CH-003

移動量微小仮定に基づく対応点探索の効率化による Point Cloud ベース 3 次元トラッキングの高速化
High-speed 6-DoF Tracking with Sequential Point Clouds Based on the Small-Displacement Assumption

田畑智志 † Satoshi Tabata　渡辺義浩 † Yoshihiro Watanabe　石川正俊 † Masatoshi Ishikawa

1. はじめに

6 自由度の剛体運動の取得 (以下, 3 次元トラッキング) は, ロボットの自己位置推定 [1], VR・AR 分野の応用 [2] において重要である. 特に, 非接触かつマーカレスである 3 次元トラッキングを高速に達成することが重要である.

本稿では, 特に, 3 次元計測によって取得された, 物体の 3 次元点群を用いた手法について述べる. また, 本稿では, 3 次元点群および距離画像を Point Cloud と呼ぶ. このような Point Cloud ベースの手法として, Iterative Closest Point アルゴリズム (ICP アルゴリズム) がある [3]. この手法は, Point Cloud 間の最近傍点探索による点の仮の対応付けと剛体運動推定を交互に行い最適化する反復解法であり, 2 つの Point Cloud のみを用いて, その間の剛体運動を推定する. しかし, ICP アルゴリズムは対応点探索処理の計算コストが高い場合や反復回数が多い場合, 計算量が多いという問題がある. これらの問題に対し, さまざまな手法が提案されている [4, 5, 6].

特に, Point Cloud が時系列に得られる場合には, 反復回数が少なくなるという利点がある. そのため, 入力として時系列に得られる距離画像を対象とし, リアルタイムに 3 次元トラッキングを行う手法 [7] が提案されている. この手法では, 点の対応付けに point-to-projection[8] を用い, 剛体運動推定の誤差関数に point-to-plane[9] を用いている. ここで, point-to-projection は, 一方の距離画像をもう一方の距離画像へ射影することによって対応付けを行う手法である. この対応付け方法は, 射影のみを行い点の探索を行わないため, 対応点を高速に見つけることができる. また, point-to-plane は, 対応付けした点と点の法線に沿った距離を誤差関数とする手法である. 距離画像の計測などの計算コストから 10 Hz 程度となっているが, 3 次元トラッキングに関しては数 ms を達成している.

本稿では, この 3 次元トラッキング手法を構造化光法に基づく高速 3 次元計測手法 [10] に適用できるよう拡張し, 1 ms 以下で 3 次元トラッキングを達成する. そのためには, 以下の 2 つの問題を解決する必要がある.

まず, Point Cloud の密度が低い場合に point-to-projection による点の対応付けの計算コストが増大するという問題がある. これは, 密度が低い場合には一方の距離画像をもう一方の距離画像へ射影したときに, 該当する点が存在しないことに起因する. これにより, 射影した後に更なる探索処理が必要となるため対応点探索処理の計算コストが増大し, さらに密度が低い場合には対応付けが行えない可能性がある. 特に, 構造化光法に基づく高速 3 次元計測手法は得られる Point Cloud の密度が低い場合が多く, この問題が避けられない. 本稿では, Point Cloud がカメラとプロジェクタからなるシステムによって計測されていることに着目し, プロジェクタ画面上で対応をとることで, この問題を解決する. 従来はカメラ画像面上で対応をとるため, 密度が低い場合には追加の処理が必要となる. 一方で, プロジェクタ画像面上で対応をとる場合, 計測はプロジェクタから投影されるパターンに基づいて行われているため, 物体の形状や移動に影響を受けず同一の画素において計測が行われる. また, 計測可能な画素は投影するパターンで決まるため, 使用する点や点ごとの隣接関係を事前に定義できる. これにより, Point Cloud の密度が低い場合にも高速に点の対応付けが可能となる.

次に, 処理時間の問題がある. 従来手法において, 3 次元トラッキングの処理時間が数 ms まで高速化されている. しかし, これには Point Cloud に対する法線推定などの処理時間が含まれていない. また, 今回の目標は, 高速 3 次元計測手法と同程度である 1 ms 以下で 3 次元トラッキングを達成することである. そのため, 更なる高速化が必要である. そこで, 時間差分を小さくすることで, 反復を行わずに運動推定を行う. そのために, 計測のサンプリングレートの高さに着目する. 計測のサンプリングレートが高いとき, 時間差分が微小である. そこで, 計測間の物体の移動量は小さいと仮定する. このとき, 2 つの Point Cloud において, プロジェクタの画像面上の同一画素で計測される点は物体表面の同一の接平面上に存在すると仮定する. この仮定により, 従来は反復によって求めていた収束先の対応点を最初の対応付けで得られるため, 反復せずに剛体運動を推定できる.

