KIPS Trans. Softw. and Data Eng.
Vol.11, No.12 pp.525~530
ISSN: 2287-5905 (Print), ISSN: 2734-0503 (Online)

https://doi.org/10.3745/KTSDE.2022.11.12.525

The Alignment of Triangular Meshes Based on the Distance Feature Between the Centroid and Vertices

Minjeong Koo† · Sanghun Jeong†† · Ku-Jin Kim†††

ABSTRACT

Although the iterative closest point (ICP) algorithm has been widely used to align two point clouds, ICP tends to fail when the initial orientation of the two point clouds are significantly different. In this paper, when two triangular meshes A and B have significantly different initial orientations, we present an algorithm to align them. After obtaining weighted centroids for meshes A and B, respectively, vertices that are likely to correspond to each other between meshes are set as feature points using the distance from the centroid to the vertices. After rotating mesh B so that the feature points of A and B to be close each other, RMSD (root mean square deviation) is measured for the vertices of A and B. Aligned meshes are obtained by repeating the same process while changing the feature points until the RMSD is less than the reference value. Through experiments, we show that the proposed algorithm aligns the mesh even when the ICP and Go-ICP algorithms fail.

Keywords : Alignment, Mesh, Distance Feature, Centroid, ICP, Go-ICP

무게중심과 정점 간의 거리 특성을 이용한 삼각형 메쉬의 정렬

구 민 정† · 정 상 훈†† · 김 구 진†††

요 약

두 개의 점군(point cloud)을 정렬(alignment)하기 위해 현재까지 ICP(iterative closest point) 알고리즘이 널리 사용되고 있지만, ICP는 두 점군의 초기 방향이 크게 다를 경우 정렬에 실패하는 경우가 많다. 본 논문에서는 두 개의 삼각형 메쉬 A, B가 서로 크게 다른 초기 방향을 가질 때, 이들을 정렬하는 알고리즘을 제안한다. 메쉬 A, B에 대해 각각 가중치 무게중심(weighted centroid)을 구한 뒤, 무게중심으로부터 정점까지의 거리를 이용하여 메쉬 간에 서로 대응될 가능성이 있는 정점들을 특징점으로 설정한다. 설정된 특징점들이 대응될 수 있도록 메쉬 B를 회전한 뒤, A와 B의 정점들에 대해 RMSD(root mean square deviation)를 측정한다. RMSD가 기준치보다 작은 값을 가질 때까지 특징점을 변경하며 같은 과정을 되풀이하여 정렬된 결과를 얻는다. 실험을 통해 ICP 및 Go-ICP 알고리즘으로 정렬이 실패할 경우에도 제안된 알고리즘으로 정렬이 가능함을 보인다.

키워드 : 정렬, 메쉬, 거리 특성, 무게중심, ICP, Go-ICP

1. 서 론

리버스 엔지니어링(reverse engineering), 3차원 프린터 출력물의 표면 검사(surface inspection), 모션 감지 게임(motion sensing game)이나 가상현실(virtual reality) 분야 등에서는 3차원 공간에 위치한 물체 표면의 위치 정보 데이터를 점군(point cloud) 등으로 획득하여 3차원 모델로 복원하는 방법이 자주 사용된다. 3차원 모델의 복원 과정에서는 획득한 부분 점군 간의 정렬(alignment) 또는 획득한 점군과 3차원 CAD 모델과의 정렬이 필요하다.

3차원 프린팅은 현대에 들어 중요성이 증대하였고 여러 응용분야에서 자주 사용된다. 특히 개인 맞춤형 임플란트 부품과 같은 경우, 3차원 프린팅을 이용한 제작의 중요성이 더욱 커지고 있다. 3차원 프린팅으로 임플란트 부품을 출력할 경우, 주로 사용되는 재료는 티타늄과 같은 금속이며, 이 경우 베드 파우더 퓨전(bed powder fusion) 방식이 일반적으로 사용된다. 베드 파우더 퓨전 방식으로 출력한 경우 크랙(crack), 블롭(blob), 거칠기(roughness) 등의 표면 결함이 발생할 수 있다. 또한, 제작된 임플란트에 대해 열처리(heat treatment)를 수행하는 과정에서도 임플란트가 휘거나 뒤틀릴 수 있고 크랙이나 블롭 등의 결함이 추가적으로 발생할 수 있다. 3차원 프린터 출력물의 표면 결함을 발견하기 위해서

※ 이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구(NRF-2020R1A2C1008912)임.
※ 이 논문은 2022년 한국정보처리학회 ASK 2022의 “거리 특성을 이용한 다면체 간의 정렬”의 제목으로 발표된 논문을 확장한 것임.
† 준 회 원 : 경북대학교 컴퓨터학부 석사과정
†† 비 회 원 : 경북대학교 소프트웨어기술연구소 위촉연구원
††† 정 회 원 : 경북대학교 컴퓨터학부 교수
Manuscript Received : July 29, 2022
Accepted : August 24, 2022
* Corresponding Author : Ku-Jin Kim(kujinkim@gmail.com)

는 입력으로 사용된 CAD 모델과 형상을 비교해야 하며, 이때 CAD 모델과 출력물의 정렬이 필요하다.

