Point Cloud의 시각화를 위한 Mesh Reconstruction

□ **이승용, 한종기** / 세종대학교

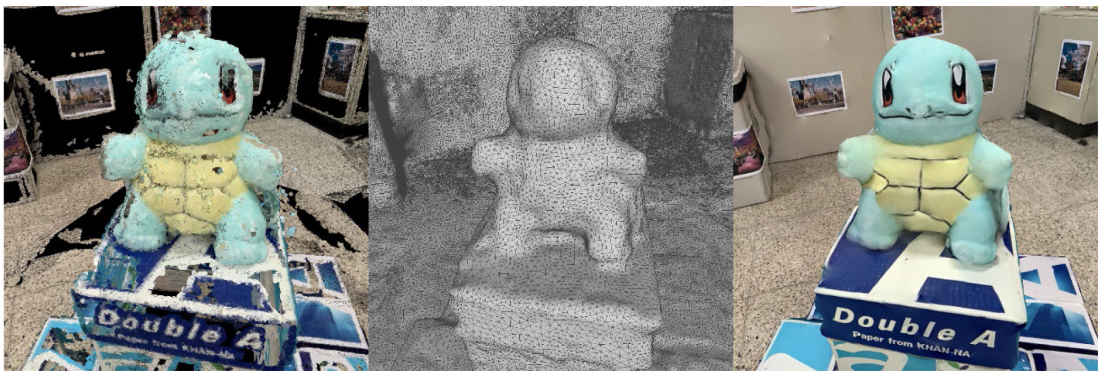
요 약

3D reconstruction은 현실 세계의 객체와 장면을 2D 이미지를 활용하여 디지털 3D 모델로 구현하는 과정이다. 이로써 디지털 세계에서 현실 세계를 더 생생하게 나타낼 수 있다. 그중에서 mesh reconstruction은 point cloud의 빈 공간을 채우고 표면을 정의하는 과정으로, 텍스처 적용 및 상호작용에 필수적인 과정이다. 본 기고문에서는 전통적인 방법 중 Delaunay tetrahedralization과 graph-cut을 이용한 방법을 설명하고, 학습 기반 방법 중 NeuralRecon을 설명한다. 마지막으론 향후 mesh reconstruction의 연구 방향과 기술 발전을 위해 이루어야 할 목표를 제시한다.

Ⅰ. 서 론

최근 가상 현실(VR), 증강 현실(AR) 및 메타버스가 주목을 받으며 3D reconstruction 기술이 함께 주목을 받고 연구되어 왔다. 3D reconstruction은 실제 세계의 객체나 장면을 촬영한 다수의 2D 이미지로부터 디지털 3D 모델을 복원하는 과정을 말하며, 이를 통해 현실 세계를 더욱 생동감 있게 디지털 세계에서 표현할 수 있다.

3D reconstruction은 크게 다수의 2D 이미지로부터 카메라 파라미터를 구하는 SfM(Structure from Motion)[1]과 각 영상의 depth map을 구하고 mesh를 형성하는 MVS(Multi View Stereo)[2]로 나눌 수 있다. SfM을 통해 2D 이미지들의 카메라 파라미터를 구한다면 각 이미지의 depth map을 구할 수 있고 이를 통해 3차원 공간상에 dense point cloud를 형성할 수 있다. Point cloud는 3차원 공간상의 좌표와 RGB 색상 정보를 포함하고 있다. 이때 point cloud를 이루는 point들 사이에는 빈 공간이 있고 표면이 정의되지 않기 때문에 완전한 물체로 표현되지 않는다. 따라서 빈 공간을 메우고 표면을 정의하기 위해 point들을 이어 다각형으로 이루어진 mesh를 구성하여