本稿では, 近年提案されている高速 3 次元計測 [11] のサンプリングレートである 1,000 Hz を対象とし, シミュレーションと実計測データに対して提案手法を適用した. シミュレーションデータに対する 6 自由度の剛体運動の誤差評価と実計測データに対する形状統合の評価を行い, 数 mm 程度の誤差で統合が可能であることを確認した. また, これらの実験によって 1 ms 以下でトラッキング可能であることを確認した.

† 東京大学大学院 情報理工学系研究科, Graduate School of Information Science and Technology, The University of Tokyo

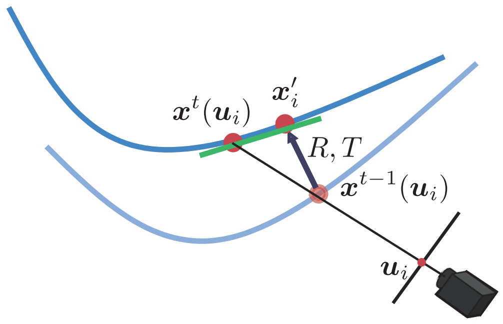

図 1: 時間差分微小時の計測点の移動. 青: 物体の表面形状, 赤: 計測点および計測点が移動した点, 緑: 平面とみなせる範囲. 時刻 $t-1$ に計測された点 $\boldsymbol{x}_i^{t-1}$ が剛体運動 R, T によって, 時刻 t で $\boldsymbol{x}'_i$ に移動したとする. 時間差分が微小である場合に $\boldsymbol{x}'_i$ は, 時刻 t において計測された $\boldsymbol{x}_i^t$ の近傍に存在し, かつ, $\boldsymbol{x}'_i$ と $\boldsymbol{x}_i^t$ の間で物体は平面的であると考えられる.

2. 移動量微小仮定に基づく対応点探索の効率化による Point Cloud ベース 3 次元トラッキングの高速化

2.1.Point Cloud の取得

本稿では, Point Cloud の取得に構造化光法に基づく高速 3 次元計測手法を用いる. 特に, プロジェクタ画像面上で距離画像を考える. そこで, プロジェクタ画像面上の 3 次元計測を行う画素 (以下, 計測点) の集合を P とし, i 番目の計測点を $\boldsymbol{u}_i$ とあらわす.　また, 時刻 t において $\boldsymbol{u}_i$ で計測した点を $\boldsymbol{x}_i^t = \boldsymbol{x}^t(\boldsymbol{u}_i)$ とあらわす.

2.2. 移動量微小仮定に基づく対応点探索の効率化

本手法では point-to-projection をプロジェクタ画像面上で考えることで Point Cloud の密度が低い場合でも適用可能とする. また, Point Cloud 間の移動量が微小であるという仮定を置くことにより, 反復せずに剛体運動を推定する. 図 1 に示すように, 高速 3 次元計測を用いる場合は計測間の時間差分が微小となるため, 計測間で物体の移動量が微小であると考えることができる. 時刻 $t-1$ から時刻 t の間の物体の運動 R, T によって, 時刻 $t-1$ に計測された i 番目の点 $\boldsymbol{x}_i^{t-1}$ が移動した先を $\boldsymbol{x}'_i$ と表す. 時間差分が微小のとき, 時刻 t に計測された i 番目の点 $\boldsymbol{x}_i^t$ と $\boldsymbol{x}'_i$ は物体表面上の近傍に存在すると考える. ここで, 物体の表面形状の曲率半径に対し物体の移動量が小さければ, $\boldsymbol{x}_i^t$ と $\boldsymbol{x}'_i$ の間は平面的であり, プロジェクタ画像面上の同一点で計測される点は物体表面の同一接平面上に存在すると仮定する.

point-to-plane[9] の場合は, 式 (1) に示す誤差関数 E を用いるため, この対応付け方法で誤差が最小となる解が得られる. ここで, $\boldsymbol{x}_i^t, \boldsymbol{x}_i^{t-1}$ は, Point Cloud の点, $\boldsymbol{n}_i^t$ は $\boldsymbol{x}_i^t$ における法線, R, T は Point Cloud 間の運動の回転行列と並進ベクトルである.

$$E = \sum_{i=0}^{m} \left| \boldsymbol{n}_i^t \cdot (\boldsymbol{x}_i^t - (R\boldsymbol{x}_i^{t-1} + T)) \right|^2 \qquad (1)$$