생물학 분야에서는 Cryo-electron microscopy (EM)을 이용하여 단백질 분자의 구조를 표현하는 기법이 자주 사용되고 있다[1,2]. Cryo-EM을 이용하면 대상 분자에 속한 원자들의 밀도가 3차원 맵 (map)으로 저장되며, 밀도 맵(density map)인 Cryo-EM 맵으로 분자를 표현한다.

3차원 밀도 맵은 3차원 공간을 그리드(grid)로 균일하게 분할하여 각 그리드 정점마다 밀도 값이 주어진 형태로 가정할 수 있다. Cryo-EM 맵은 3차원 밀도 맵으로, https://www.ebi.ac.uk/pdbe/emdb에서 데이터베이스가 운영되고 있다.

Cryo-EM 맵에 대한 가시화를 위해서는 지정된 윤곽 수준(contour level)에 따라 동일한 밀도 값을 갖는 위치 정보를 곡면으로 추출하는 작업이 자주 수행되며, 마칭 큐브 알고리즘[3,4] 또는 듀얼 컨투어링(dual contouring) 알고리즘[5] 등을 적용하여 등밀도 곡면을 추출할 수 있다. 동일한 단백질 분자에 대해 그리드 크기를 달리하여 서로 다른 해상도로 Cryo-EM 맵을 구성한 경우, 전체적인 형상은 유사하지만 지역적으로는 형상이 서로 다른 등밀도 곡면들이 추출된다. 이런 경우, 서로 다른 해상도의 등밀도 곡면에 대한 정렬이 필요하다.

점군에 대한 정렬을 위해 ICP (iterative closest point) 알고리즘[6-8]이 널리 사용되고 있다. ICP 알고리즘은 1992년에 제안된 이후 현재까지 두 점군 간의 정렬을 위해 자주 사용되며, 두 점군이 표현하는 형태가 완전히 서로 겹치는 경우뿐만 아니라, 서로 겹치는 부분이 일부 영역으로 제한된 경우에 대한 정렬 방법도 연구되어 여러 응용 분야에서 활용된다.

두 점군 $X=\{x_1,x_2,...,x_N\}\in\mathbb{R}^3$와 $Y=\{y_1,y_2,...,y_M\}\in\mathbb{R}^3$를 정렬하기 위해 ICP 알고리즘은 Equation (1)에 제시된 오차 $E(R,T)$를 최소화하는 회전 행렬(rotation matrix) $R\in SO(3)$과 이동 벡터(translation vector) $T\in\mathbb{R}^3$를 발견하는 것을 목적으로 R과 T를 반복적으로 수정한다.

$$E(R,T)=\frac{1}{N}\sum_{i=1}^{N}\| Rx_i+T-y_{j^*}\|^2 \tag{1}$$

Equation (1)에서 y_{j^*}는 Rx_i+T로부터 최소 거리에 있는 Y의 점이며 $j^*=\underset{j\in\{1,2,...,M\}}{argmin}\| Rx_i+T-y_j\|$ 이다.

ICP 알고리즘은 정렬 결과의 정확성이 높지만, 성공적인 정렬을 위해서는 점군의 초기 위치나 방향이 서로 유사해야 한다. 그렇지 않을 경우에는 정렬에 실패하기 쉬우므로, 이런 문제점을 개선하기 위해 [9-11] 등의 연구가 수행되었다. [12,13]에서는 최근까지 개발된 다양한 정렬 알고리즘을 제시한다.

Du 등[9]은 2차원 점군을 정렬하기 위해 ICP의 목적 함수(objective function)에 무게중심과 정점 간의 거리 식을 추가한 알고리즘을 제안하였다. Yang 등[10]은 ICP 알고리즘이 목적함수의 전역적인 최소값을 찾는 대신 지역적인 최소값을 발견하는 문제점을 해결하기 위해 Go-ICP 알고리즘을 제안하였다.

