

## (d) 情報

## GPGPU を用いた消波ブロックの高速な姿勢推定

## Fast pose estimation of a wave absorbing block with GPGPU

安部 美穂<sup>†</sup> 植野 裕司<sup>††</sup> 谷川 俊介<sup>†</sup> 玉木 徹<sup>†</sup> 一井 康二<sup>†</sup> 金田 和文<sup>†</sup>Miho Abe<sup>†</sup> Yuji Ueno<sup>††</sup> Shunsuke Tanigawa<sup>†</sup> Toru Tamaki<sup>†</sup> Koji Ichii<sup>†</sup> Kazufumi Kaneda<sup>†</sup><sup>†</sup> 広島大学大学院 工学研究科 <sup>††</sup> 広島大学 工学部

## 1 はじめに

本研究では、沿岸部にて波の被害を軽減するために設置されている消波ブロックの配置状況を把握することを目的としている。海に囲まれる日本では、波浪の沿岸災害が多い。この対策の1つとして、波の力を弱める消波ブロックが海岸に設置されている。しかし、波の影響で消波ブロックが移動・落下し、配置が変わると、本来の機能が低下して被害が生じる可能性がある。この危険性を回避するため、消波ブロックの配置を常に確認する必要がある。これを手作業で調査している膨大な時間と労力を要するため、消波ブロック配置確認の高速な自動化が望まれている。

本稿では、多数の消波ブロックの配置を確認するために、単一の消波ブロックの3次元姿勢を推定する手法について述べる。得られる形状は3次元点の集合であり、既知の消波ブロック形状のCADモデルを当てはめる必要がある(図2参照)。そこで、ここでは従来用いられているICP法を改良し、さらにGPUを用いて処理を高速化する手法について述べる。



図 1: 消波ブロック

## 2 ICP 法による姿勢推定とその問題点

データ点群とモデル形状の姿勢推定問題を解決するために、これまでに多くの手法が提案されている。特によく知られている手法はICP(Iterative Closest Point)法[1][2]である。ICPアルゴリズムは、初期状態においてデータ点群がモデル形状に大まかに位置合わせされていることを仮定し、各データ点に最も近いモデル形状上の点への対応と、姿勢パラメータの推定を、反復

計算することで、対応付けと姿勢推定を同時に解くアルゴリズムである。反復計算において誤差関数は減少するので、極所解に単調に収束することが保証されている。オリジナルのICPアルゴリズム[1]ではデータ形状の各点から最も近いモデル形状の点に対応付けている(point-to-point)のに対して、ChenとMedioni[2]が提案した手法では、各制御点(control point)を、面の法線を計算し、点と面の距離を最小化する点を求めていく(point-to-plane)。しかし、いずれの手法でも問題となるのが対応点探索の計算量である。基本的なICPでは最近傍点を全ての点に対して計算するため、点群数を等しく $N$ とした場合、対応点探索の計算量は $O(N^2)$ [3]である。

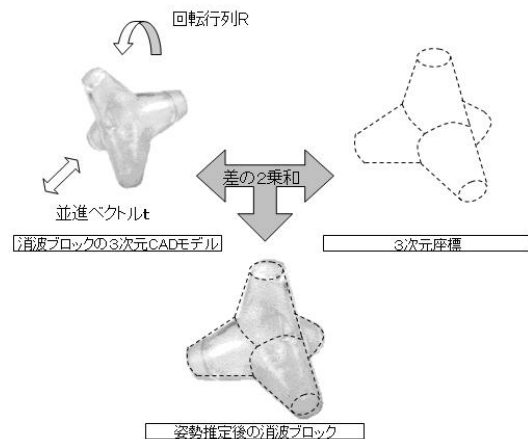


図 2: 消波ブロックの姿勢推定

## 3 ICP アルゴリズム

ここでは、従来のICPアルゴリズム[1]について述べる。まず、距離の定義を行う。Aを $N_a$ 個の点 $a_i$ からなる3次元点群

$$A = \{a_1, a_2, \dots, a_{N_a}\} \quad (1)$$

とし、3次元空間中の任意の点 $p$ と点群Aの距離を次のように定義する。

$$d(A, p) = \min_i \left\| p - a_i \right\|^2 \quad (2)$$

式 (3) より, 点群  $A$  に含まれる点の中で最も距離の近い点との距離が  $p$  と点群  $A$  との距離となる. いま,  $N_p$  個の点からなる 3 次元モデル上の点群  $P = \{P_1, P_2, \dots, P_{N_p}\}$  に対し,  $N_a$  個の点からなる 3 次元座標群  $A$  の姿勢推定を行う. まず点群  $A$  の各点  $a_i$  と  $P$  との距離  $d(A, p)$  を最近する点を  $u_i \in P$  点とすると,  $A$  と  $P$  の対応点集群  $U$  は

$$U = \{a_i, u_i\} = (A, P) \quad (3)$$

と書ける. ここで  $C$  は最近傍点を求める関数とする. 点群  $A$  に対応する点群  $U$  が求まれば, 姿勢推定の姿勢パラメータ  $3 \times 3$  回転行列  $R$ , 移動ベクトル  $t$  は次式の誤差関数を最小化することで求められる.