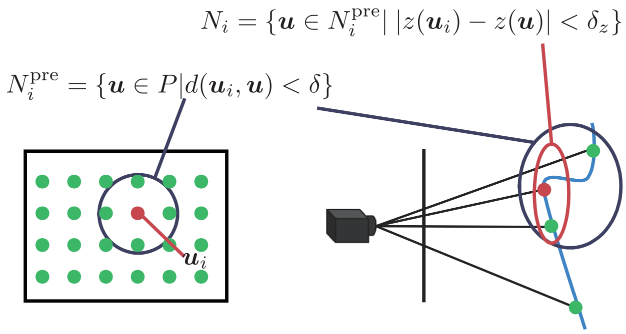

図 2: 法線推定における近傍の定義. 青: 物体表面形状, 緑: 計測点の集合 P を構成する点, 赤: i 番目の計測点. 事前に i 番目の計測点に対し画像面上で近傍に存在する点の集合を N_i^{pre} として定義し, 計測時には N_i^{pre} のうち距離の差分が閾値 δ_z 以上となるものを除外する.

これにより, 従来の ICP アルゴリズムで収束した場合と同等の精度を達成しつつ, Point Cloud の密度が低い場合にも高速な 3 次元トラッキングが可能となる.

2.3. 法線の推定

式 (1) に示す point-to-plane で用いられる誤差関数を用いるには, 点の座標だけでなく法線も必要である. そのため, Point Cloud から各点の法線を推定する必要がある. 法線の推定には各点の近傍点の情報が必要であるが, Point Cloud の近傍探索は計算コストが高い. そこで, 事前に探索範囲を制約することで計算コストを抑える. そのために, 画像面上で近傍に存在しない点は 3 次元空間中においても近傍には存在しないことを利用する. このとき, 近傍を探索する範囲をプロジェクタ画像面上の近傍に絞る. これにより, 近傍探索範囲の次元を下げるとともに, 事前にその範囲も制限することができるため, 計算コストを大幅に抑えることが可能である.

具体的には, 図 2 に示すように, P の各要素 $\boldsymbol{u}_i$ に対して, 画像面上でサンプリングを行い, 近傍点群 N_i^{pre} を定義する. ここで, 式 (2) に示すように画像面上の距離が閾値 δ 以下になる点群を近傍とする.

$$N_i^{\mathrm{pre}} = \{\boldsymbol{u} \in P | d(\boldsymbol{u}_i, \boldsymbol{u}) < \delta\} \qquad (2)$$

計測時には, 3 次元計測により $\boldsymbol{u}_i$ に対して距離 $z(\boldsymbol{u}_i)$ が得られるため, 近傍点群 N_i^{pre} を確認し, 対応する距離の差分が閾値 δ_z 以下になる点の集合 N_i を求め, 3 次元空間における近傍点として扱う.

$$N_i = \{\boldsymbol{u} \in N_i^{\mathrm{pre}} | |z(\boldsymbol{u}_i) - z(\boldsymbol{u})| < \delta_z\} \qquad (3)$$

各点に対し, その近傍点の集合 N_i から法線を推定する. 近傍の点がわかっているとき法線は外積や主成分分析, 関数フィッティングにより求めることができる [12]. ここで, 3 次元計測の計測誤差の影響を低減するためには, 冗長な点数を用いる手法であることが望ましい. また, Point Cloud の密度が低い場合は近傍の点の範囲におい

て平面的と仮定することはできない. そこで, 各点の近傍を 2 次曲面で近似し, その偏微分から法線を求める.

2.4. トラッキングアルゴリズム

2.4.1. 対応点取得

従来の手法では, 2 つの Point Cloud 間で対応する点を探索によって求める必要があった. しかし, 提案手法では事前に定義できるため探索は不要となる. そこで, 時刻 $t-1$ と t においてプロジェクタ画像面上の i 番目の計測点 $\boldsymbol{u}_i$ で計測された点 $\boldsymbol{x}_i^{t-1}$ と $\boldsymbol{x}_i^t$ を対応付ける. また, 物体の形状や移動によっては計測されない場合がある. そのため, $\boldsymbol{x}_i^{t-1}$ と $\boldsymbol{x}_i^t$ がともに計測されている場合のみ, その組を運動推定に使用する.