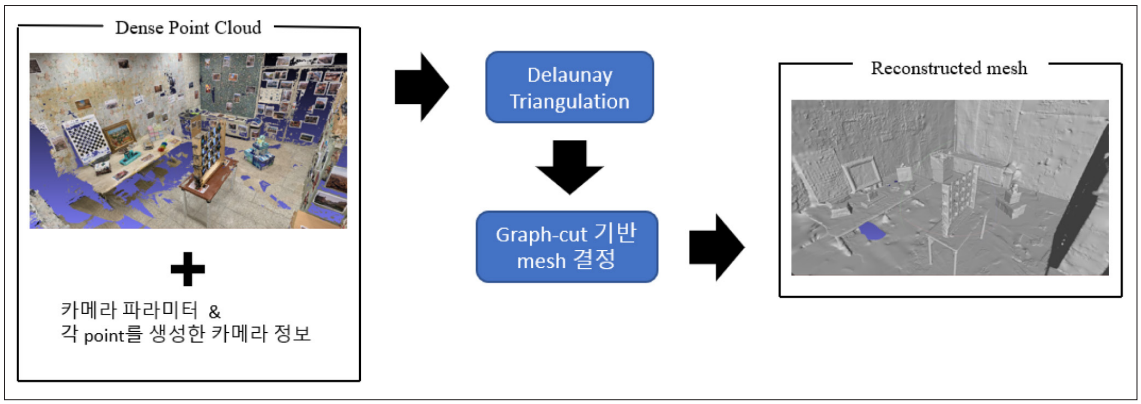
<그림 1> (a): Dense Point Cloud, (b): Reconstructed Mesh, (c): Textured Mesh

물체나 환경을 표현하고 이 과정을 mesh reconstruction 이라고 한다. Mesh reconstruction을 통해 물체나 환경의 표면을 정의함으로써, 표면에 텍스처를 적용하여 미디어에 생동감을 더할 수 있고 다른 물체와의 상호작용 또한 적용할 수 있다. 때문에 mesh reconstruction은 3D 미디어 생성의 핵심 과정 중 하나라고 할 수 있다.

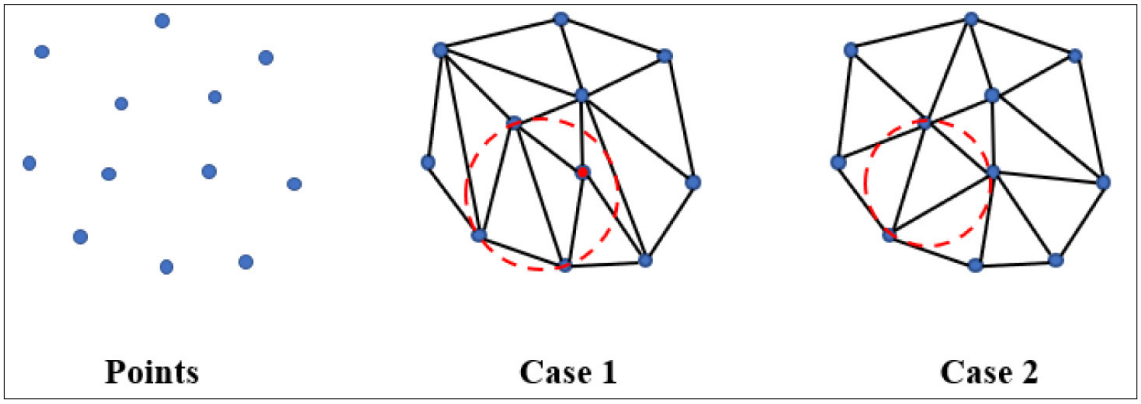
본 기고문에서는 conventional mesh reconstruction 방법들 중 Delaunay tetrahedralization과 graph-cut을 이용한 방법[3,4]에 대해 설명하고, learning-based mesh reconstruction 방법들 중 NeuralRecon[5]를 중점으로 소개한다.

Ⅱ. Conventional Mesh Reconstruction

〈그림 2〉는 Graph-cut based mesh reconstruction [3,4]의 과정을 간략히 나타낸 것이다. 입력으로 depth map을 이용해 생성한 dense point cloud와, 영상들의 intrinsic parameter와 extrinsic parameter를 받아 Delaunay Triangulation(들로네 삼각분할)으로 point cloud의 점들을 이어 사면체를 구성하고, Graph-cut 기반 mesh decision을 통해 물체의 표면만 mesh로 남긴다. Mesh reconstruction 과정은 point cloud에 포함된 노이