본 논문에서는 의료용 임플란트의 한 종류인 bone plate의 CAD 모델과 이에 대한 3차원 출력물, 동일한 단백질 분자에 대한 해상도가 서로 다른 Cryo-EM 맵에서 추출한 모델과 같이 동일한 형상이 아니지만, 형상의 유사도가 높은 두 모델에 대해 효율적으로 정확도가 높은 정렬을 수행하는 알고리즘을 제안한다.

Koo 등[14]은 다면체에 대해 무게 중심(centroid)를 이용하여 특징점을 선정하고, 소수의 특징점을 이용하여 다면체를 정렬하는 아이디어를 제안하였다. 본 논문에서는 이 연구를 확장하여 특징점을 선택하는 방법을 제시하고, 반복하여 특징점을 선정함으로써 두 개의 삼각형 메쉬(triangular mesh)를 정렬하는 알고리즘을 제시한다. 또한 실험을 통해 제안된 알고리즘이 적용된 예를 보이고, ICP 및 Go-ICP를 적용하여 정렬한 결과와 비교한다.

본 논문의 구성은 다음과 같다. 2절에서는 제안된 알고리즘에 대해 설명한다. 3절에서는 실험 결과를 보이고 4절에서 결론을 맺는다.

2. 무게중심으로부터의 거리 기반의 정렬 알고리즘

정렬 알고리즘에 대한 입력은 3차원 삼각형 메쉬인 A와 B로 주어진다. 각 메쉬는 정점(vertex)의 집합과 면(face)의 집합으로 정의되며, A와 B를 각각 Equation (2)와 같이 표현한다.

$$\begin{aligned}&A=\{V,F\}\text{ with the vertex set }V=\{v_1,v_2,...,v_n\}\in\mathbb{R}^3\\&\quad\text{and the face set }F=\{f_1,f_2,...,f_s\}\\&B=\{W,G\}\text{ with the vertex set }W=\{w_1,w_2,...,w_m\}\in\mathbb{R}^3\\&\quad\text{and the face set }G=\{g_1,g_2,...,g_t\}\end{aligned} \tag{2}$$

제안하는 정렬 방법은 각 메쉬에 대해 3개의 정점을 특징점으로 선택한 뒤, 서로 대응하는 특징점들이 일치하거나 근거리에 위치하도록 메쉬 A에 가까운 거리로 메쉬 B를 회전하는 과정을 반복한다.

반복 횟수에 대한 제한 기준치로 값 α를 입력으로 사용한다. 반복 과정 중에 회전된 B와 A 간의 RMSD[15]가 δ 이하이면 정렬에 성공한 것으로 판정하여 알고리즘을 종료한다.

RMSD는 B의 정점 w_i에 대해 최소 거리에 있는 A의 정점이 $C(w_i)$일 때, Equation (3)과 같이 계산한다.

$$RMSD(A,B)=\sqrt{\frac{\sum_{j=1}^{m}(w_j-C(w_j))^2}{m}} \tag{3}$$

RMSD는 메쉬 B의 정점에서 메쉬 A에 있는 가장 가까운 점까지의 거리를 이용하여 계산되고, 거리값은 메쉬의 크기에 따라 상대적으로 정렬 상태의 정확도를 나타낸다. 이에 따라 본 논문에서는 RMSD에 대한 허용 오차를 메쉬의 크기에

비례하여 결정한다. 각 메쉬에 대한 AABB (axis aligned bounding box)를 구한 뒤, 메쉬 A와 B의 AABB의 최대 변 길이 LEN_A와 LEN_B를 이용하여 Equation (4)와 같이 δ 를 결정하였다.

$$\delta = (LEN_A + LEN_B)/200 \tag{4}$$

응용 분야에 따라 시간이 걸려도 더 정확한 정렬 결과가 필요한 경우에는 δ를 더 작은 값으로 줄여서 사용할 수 있고, 정확성보다 정렬에 소요되는 계산 속도가 중요한 경우에는 δ 를 더 큰 값으로 결정할 수 있다.