2.4.2. 運動推定

計測した 3 次元空間中の点 $\boldsymbol{x}_i^{t-1}, \boldsymbol{x}_i^t$, およびその法線 $\boldsymbol{n}_i^t = \boldsymbol{n}^t(\boldsymbol{x}_i^t)$ を用いて, 運動を推定する. 時刻 $t-1$ から時刻 t の間の運動の回転行列を R, 並進ベクトルを T とすると, 時刻 $t-1$ において計測された点 $\boldsymbol{x}_i^{t-1}$ は, 時刻 t において式 (4) であらわす $\boldsymbol{x}'_i$ に移動する.

$$\boldsymbol{x}'_i = R\boldsymbol{x}_i^{t-1} + T \tag{4}$$

ここで, 図 1 に示すように $\boldsymbol{x}'_i$ が時刻 t において計測された点 $\boldsymbol{x}_i^t$ と同一接平面上に存在すると仮定する. このとき, 2 点 $\boldsymbol{x}'_i, \boldsymbol{x}_i^t$ と法線 $\boldsymbol{n}_i^t$ を用いた最小化問題として R, T を求めることができる. 回転の変数をオイラー角を用いて $\boldsymbol{r} = (\alpha, \beta, \gamma)^{\mathrm{T}}$ とおき, この最小化問題を式 (5) のようにあらわす.

$$\begin{pmatrix} \boldsymbol{r}_{\mathrm{opt}} \\ T_{\mathrm{opt}} \end{pmatrix} = \mathop{\arg\min}_{\boldsymbol{r},T} \sum_i \left|\boldsymbol{n}_i^t \cdot (\boldsymbol{x}_i^t - \boldsymbol{x}'_i)\right|^2 + \lambda_R|\boldsymbol{r}|^2 + \lambda_T|T|^2 \tag{5}$$

ここで, 式 (5) の第 2 項, 第 3 項は 平面や球面など対称性を持った Point Cloud の分布によって不良設定問題となる場合に移動量が最小の解を選択するための制約項である.

式 (5) は, 回転の変数に対して非線形な最小化問題であるため, 解析的に解くことができない. そこで, 解析的に解を得るため, 問題の線形化を行う. いま, 計測の時間差分は微小であるから, その間の運動も微小であると考えられる. このことから, 回転 R は線形近似できるとする [13] と,

$$R \approx \begin{pmatrix} 1 & -\gamma & \beta \\ \gamma & 1 & -\alpha \\ -\beta & \alpha & 1 \end{pmatrix} \tag{6}$$

とあらわすことができる. このとき,

$$\begin{aligned} \left|\boldsymbol{n}_i^t \cdot (\boldsymbol{x}_i^t - \boldsymbol{x}'_i)\right|^2 &= \left|\boldsymbol{n}_i^t \cdot (\boldsymbol{x}_i^t - (\boldsymbol{r} \times \boldsymbol{x}_i^{t-1} + T))\right|^2 \\ &= \left|(\boldsymbol{n}_i^t \times \boldsymbol{x}_i^{t-1}) \cdot \boldsymbol{r} + \boldsymbol{n}_i^t \cdot T - \boldsymbol{n}_i^t \cdot \boldsymbol{x}_i^t\right|^2 \end{aligned}$$

となる. ここで, 点の数を m として,

$$A = \begin{pmatrix} (\boldsymbol{n}_1^t \times \boldsymbol{x}_1^{t-1})^{\mathrm{T}} & (\boldsymbol{n}_1^t)^{\mathrm{T}} \\ (\boldsymbol{n}_2^t \times \boldsymbol{x}_2^{t-1})^{\mathrm{T}} & (\boldsymbol{n}_2^t)^{\mathrm{T}} \\ \vdots & \vdots \\ (\boldsymbol{n}_m^t \times \boldsymbol{x}_m^{t-1})^{\mathrm{T}} & (\boldsymbol{n}_m^t)^{\mathrm{T}} \end{pmatrix}, \boldsymbol{b} = \begin{pmatrix} \boldsymbol{n}_1^t \cdot \boldsymbol{x}_1^t \\ \boldsymbol{n}_2^t \cdot \boldsymbol{x}_2^t \\ \vdots \\ \boldsymbol{n}_m^t \cdot \boldsymbol{x}_m^t \end{pmatrix}$$

$$\Lambda = \mathrm{diag}\left[\lambda_R, \lambda_R, \lambda_R, \lambda_T, \lambda_T, \lambda_T\right]$$

とおき, 式 (5) の解を

$$\begin{pmatrix} \boldsymbol{r}_{\mathrm{opt}} \\ T_{\mathrm{opt}} \end{pmatrix} = \left(A^{\mathrm{T}}A + \Lambda\right)^{-1} A^{\mathrm{T}}\boldsymbol{b} \tag{7}$$

として得る. また, $\left(A^{\mathrm{T}}A + \Lambda\right)$ は, 6×6 の対称行列であるため, 修正コレスキー分解を用いて高速に解くことができる.

回転行列である R は直交行列であるが, 式 (7) を計算して得られる $\boldsymbol{r}_{\mathrm{opt}}$ は誤差を含むため, 式 (6) に代入して得られる行列は必ずしも直交行列とならない. そこで, $\boldsymbol{r}_{\mathrm{opt}}$ をオイラー角として再度回転行列を計算し, 求める回転行列 R とする.