<그림 2> Mesh Reconstruction 과정

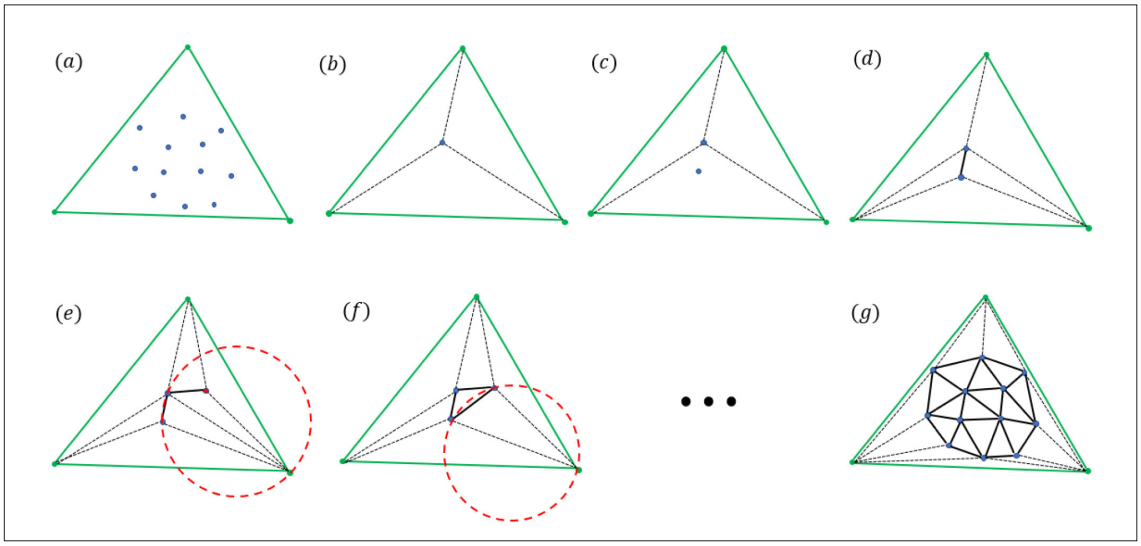


<그림 3> Triangulation 방법 비교

즈를 고려하면서, point cloud 사이의 빈 공간을 채움과 동시에 물체의 표면이 아닌 영역을 제거해야 한다.

1. Delaunay Triangulation

Delaunay Triangulation은 평면 위의 점들을 삼각형으로 연결하여 공간을 분할할 때, 삼각형들의 내각의 최소값이 최대가 되도록 분할하는 방법이다. 〈그림 3〉은 평면상의 점들을 각각 다른 방법으로 triangulation한 결과이다. Case 2는 Delaunay triangulation을 사용한 경우이고 Case 1은 그렇지 않은 경우이다. 〈그림 3〉에서 볼 수 있듯이 Case 2의 분할 방법이 Case 1에 비해 삼각형들의 내각이 작아 정삼각형에 가까운 모습을 보인다. Delaunay triangulation의 가장 중요한 특징은 어떠한 삼각형의 외접원도 다른 점을 포함하지 않는다는 것이다. Case 1의 외접원 안에는 다른 점이 포함된 경우가 있지만 Case 2는 어떠한 외접원도 다른 점을 포함하지 않는다. 이러한 Delaunay triangulation의 조건을 만족하는지에 대