제안된 알고리즘의 개요를 Fig. 1에서 보인다.

```
Input: A = {V, F} and B = {W, G} // meshes
  δ // threshold for the RMSD between two meshes
  α // threshold for the iteration
Step 1: // Compute the weighted centroid
  C_A := the weighted centroid of A;
  C_B := the weighted centroid of B;
Step 2: // Translate A and B for C_A and C_B to be located
          // at origin O
  for each v_i ∈ V do v_i := v_i – C_A;
  for each w_j ∈ W do w_j := w_j – C_B ;
Step 3: // Decide the feature points in A
  F_A := the vertex v* such as ||v* – O|| = MAX_{1≤ i ≤n}(||v_i – O||);
  N_A := the vertex v* such as ||v* – O|| = MIN_{1≤ i ≤n}(||v_i – O||);
Step 4: //Sort the vertices in B with respect to its distance to O
  (b_(1), b_(2), ..., b_(m)) := the sequence of the indices of vertices of B,
  where ||w_b(j) – O|| ≥ ||w_b(j+1) – O||, 1 ≤ j < m ;
Step 5: // Accumulatively rotate B according to the
          // updated feature points
  B* = B;
  MIN_RMSD := RMSD (A, B*);
  x := 1;
  y := m;
  count := 0;
  while (count < α) do begin
    B := rotated B with the rotation axis (w_b(x) – O) X (F_A – O)
         for the line segment Ow_b(x) to be overlapped with OF_A ;
    B := rotated B with the rotation axis (F_A – O) with the angle
         acos((N_A – O)•(w_b(m) – O)/(|| N_A – O || || w_b(m) – O ||));
    if (||F_A – w_b(x)|| ≤ 10δ and ||N_A – w_b(y)|| ≤ 10δ) then begin
      if (RMSD(A, B) < δ) then Output B and Exit;
        if (MIN_RMSD > RMSD (A, B)) then begin
          MIN_RMSD := RMSD (A, B);
          B* = B;
        end
      end
    if ||F_A – w_b(x)|| > ||N_A – w_b(y)|| then x := x + 1;
    else y := y – 1;
    count++;
  end
Step 6: Output B*;
```

Fig. 1. The Suggested Algorithm

2.1 가중치 무게중심의 계산 및 메쉬의 평행이동

알고리즘의 step1, 2에서는 메쉬 A, B의 가중치 무게중심에 따라 각 무게중심이 원점으로 이동하도록 A, B를 각각 평행이동한다. 메쉬 A, B의 가중치 무게중심(weighted centroid) C_A, C_B를 계산하는데, 이때 가중치는 각 정점의 면적(area)을 이용하여 부여한다. Meyer 등[16]은 미분 기하(differential geometry)를 기반으로 다각형 메쉬에서 정점의 면적을 근사하는 방법을 소개하였다.

제안된 알고리즘에서는 메쉬 A에 속한 각각의 정점 $v_i(1 \le i \le n)$에 대한 면적을 계산한 뒤, A의 가중치 무게중심을 아래와 같이 계산하였다. v_i와 인접한 삼각면(triangular face)들의 면적을 합산한 후 1/3을 취한 값을 v_i의 면적이라 한다. 정점 v_i의 면적이 a_i이고, 모든 정점의 면적의 합이 $a = \sum_{i=1}^{n} a_i$ 일 때, 무게중심은 Equation (5)와 같이 계산되며, 메쉬 B에 대한 가중치 무게중심 C_B도 동일한 방법으로 계산된다.

$$C_A = (\sum_{i=1}^{n} a_i v_i)/(an) \tag{5}$$

무게중심이 좌표계의 원점으로 이동하도록 메쉬 A의 각 정점 $v_i(1 \le i \le n)$에 대해 $v_i - C_A$를 적용하고, 메쉬 B의 각 정점 $w_j(1 \le j \le m)$에 대해 $w_j - C_B$를 적용한다. 이를 수행한 뒤에는 메쉬 A, B의 무게중심이 원점 $O = (0,0,0)$으로 고정된다.

2.2 특징점의 결정 및 메쉬 B의 회전

알고리즘의 step 3에서는 정렬을 위해 메쉬 A에서 무게중심 O 와 O 로부터 거리가 가장 먼 정점 F_A, 거리가 가장 가까운 정점 N_A를 특징점으로 정한다. 메쉬 A의 특징점은 이후 정렬 과정에서 변화 없이 고정되며, 메쉬 B의 특징점을 수정하면서 메쉬 A와 가장 오차가 적은 정렬 방향을 찾는 것이 제안된 알고리즘의 수행 방식이다.

알고리즘의 step 4에서는 메쉬 B의 특징점을 효율적으로 결정하기 위한 선행 작업을 수행한다. 메쉬 B의 모든 정점을 O로부터 거리가 먼 것부터 가까운 것의 순서로 정렬한 정점의 인덱스(index) 열 $(b_{(1)},b_{(2)},...,b_{(m)})$을 구한다.