3. 実験

実験には Intel(R) Xeon(R) CPU E5-2687W v2 @ 3.40GHz (2 プロセッサ) を搭載した PC を用いた. シミュレーションと実環境で得られた時系列の Point Cloud に対し, 提案手法を適用した. また, 今回の実験において式 (5) のパラメタを $\lambda_R = 0.6, \lambda_T = 0.05$ とした.

3.1. シミュレーション

3.1.1. 実験条件

OpenGL によるシミュレーションを行い, 計測対象とするモデルに対して回転と並進を適用したときの距離画像と法線画像を生成した. これらの画像に対しデータを疎にサンプリングし, 提案手法を適用した. ここで, 画像は幅を 512 px, 高さを 512 px とし, 画像面から 2205 点をサンプリングした. 計測するモデルは The Stanford 3D Scanning Repository[14] より, Stanford Bunny, Armadillo, Lucy, Happy Buddha をシステムにあわせてスケーリングして用いた. また, 各計測対象を計測システムの前方 650 mm の位置に配置し, 物体座標系において y 軸を中心に 0.72 度/フレームで回転, y 軸に沿って 0.15 mm/フレームで並進させ, 1000 フレーム動かした.

3.1.2. 回転・並進の誤差

シミュレーション時に適用した回転・並進を真値とし, 提案手法で推定された各フレーム間の相対運動の誤差と運動開始時を基準とした絶対運動を評価した. ここで, 回転の誤差はクォータニオン $Q = (Q_x, Q_y, Q_z, Q_w)$ を用いて $E_Q = |Q - Q_{\mathrm{truth}}|$ とし, 並進の誤差は $E_T = |T - T_{\mathrm{truth}}|$ とした. ここで, 並進の値は基準とする座標系の原点位置によって異なる. この実験では, 並進の誤

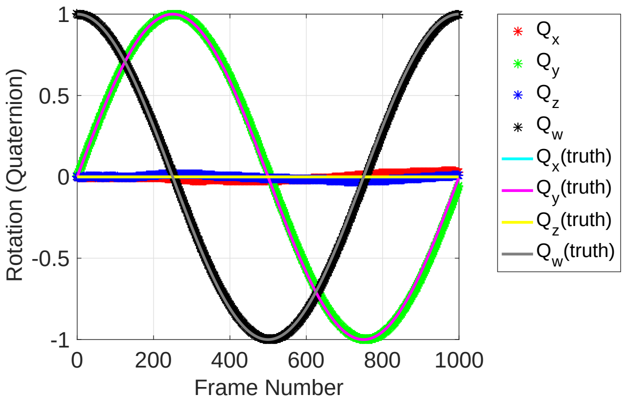

図 3: シミュレーション実験における運動開始時を基準とした絶対運動の回転成分 (Armadillo)

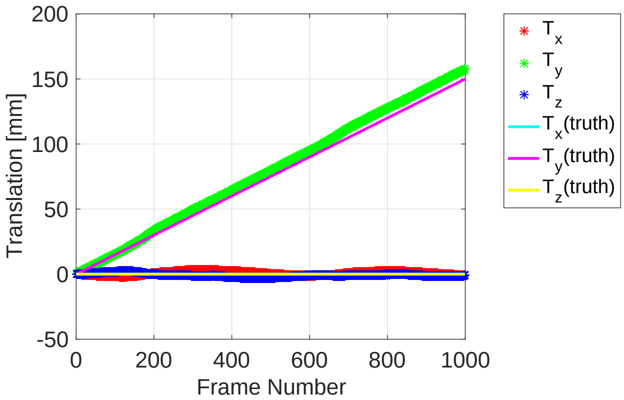

図 4: シミュレーション実験における運動開始時を基準とした絶対運動の並進成分 (Armadillo)

差を真値と比較するため, 座標系の原点を運動開始時の物体の重心とした.

このときの平均二乗誤差 (RMSE) と誤差の最大値を表 1 に示す. また, Armadillo を計測対象としたときの, 運動開始時を基準とした絶対運動を図 3, 図 4 に示す.