<그림 4> Incremental 알고리즘

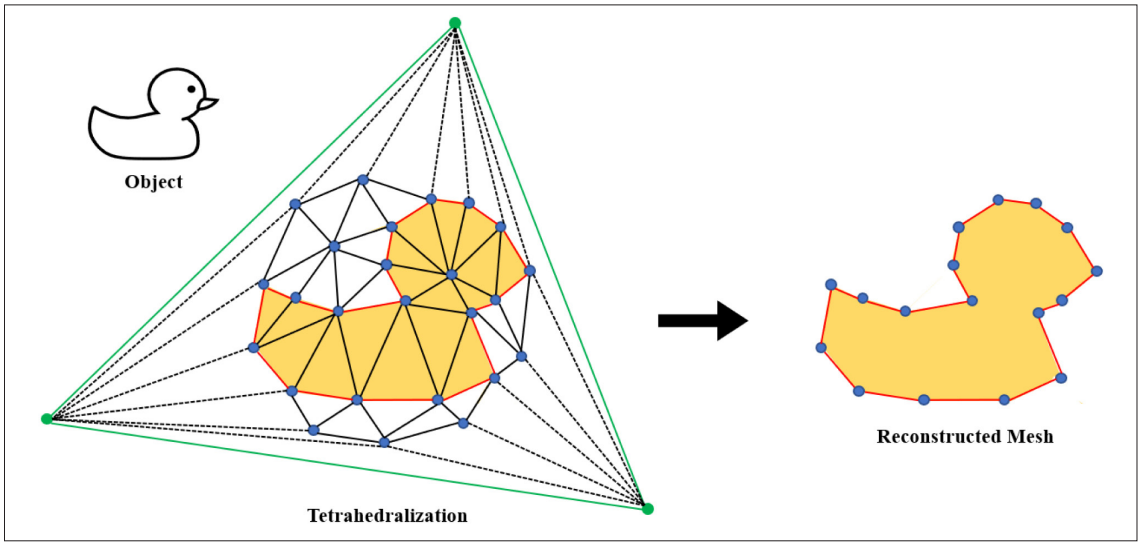
한 검사를 flipping 검사라고 하고 이를 이용해 Delaunay triangulation을 만족하도록 점을 이을 수 있다.

〈그림 4〉는 Delaunay triangulation을 구하는 incremental 알고리즘의 과정을 그림으로 나타낸 것이다. (a)에선 모든 점을 포함하는 삼각형을 구성하는 외부의 점 3개를 추가하고 (b)~(d)에선 점을 하나씩 추가하고 triangulation하며 flipping 검사를 수행한다. (e)에선 새로운 점을 추가하고 triangulation 했을 때, flipping 검사를 만족하지 못하는 경우가 발생한다. (f)에서는 이를 해결하기 위해 기존에 연결된 선을 지우고 새로 연결하여 flipping 검사를 만족하도록 한다. 이와 같은 과정을 모든 점들이 Delaunay triangulation의 조건을 만족하는 (g)와 같은 상태가 될 때까지 반복한다.

앞서 설명한 내용은 2차원 평면상에 존재하는 점들에 대한 Delaunay triangulation 방법이다. Mesh reconstruction은 3차원 공간상에 존재하는 dense point cloud의 점들에 대한 Delaunay triangulation을 구해야 하므로 삼각형은 사면체로, 원은 구로 확장하여 tetrahedralization을 수행해야 한다. 또한 점을 추가할 때, point cloud에서 점과 점 사이의 거리가 임계값 이상인 점들만 추가하여, 데이터가 지나치게 복잡해지는 것을 방지한다. 이때 각 point마다 point를 포함하고 있는 view에 대해서 view weight α_{vis}를 업데이트하고, 새로 추가한 point가 이미 추가된 어떤 point p와의 거리가 임계값 미만일 경우, 새로 추가한 point를 삭제하고 point p의 α_{vis}를 업데이트한다.

2. Graph-cut based Mesh Decision

Delaunay triangulation을 마친 후에 3차원 공간상의 point cloud를 이루는 점들은 사면체로 연결되어 있다. 이렇게 연결된 사면체들 중, 물체의 표면을 구성할 mesh를 결정하는 과정을 mesh decision이라고 한다. 〈그림 5〉는 object를 Incremental 알고리즘을 이용해 Delaunay tetrahedralization을 수행하고 mesh decision을 통해 mesh를 구성한 결과를 2차원상에 나타낸 것이다. 여기서 물체에 해당하는 사면체는 노란색으로 표기된 사면체이고, 이에 따라 mesh는 〈그림 5〉에서 빨간색으로 칠해진 면으로 결정되어야 한다. Graph-cut 기반 mesh decision의 목표는 사면체를 graph의 node로, 사면체와 다른 사