Step 5에서는 메쉬 B의 특징점으로 무게중심 O 와 O 로부터 거리가 가장 먼 정점 $w_{b(1)}$과 가장 가까운 정점 $w_{b(m)}$을 선택한다. 메쉬 B의 특징점 $w_{b(1)}$이 A의 특징점 F_A와 가까워지는 방향으로 B를 회전한다. 이를 위해 $(w_{b(1)} - O) \times (F_A - O)$를 회전축으로 정하며, 회전각 θ를 Equation (6)과 같이 계산한다(Fig. 2참조).

$$\theta = \text{acos} \frac{(F_A - O) \cdot (w_{b(1)} - O)}{\| F_A - O \| \ \| w_{b(1)} - O \|} \tag{6}$$

회전된 메쉬 B에 대해, A의 특징점 N_A와 B의 특징점

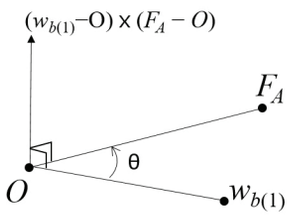

Fig. 2. The Rotation Axis and Angle for $w_{b(1)}$ and F_A

$w_{b(m)}$을 고려하여 두번째 회전을 수행한다. 이때 회전축은 $(F_A - O)$로 정하고, 회전각은 Equation (7)과 같이 계산한다.

$$\theta = \text{acos} \frac{(N_A - O) \bullet (w_{b(m)} - O)}{\| N_A - O \| \| w_{b(m)} - O \|} \tag{7}$$

두 번의 회전을 적용한 B에 대해 $\| F_A - w_{b(x)} \|$ 또는 $\| N_A - w_{b(y)} \|$가 10δ보다 큰 경우에는 대응되는 특징점들이 서로 일치하지 않는 것으로 판단하며, 이 경우에는 RMSD를 계산하지 않고 B의 특징점을 수정하는 과정으로 넘어간다. 그렇지 않은 경우에는 B에 대해 A와 RMSD를 계산하여 RMSD가 δ보다 작으면 정렬이 완료된 것으로 판단한다.

정렬이 완료되지 않은 경우, A와 B에서 서로 대응되는 특징점의 쌍이 갖는 거리를 계산한 뒤 거리가 더 먼 쌍에 대해서는 B의 해당 특징점을 수정하여 다시 앞의 과정을 반복한다. 즉, $\| F_A - w_{b(1)} \|$과 $\| N_A - w_{b(m)} \|$을 비교하여, $\| F_A - w_{b(1)} \|$의 거리가 더 짧은 경우에는 F_A와 $w_{b(1)}$ 쌍이 N_A와 $w_{b(m)}$ 쌍에 비해 정렬이 성공적이라고 할 수 있으므로, $w_{b(1)}$은 수정하지 않고 그대로 유지한다. 그리고, 다른 쌍 N_A와 $w_{b(m)}$ 의 정렬 결과를 개선하기 위해 특징점으로 $w_{b(m)}$를 대신하여 $w_{b(m-1)}$을 특징점으로 수정한다. $\| N_A - w_{b(m)} \|$의 거리가 더 짧은 경우에는 반대로 특징점 $w_{b(1)}$을 $w_{b(2)}$로 수정한다. 수정된 특징점에 대해 다시 B를 회전하는 과정을 반복한다.

메쉬 A와 B의 RMSD가 δ보다 작은 경우가 발생하면 성공적으로 정렬이 수행되었다고 판단할 수 있으나, 정해진 반복 회수 α까지 반복을 수행해도 정렬에 성공하지 않는 경우가 존재한다. 이 경우에는 반복 과정에서 계산한 RMSD 중 최소값에 해당하는 메쉬 A와 B의 정보를 출력하며 알고리즘의 수행을 마친다.

Fig. 3에서는 EMD1815의 고해상도 메쉬를 A, 저해상도 메쉬를 B로 입력하여 제안된 알고리즘의 수행 과정에 따라 메쉬 B의 방향이 변화하는 예를 RMSD와 함께 보인다.

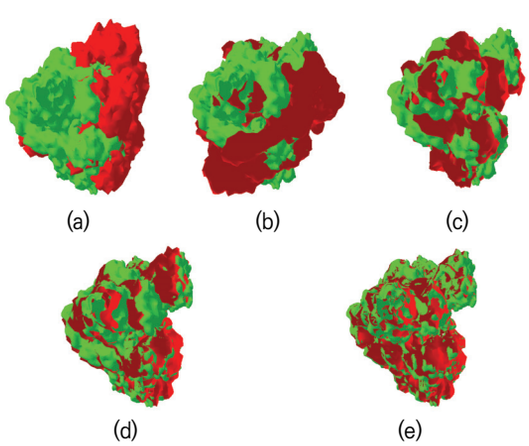

Fig. 3. The High Resolution Mesh (Green) and the Low Resolution Mesh (Red) of EMD1815: (a) the Initial State, (b) the 1st Iteration (RMSD: 6161.465660), (c) the 2nd Iteration (RMSD: 716.359659) (d) the 7th Iteration (RMSD: 11.546572). and (e) the 64th (Final) Aligned Result (RMSD: 0.261572).