3.2. 実環境下の 3 次元計測データに対する適用

3.2.1. 実験条件

本稿では, 3 次元計測手法として田畑らの手法 [10] を用いる. この手法は図 5 に示すセグメントパターンを投影する構造化光法である. この手法では法線を計測

表 1: 相対運動の誤差の RMSE と最大値

	Rotation(Quaternion)		Translation[mm]	
	RMSE	Max	RMSE	Max
Bunny	0.000732	0.00347	0.113	0.736
Armadillo	0.000558	0.00203	0.0927	0.331
Lucy	0.000357	0.00152	0.0932	0.328
Buddha	0.000461	0.00178	0.109	0.374

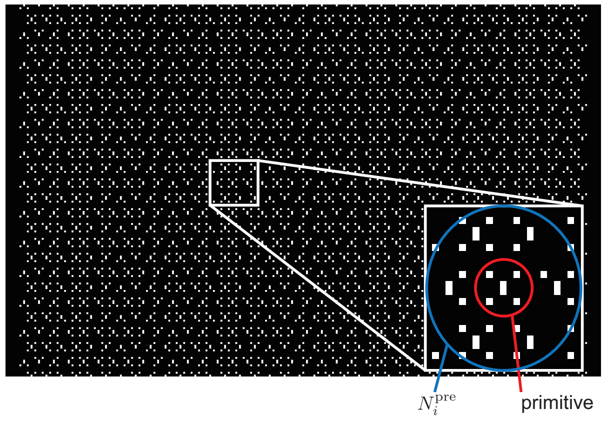

図 5: 使用した投影画像 (セグメントパターン). 図中右下に示すように, パターンを構成する各基本要素に対し, 隣接する基本要素までを近傍点群とした.

しないため, 法線推定が必要である. そこで, 近傍点群 N_i^{pre} をセグメントパターンにあわせて設定し, 法線推定を行った. セグメントパターンは基本要素と呼ばれる構成要素を持ち, ひとつの基本要素が 3〜5 個の点によって構成されている. 一方で, 2 次曲面近似を行うには最低 7 点必要であるため, ひとつの基本要素内では必要な点数に満たない. そこで, 各基本要素に対し基本要素内の点と隣接する基本要素を近傍点群 N_i^{pre} とした. また, 計算コストを抑えるため, 2 次曲面近似を全点に対してではなく基本要素内ごとに行った. 実験の様子を図 6 上段に, このときの運動推定結果の一部を図 6 下段に示す.

3.2.2. 形状統合

図 7 左に示す 3 種類の物体を計測し, 得られた Point Cloud とトラッキング情報を統合することで形状統合を行った. 図 7 上段に, 形状統合結果とモデルデータのハウスドルフ距離を色で示す. ここで, 形状統合結果として 2 秒間 (2000 フレーム) トラッキングしたデータから 25 フレームごとに抽出した Point Cloud を統合したデータを使用し, モデルデータとして Artec Group 社製の Artec Eva によって計測したデータを使用した.

また, 形状統合結果に対して, Smooth Signed Distance Surface Reconstruction[15] によってメッシュを生成した結果を図 7 下段に示す.

3.2.3. 速度

提案手法を用いて実環境で計測を行い, 処理時間を計測した. 計測対象として Stanford Bunny を用いた. 各部分の処理時間を表 2 に示す. ここで, Pre-Process は使用する点の選択・外れ値除去である.

