbf{p}_{i+1} - \mathbf{p}_i}{|\mathbf{p}_{i+1} - \mathbf{p}_i|} \quad (5)$$

\mathbf{r}_a については3次元モデルの頂点の座標の一部 (\mathbf{p}_a で使用していない点) を用いる。

$\mathbf{n}_a = \mathbf{n}_1, \dots, \mathbf{n}_i, \dots$ については \mathbf{v}_a の座標を用いて以下の式で用いる。

$$\mathbf{n}_i = \frac{\mathbf{n}_{i+1}\mathbf{n}_i^\top}{|\mathbf{n}_{i+1}\mathbf{n}_i^\top|} \quad (6)$$

\mathbf{R}^0 は世界座標系とカメラ座標系の関係から以下の行列を設定する。また、 $thd = 1.0^{-8}$ とする。

$$\mathbf{R}^0 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \quad (7)$$

また、カメラ位置の推定結果については、 $\mathbf{c} = (0.0, 0.0, 1.5)$ 程度になると考えられる。

6.2 実行結果

出力された回転行列 \mathbf{R} 、及びカメラ位置 \mathbf{c} を以下に示す。

$$\mathbf{R} = \begin{pmatrix} 0.99848126 & 0.01266343 & -0.05361733 \\ -0.05432481 & 0.06440953 & -0.99644379 \\ -0.00916493 & 0.9978432 & 0.06499965 \end{pmatrix} \quad (8)$$



$$\mathbf{c} = \begin{pmatrix} 0.99930194 \\ 2.46231046 \\ 1.18038819 \end{pmatrix} \quad (9)$$

回転行列は \mathbf{R}^0 と近い値が出力された。ただ、カメラ位置は y の値が少し大きくなってしまうことがわかった。この原因として、線特徴の選び方が良くなかったことが考えられる。最低でも 3 つの平面上の線特徴を選択するようにプログラムを変更して(図 5)、出力された回転行列 \mathbf{R} 、及びカメラ位置 \mathbf{c} を以下に示す。

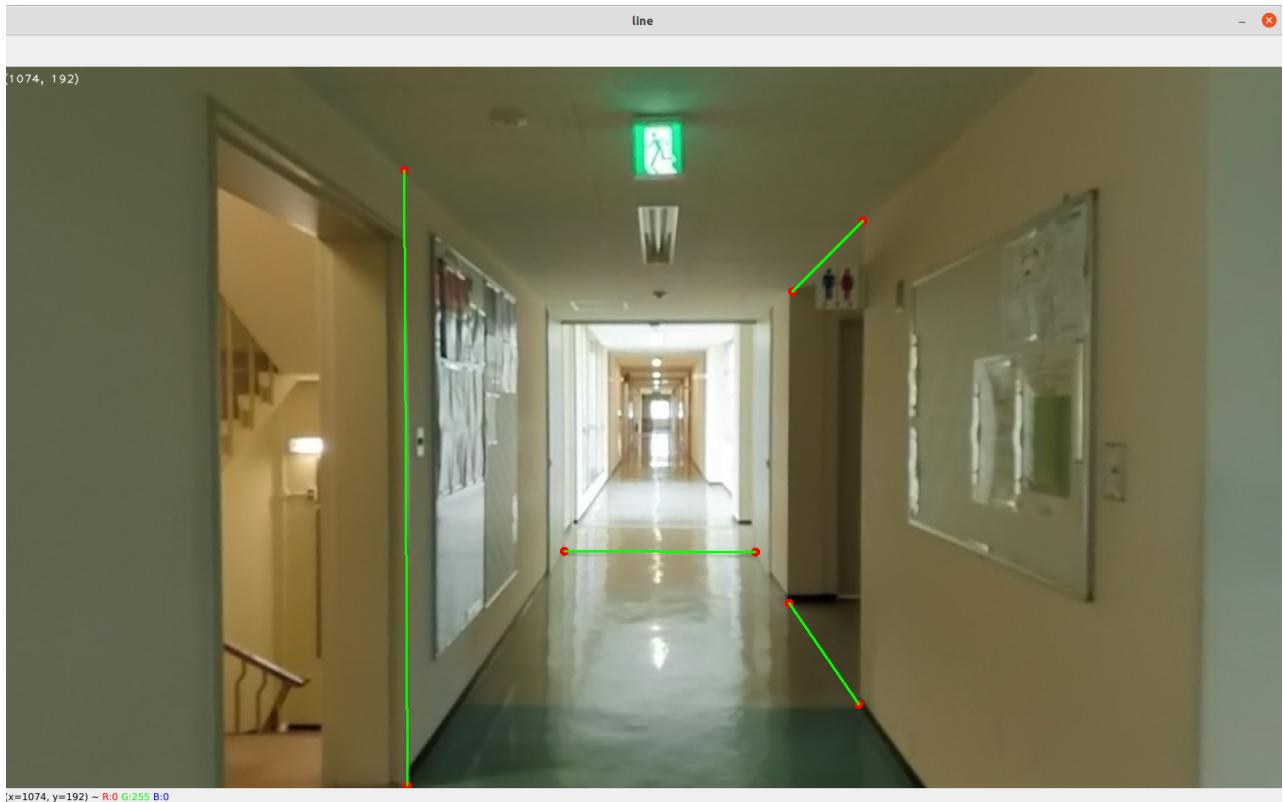


図 5: 線特徴を選択

$$\mathbf{R} = \begin{pmatrix} 0.99862515 & 0.03929384 & -0.03469576 \\ -0.02785779 & -0.16286014 & -0.98625581 \\ -0.04440433 & 0.98586641 & -0.16154159 \end{pmatrix} \quad (10)$$

$$\mathbf{c} = \begin{pmatrix} 0.79346609 \\ 1.87221025 \\ 1.37222923 \end{pmatrix} \quad (11)$$

カメラ位置は y の値が小さくなり改善されていることがわかる。

7 今後の計画

今後の計画として、計算されたカメラ運動行列を用いてテクスチャの貼り付けに早急に取り掛かりたい。具体的には 6 月中にテクスチャを貼り付けた 3 次元モデルを完成させる。

参考文献

- [1] 菅谷保之, 「直交射影の共線性と共面性を用いたカメラ姿勢の推定」