3. 실험 결과

본 절에서는 제안된 정렬 방법의 성능 평가를 위해 진행한 실험 결과를 소개한다. 실험은 AMD Ryzen 5 2600 Six-Core Processor CPU, 32GB RAM, NVIDIA GeForce RTX 2060 SUPER GPU가 장착된 데스크탑 PC에서 수행하였다.

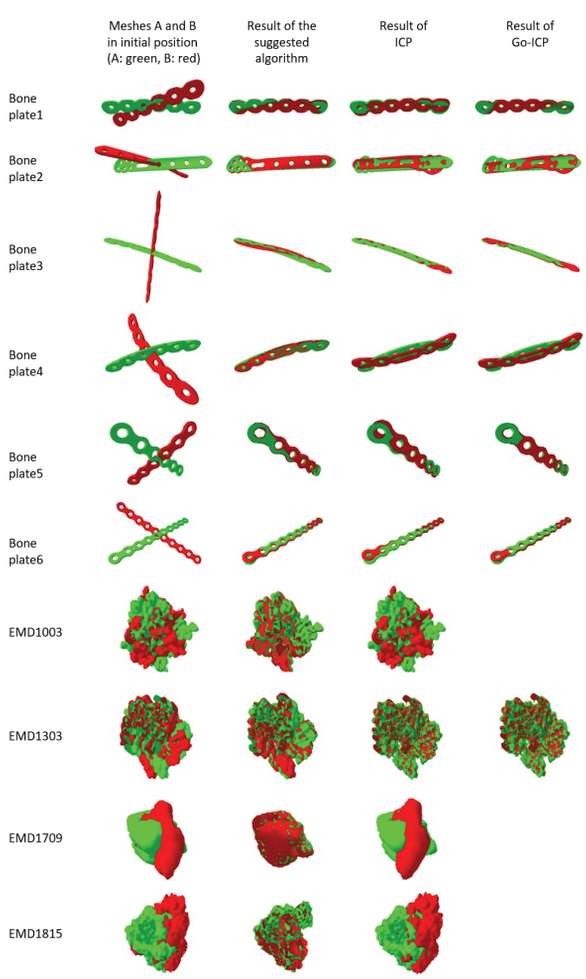

Fig. 4. The Initial Position and the Aligned Result of the Suggested Algorithm, ICP Algorithm, and Go-ICP Algorithm for the Meshes.

Table 1. The Experimental Results of the Suggested Algorithm, ICP, and Go-ICP. The Euclidean Fitness Epsilon for ICP is 0.01, and the Mean Squared Error (MSE) Convergence Threshold for Go-ICP is 0.01. The Iteration Threshold for the Suggested Algorithm and ICP is 100.

Dataset	The number of vertices		The suggested algorithm			ICP		Go-ICP	
	Mesh A (CAD model for bone plates 1-6 and high resolution models for EMD)	Mesh B (Scan model for bone plates 1-6 and low resolution models for EMD)	δ	RMSD	Computation time(s)	RMSD	Computation time(s)	RMSD	Computation time(s)
Bone plate1	47,010	47,455	0.049522	0.019370	21.079	0.0869127	8.151	0.005954	150.455
Bone plate2	61,933	62,184	0.917643	0.345508	14.044	48.706848	20.999	18.339924	108.898
Bone plate3	64,172	64,717	1.040153	126.009921	178.310	7.863900	15.722	0.018506	72.345
Bone plate4	55,738	76,109	0.746893	0.011441	2.971	231.290955	25.514	230.648926	151.828
Bone plate5	18,018	18,027	0.321687	0.249331	1.825	0.400073	3.383	0.223563	52.486
Bone plate6	26,396	26,441	0.559950	0.549446	20.382	0.659200	4.079	0.394896	53.208
EMD1003	80,454	19,204	0.877014	0.254518	44.463	3373.578857	2.027	-	-
EMD1303	136,030	32,682	1.076592	15.057670	82.173	0.017847	18.598	0.017838	109.804
EMD1709	12,954	3,116	0.497445	0.150707	0.345	1115.324341	0.754	-	-
EMD1815	43,054	10,064	0.829870	0.261572	16.893	14423.14602	2.562	-	-