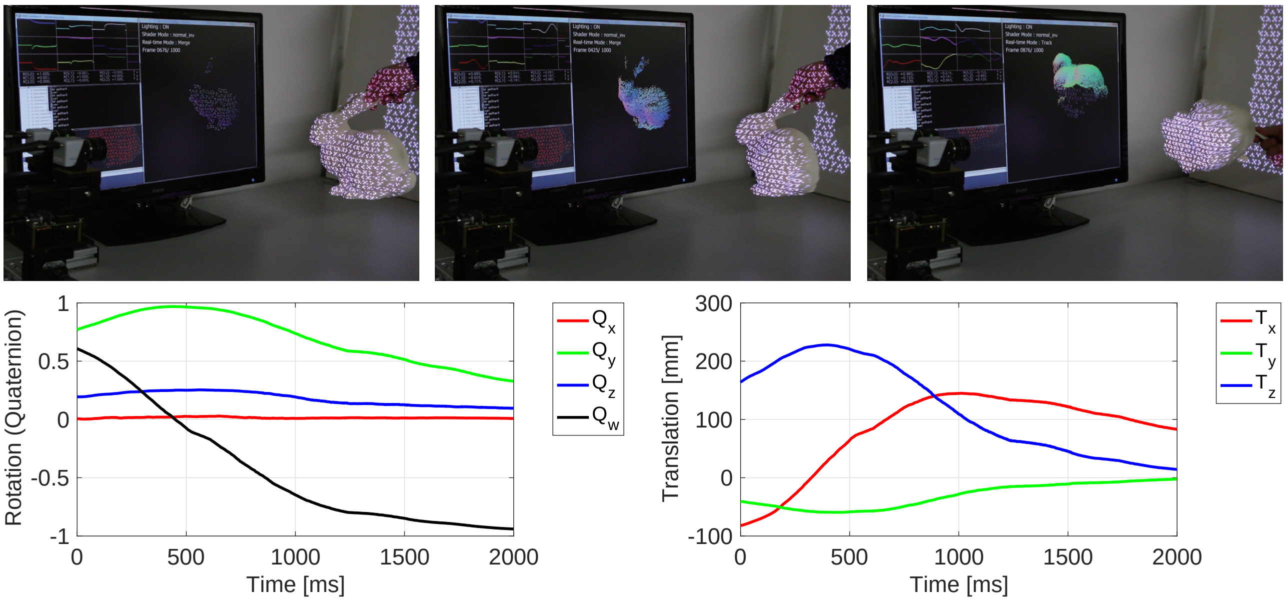

図 6: 実環境下の高速 3 次元トラッキング. 上段: 実験の様子, 下段: 推定された運動.

4. 考察

4.1. 運動推定精度

シミュレーション実験においては, フレーム間に物体座標系で 0.72 度の回転と 0.15mm の並進をしたデータに対し, 表 1 に示す誤差が確認された. 提案手法では物体表面の法線方向と計測システムの視線方向のなす角度によって対応付けている点の移動量が異なるため, 表 1 に示すように物体の形状によって誤差の大きさが異なる. また, 計測システムと物体の相対位置にも依存しており, 並進成分が大きくなっているときがあることが表 1 の並進成分の最大値から確認できる. しかし, 図 3 や図 4 に示すように, 1,000 フレームの間の相対運動の積として求めた絶対運動では, 真値と近い値が得られており, これら影響は大きくないと考えられる. 一方で, より長期間の絶対運動においては, このような微小な誤差の蓄積によって真値から大きく離れてしまうことが考えられる. その場合には, フレーム間の相対運動だけでなく, 時系列に得られる Point Cloud 全体の同時最適化などの処理が必要になると考えられる.

また, 図 6, 図 7 に示すように, 実際に高速 3 次元計測によって計測したデータに対しても, 3 次元トラッキングと形状統合を達成している. 図 7 上段に示すように, モデルに対して数 mm 程度の誤差で統合が可能であり, 図 7 下段に示すように, 計測後に得られたデータに対して既存のメッシュ生成手法による形状復元も可能であることが確認できる.

表 2: 提案手法による各処理の実行時間

Process	Time [ms]
Pre-Process	0.02
Normal Estimation	0.57
Registration	0.26
All	0.84

4.2. 処理速度

表 2 に示すように, 全体の処理が 1 ms 以下で達成されており, 高速 3 次元計測と同等のスループットが実現されている. 特に, 法線推定などの処理を除いた 3 次元トラッキングの処理は 0.26 ms 程度で達成されており, 従来よりも 10 倍程度高速化されている. また, 最も時間のかかっている処理は法線推定であった. この処理時間は 2 次曲面の近似計算によるものであり, 3 次元計測の手法や要求する速度や精度に応じて法線計算の処理を変更することによって更なる高速化が可能であると考えられる.

5. まとめ

本稿では, 構造化光法においてプロジェクタ画面上では常に同一の画素で計測されることと高速 3 次元計測における計測間の時間差分が微小であることに着目することで Point Cloud の密度が低い場合にも適用可能である高速 3 次元トラッキングを実現した. フレーム間の物体の移動量が微小であるという仮定から運動推定の処理を単純なものとすることで, 高速 3 次元計測と同程度の速度である 1,000 Hz を達成した.