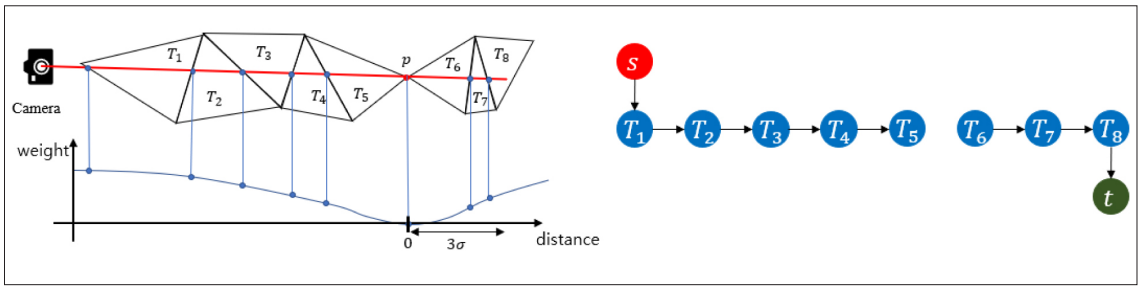


<그림 5> Mesh Decision

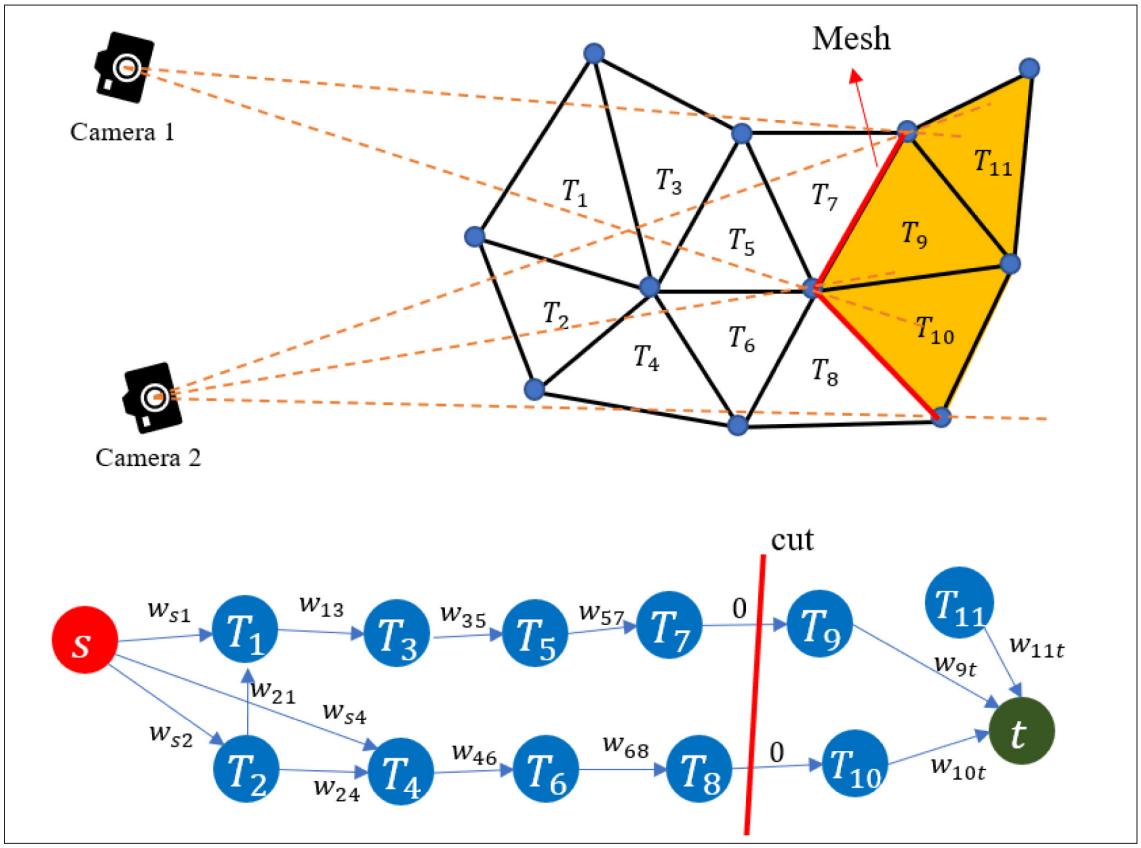
면체가 만나는 면은 graph의 edge로 설정하고, 각 edge에 weight를 부여하여 graph-cut 방법을 이용해 물체의 외부와 내부를 구성하는 node를 구분하는 것이다.