실험에 대한 입력으로 삼각형 메쉬로 주어진 CAD 모델과 그 모델의 3D 프린터 출력물을 스캔하여 복원한 스캔 모델, 그리고 Cryo-EM 맵에 대해 구성한 서로 다른 해상도의 메쉬를 사용하였다. CAD와 스캔 모델의 정렬에서는 CAD 모델을 A로 사용하고 스캔 모델을 B로 사용하였다. Cryo-EM 맵에서는 고해상도 메쉬를 A로 사용하고 저해상도 메쉬를 B로 사용하였다(Table 1참조). Fig. 4에서는 입력 메쉬 및 제안된 알고리즘, ICP 알고리즘, Go-ICP 알고리즘을 적용하여 정렬이 적용된 후의 메쉬를 렌더링한 결과를 보인다.

ICP 알고리즘은 PCL version 1.11.1 [17]을 이용하여 구현되었으며, 메쉬 B에 속한 정점으로부터 최소거리에 있는 메쉬 A의 정점을 구하는 과정에서도 동일한 라이브러리(library)에서 제공하는 kd-tree가 사용되었다. Go-ICP는 저자가 제공하는 코드 [18]를 이용하여 수행하였다. 메쉬의 렌더링은 OpenGL version 4.6를 이용하여 구현하였다.

Table 1에서는 제안된 알고리즘과 ICP 알고리즘, Go-ICP 알고리즘에 대한 실험 결과를 보인다.

6개의 Bone plate 모델에 대해 실험한 결과 Bone plate3의 경우 제안된 알고리즘은 정렬에 실패하였으나, ICP와 Go-ICP는 빠른 시간 내에 정렬에 성공하였다. 나머지 경우는 제안된 알고리즘을 적용할 때, 정해진 기준치 내의 RMSD를 갖는 정렬 결과를 얻었으며 Bone plate6을 제외한 나머지 모델에 대해 Go-ICP 보다 빠른 시간 내에 정렬을 완료하였다.

4개의 EMD 모델에 대해 실험한 결과 모델 EMD1003, EMD1709, EMD1815의 경우 Go-ICP 알고리즘을 1시간 이상 수행해도 종료되지 않아서 해당 알고리즘의 계산시간과 RMSD가 제시되지 않았다. ICP 및 Go-ICP 알고리즘은 EMD1303 모델의 경우에만 정렬에 성공하고 나머지 3가지 경우에는 정렬에 실패하였다. 이에 비교하여, 제안된 알고리즘은 EMD1303에 대해서는 기준치 이하의 RMSD를 갖는 정렬 결과를 얻지 못 하여 정렬에 실패했지만, 나머지 3가지 경우에는 모두 정렬에 성공하였다.

4. 결 론

본 논문에서는 초기 방향이 크게 다르고 형상이 유사한 두 개의 삼각형 메쉬 A, B를 정렬하는 알고리즘을 제안하였다. 메쉬 A, B에서 각각 구한 가중치 무게중심을 포함하여, 무게중심으로부터 가장 먼 거리와 가까운 거리에 있는 정점을 특징점으로 선택하여 정렬한 뒤, 정렬 결과에 따라 특징점을 수정하며 정렬을 반복한다. 실험을 통해 ICP 알고리즘이나 Go-ICP 알고리즘으로 정렬이 실패한 경우에 제안된 알고리즘으로 정렬이 가능함을 보였다.

제안된 알고리즘은 메쉬들의 무게중심을 원점으로 이동한 후, 메쉬의 회전만으로 정렬을 수행한다. 형태가 유사한 메쉬에 대해 가중치 무게중심을 구했으므로, 한 번의 이동 후에 회전만 적용하여도 좋은 정렬 결과를 얻을 수 있었다. 그러나, 정렬한 후의 상태에서 모델 각각의 가중치 무게중심 간의 거리가 먼 경우에는 제안된 알고리즘으로 정렬에 성공하기가 어려울 것으로 예측된다. 향후에는 RMSD를 정점 간의 최소거리보다 정밀하게 계산하는 연구와 메쉬의 회전뿐 아니라 평행이동도 병행되어 더욱 정확한 정렬을 수행하는 연구를 수행하고자 한다.

References

[1] P. B. Rosenthal, "Interpreting the cryo-EM map," *IUCrj*, Vol.6, pp.3-4, 2019.