$$E(R, t) = \sum_{i=1}^{N_i} \|u_i - Ra_i - t\|^2 \quad (4)$$

これを誤差関数が十分小さくなるまで繰り返すことで姿勢が求まる. 以下は ICP アルゴリズムである.

(1) 点群  $A$  と  $P$  の最近点群  $U$  を求める.

$$U = C(A, P) \quad (5)$$

(2) 誤差  $E$  を最小にする  $R, t$  を求める.

$$E(R, t) \quad \min \quad (6)$$

(3) 点群  $A$  を求められた  $R, t$  で変換する.

$$a_i \rightarrow Ra_i + t, i = 1, \dots, N_a \quad (7)$$

(4) 誤差関数の減少が閾値 以下であれば反復計算を終了する. それ以外の場合は (1) に戻る.

#### 4 面に対応した ICP アルゴリズム

ICP 法が点と点で最近傍を求めることにに対し, 本研究では点と消波ブロックの各面への最近傍を求めなければならない. そこで, 以下の手法を提案する. まず, 点群  $A$  の点  $a_i$  から最も近い面  $p_i$  を求める. そして, 点  $a_i$  を平面に射影した点  $\hat{a}_i$  を求め,  $\|a_i - \hat{a}_i\|^2$  を  $a_i$  と  $p_i$  との距離とする.

しかしながら, 図 3 における点  $a_6$  のように, 3 次元点群が頂点付近にある場合は, 最近傍面を一意に決定できない. そのため, 誤差関数の計算には含めないことにする.

#### 5 CUDA による高速化

姿勢推定を高速に行うために, GPU を用いた汎用計算手法 (GPGPU) を用いる. たとえば, 式 (5) において  $i = 100$  であるとき 100 の並列計算が必要となる. このとき GPGPU [4][5] を用いることで最適化の高速な反復計算を実装することが可能である.

また, 多数の消波ブロックを同時に姿勢推定する場合も, 高速化が見込まれる. 現在は nVidia 社の CUDA を用いた GPGPU による最適化の実装について検討している.

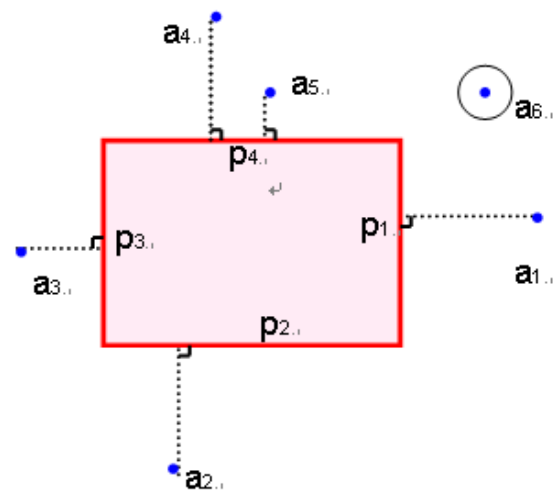


図 3: 最近傍面の決定

#### 参考文献

- [1] P. J. Besl, N. D. McKay: "A Method for Registration of 3-D Shapes", IEEE Trans. Pattern Anal. Machine Intell, Vol.14, No.2, pp.239?-256, 1992.
- [2] Y. Chen, G. Medioni: "Object modeling by registration of multiple range images", Image and Vision Computing, Vol.10, No.3, pp.145-155, 1992.
- [3] 大石岳史, 中澤篤志, 池内克史: "インデックス画像を用いた複数距離画像の高速同時位置合わせ", 電子情報通信学会論文誌 D, Vol.J89-D, No.3, pp.513-521, 2006.
- [4] NVIDIA CUDA Compute Unified Device Architecture Programming Guide 2.0 beta2.
- [5] 大島 聡史, 平澤 将一, 本多 弘樹: "既存の並列化手法を用いた GPGPU プログラミングの提案", 情報処理学会研究報告, 2007-ARC-175, Vol.2007, No.115, pp.7-10, 2007.