参考文献

[1] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-

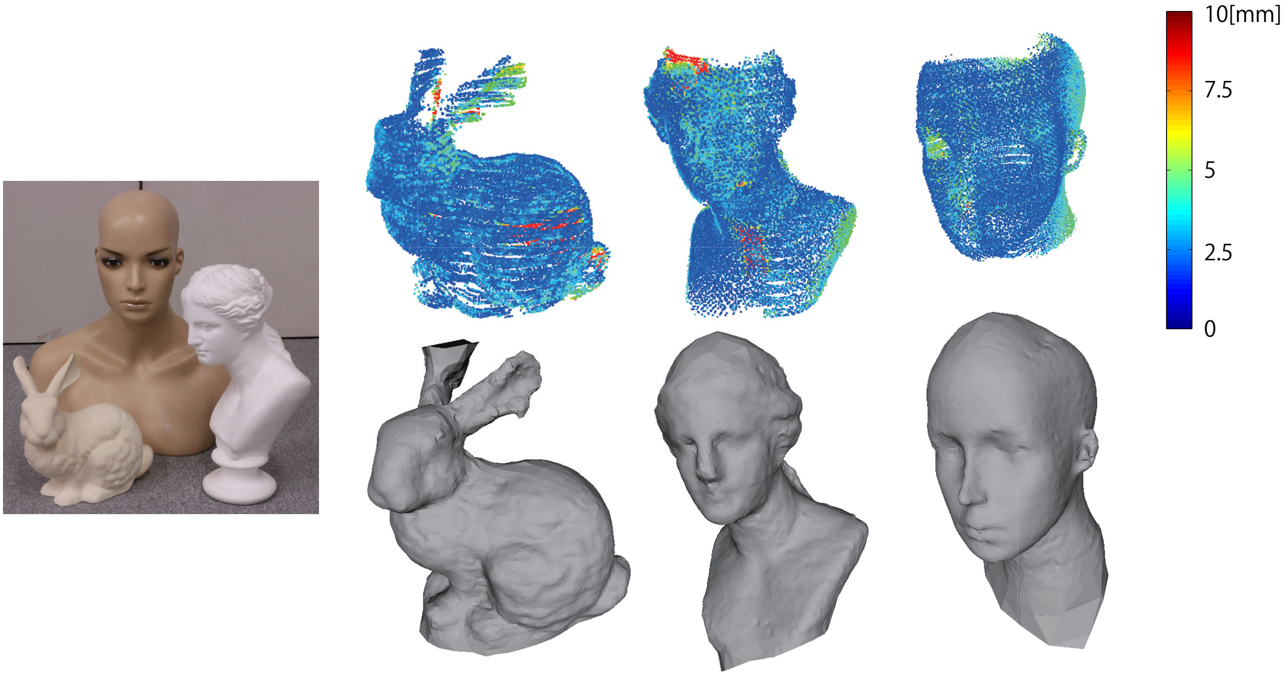

図 7: 実環境下の形状統合結果. 上段: モデルデータと計測データのハウスドルフ距離 (青 0mm-赤 10mm). 下段: 計測データに対する Smooth Signed Distance Surface Reconstruction[15] によるメッシュ生成処理結果.

perception age. *IEEE Transactions on Robotics*, Vol. 32, No. 6, pp. 1309–1332, 2016.

[2] Eric Marchand, Hideaki Uchiyama, and Fabien Spindler. Pose estimation for augmented reality: a hands-on survey. *IEEE transactions on visualization and computer graphics*, Vol. 22, No. 12, pp. 2633–2651, 2016.

[3] Paul J Besl and Neil D McKay. Method for registration of 3-D shapes. In *Robotics-DL tentative*, pp. 586–606. International Society for Optics and Photonics, 1992.

[4] Timothée Jost and Heinz Hugli. A multi-resolution ICP with heuristic closest point search for fast and robust 3D registration of range images. In *Fourth International Conference on 3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings.*, pp. 427–433. IEEE, 2003.

[5] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the ICP algorithm. In *Third International Conference on 3-D Digital Imaging and Modeling, 2001. Proceedings.*, pp. 145–152. IEEE, 2001.

[6] Joaquim Salvi, Carles Matabosch, David Fofi, and Josep Forest. A review of recent range image registration methods with accuracy evaluation. *Image and Vision computing*, Vol. 25, No. 5, pp. 578–596, 2007.

[7] Szymon Rusinkiewicz, Olaf Hall-Holt, and Marc Levoy. Real-time 3D model acquisition. *ACM Transactions on Graphics (TOG)*, Vol. 21, No. 3, pp. 438–446, 2002.

[8] Gérard Blais and Martin D. Levine. Registering multiview range data to create 3D computer objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 17, No. 8, pp. 820–824, 1995.

[9] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, Vol. 10, No. 3, pp. 145–155, 1992.

[10] 田畑智志, 野口翔平, 渡辺義浩, 石川正俊. 3 視点拘束に基づくセグメントパターン投影型高速 3 次元計測. 計測自動制御学会論文集, Vol. 52, No. 3, pp. 141–151, 2016.

[11] Yoshihiro Watanabe. High-speed optical 3D sensing and its applications. *Advanced Optical Technologies*, Vol. 5, pp. 367–376, 2016.

[12] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface Reconstruction from Unorganized Points. In *Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH ’92, pp. 71–78, New York, NY, USA, 1992. ACM.

[13] Kok-Lim Low. Linear Least-Squares Optimization for Point-to-Plane ICP Surface Registration. *Chapel Hill, University of North Carolina*, Vol. 4, , 2004.

[14] The stanford 3d scanning repository.

[15] Fatih Calakli and Gabriel Taubin. SSD: Smooth signed distance surface reconstruction. In *Computer Graphics Forum*, Vol. 30, pp. 1993–2002. Wiley Online Library, 2011.