[2] P. V. Afonine, et al., "New tools for the analysis and validation of cryo-EM maps and atomic models," *Acta Crystallographica Section D: Struct Biology*, Vol.74, No.9, pp.814-840, 2018.
[3] W. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," *Computer Graphics (SIGGRAPH 87 Proceedings)*, Vol.21, No.4, pp. 163-170, 1987.
[4] T. S. Newman and H. Yi, "A survey of the marching cubes algorithm," *Computers & Graphics*, Vol.30, No.5, pp.854-879, 2006.
[5] T. Ju, F. Losasso, S. Schaefer, and J. Warren, "Dual contouring of hermite data," *ACM Transactions on Graphics*, Vol.21, No.3, pp.339-346, 2002.
[6] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.14, No.2, pp.239-256, 1992.
[7] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image and Vision Computing*, Vol.10, No.3, pp.145-155, 1992.
[8] L. Cheng, S. Chen, X. Liu, H. Xu, Y. Wu, M. Li, and Y. Chen, "Registration of laser scanning point clouds: A review," *Sensors*, Vol.18, No.5, pp.1641, 2018.
[9] S. Du, Y. Xu, T. Wan, H. Hu, S. Zhang, G. Xu, and Z. Zhang, "Robust iterative closest point algorithm based on global reference point for rotation invariant registration," *PLoS ONE*, Vol.12, No.11, pp.e0188039, 2017.
[10] J. Yang, H. Li, D. Campbell, and Y. Jia. "Go-ICP: A globally optimal solution to 3D ICP point-set registration," *IEEE transactions on pattern analysis and machine intelligence*, Vol.38, No.11, pp.2241-2254, 2015.
[11] Q. -Y. Zhou, J. Park, and V. Koltun, "Fast global registration," in *European Conference on Computer Vision*, pp. 766-782, 2016.
[12] H. Zhu, B. Guo, K. Zou, Y. Li, K.-V. Yuen, L. Mihaylova, and H. Leung, "A review of point set registration: From pairwise registration to groupwise registration," *Sensors*, Vol.19, No.5, pp.1191, 2019.
[13] L. Li, R. Wang, and X. Zhang, "A tutorial review on point cloud registrations: principle, classification, comparison, and technology challenges," *Mathematical Problems in Engineering*, Vol.2021, Article ID. 9953910, 2021.
[14] M. Koo, S. Jeong, and K.-J. Kim, "Alignment of Polyhedra Based on Distance Feature," in *Proceedings of the Annual Spring Conference of Korea Information Processing Society Conference (KIPS) 2022*, Vol.29, No.1, pp.32, 2022.
[15] O. Carugo and S. Pongor, "A normalized root-mean-square distance for comparing protein three-dimensional structures," *Protein Science*, Vol.10, No.7, pp.1470-1473, 2001.
[16] M. Meyer, M. Desbrun, P. Schroder, and A. H. Barr, "Discrete differential-geometry operators for triangulated 2-manifolds," in *Visualization and Mathematics III*, pp. 35-57. Springer, Berlin, Heidelberg, 2003.
[17] Point Cloud Library, PCL 1.11.1 [Internet], https://pointclouds.org
[18] Go-ICP Source Code [Internet], http://jlyang.org/go-icp/

구 민 정

https://orcid.org/0000-0002-4040-7548
e-mail : koomin1229@gmail.com
2020년 경북대학교 컴퓨터학부(학사)
2020년 ~ 현 재 경북대학교 컴퓨터학부 석사과정
관심분야 : 컴퓨터 그래픽스, 계산생물학

정 상 훈

https://orcid.org/0000-0002-8483-7232
e-mail : devY30000@gmail.com
2016년 경북대학교 농업토목공학과(학사)
2018년 경북대학교 컴퓨터학부(석사)
2019년 ~ 현 재 Y30000 대표
2021년 ~ 현 재 경북대학교 소프트웨어기술연구소 위촉연구원
관심분야 : 컴퓨터 그래픽스, 게임프로그래밍, 3D 애니메이션

김 구 진

https://orcid.org/0000-0001-8743-4650
e-mail : kujinkim@gmail.com
1990년 이화여자대학교 전자계산학과(학사)
1992년 KAIST 전자계산학과(석사)
1998년 POSTECH 컴퓨터공학과(박사)
1998년 ~ 2000년 Dept. of Computer Sciences, Purdue University, PostDoc.
2000년 ~ 2002년 아주대학교 정보통신전문대학원 BK21 조교수
2002년 ~ 2003년 Dept. of Mathematics and Computer Science, University of Missouri-St. Louis, Visiting Assistant Professor
2004년 ~ 현 재 경북대학교 컴퓨터학부 교수
관심분야 : 컴퓨터 그래픽스, 기하모델링, 계산생물학