$$w = \alpha_{vis}\left(1 - exp\left(-\frac{d^2}{2\sigma^2}\right)\right) \quad (1)$$

〈그림 6〉은 graph의 edge에 weight w를 주는 방법을 그림으로 나타낸 것이다. 카메라에서 생성된 point p와 카메라를 잇는 선분을 그리고, 노이즈에 보다 강인한 mesh를 생성하기 위해 p에서 3σ만큼 연장한 선분을 정한다. 선분과 사면체의 교점과 p와의 거리를 d라고 한다. d들의 중앙값을 σ라고 했을 때 각 edge에 부여하는 가중치 weight



<그림 6> Graph의 edge에 weight를 주는 방법



<그림 7> Graph-cut based Mesh Decision

<그림 8> Dense Point Cloud를 이용해 Mesh reconstruction을 수행한 결과

는 식(1)과 같다. 카메라에서 p에 도달할 때까지의 교점은 graph의 source와 연결되고 p 뒤에 있는 교점은 sink와 연결되어 weight를 부과한다.

이러한 과정을 모든 point들에 대해 진행하여 edge의 weight를 누적하여 계산하고 graph를 형성한다. 이때 source와 sink에 연결된 node들의 집합을 각각 S와 T로 정하고 각 노드들이 연결된 순서쌍의 집합을 X라고 할 때 graph-cut을 위한 에너지함수는 식(3)과 같이 정의할 수 있다.

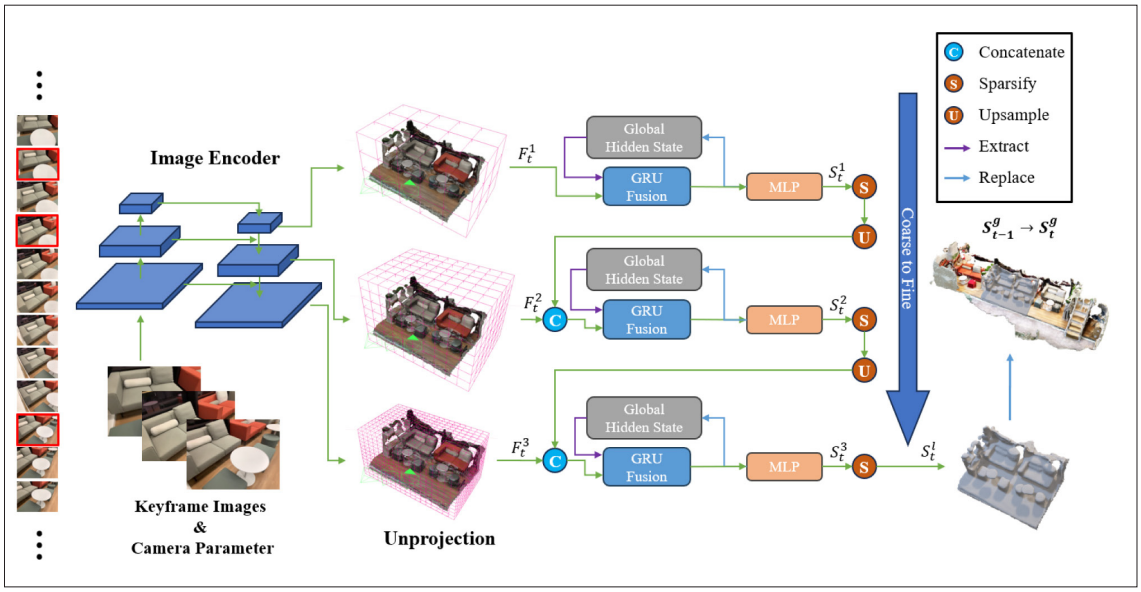
$$d_{ij} = \begin{cases} 1 & if\ i \in S \wedge j \in T \\ 0 & otherwise \end{cases} \quad (2)$$

$$E(S,T) = \sum_{(i,j)\in X} w_{ij} d_{ij} \quad (3)$$

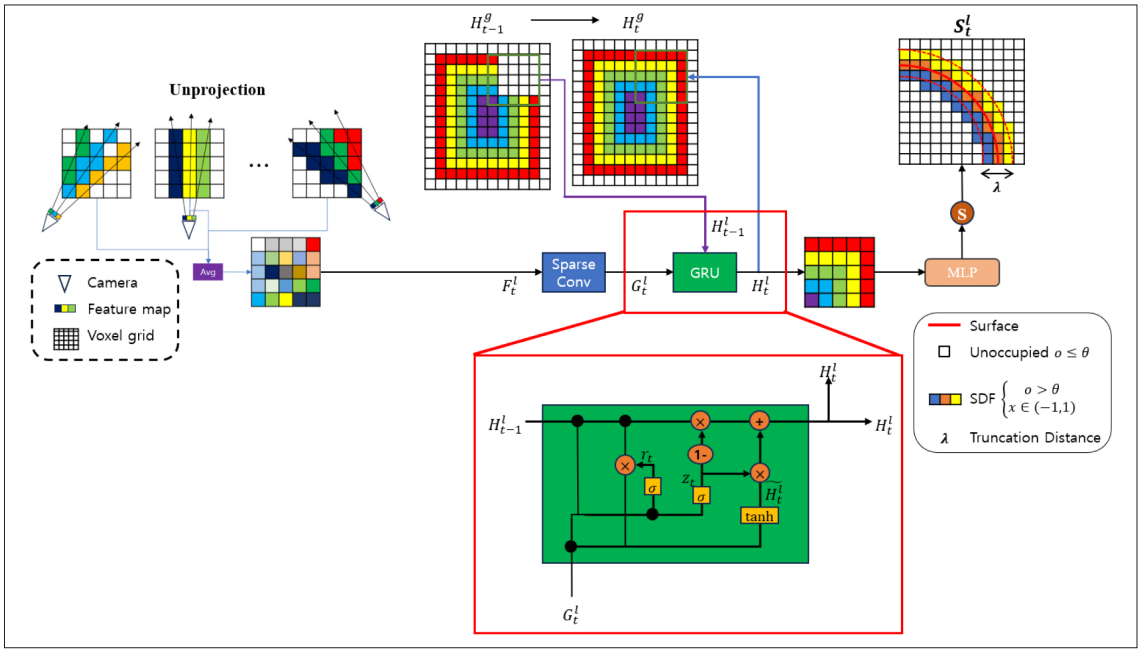
에너지함수 $E(S,T)$를 최소화할 수 있는 cut에 해당하는 edge, 즉 해당 사면체의 면을 최종 mesh로 결정한다.

Ⅲ. Learning-Based Mesh Reconstruction

2장에서 설명한 conventional method는 depth map에 의해 dense point cloud가 형성이 되고, dense point cloud를 기반으로 Delaunay tetrahedralization을 수행한 뒤, 이 중에서 mesh를 결정한다. 따라서 정확한 dense point cloud를 얻는 것, 즉 정확한 depth map을 구하는 것이 mesh의 품질에 큰 영향을 끼친다. 딥러닝을 이용하지 않고 depth map을 추정하는 [6]과 같은 방법은 특징점이 없는 영역이나 빛 반사 있는 영역 등에서 depth를 정확히 추정하지 못하고, 계산량이 많아 시간이 오래 걸린다는 단점이 있다. 이러한 이유로 딥러닝을 통해 더 빠르고 정확한 depth map을 얻는 [7,8,9]와 같은 연구들이 발표되었다. 이러한 연구들은 conventional depth map 추정 방법의 단점을 극복하여 보다 빠르고 정확한 depth map을 추정할 수 있도록 하였다. 하지만 depth map을 이용하여 mesh reconstruction을 수행하는 depth-based 방법은 여전히 문제점을 가지고 있다. Depth-based 방법들은 각 영상의 depth map을 추정할 때, 각 추정마다 이전 결과를 참조하지 않고 처음부터 추정을 수행한다. N번째 영상과 N+1번째 영상에서 중복되는 영역이 많아도 각 영상에서 개별적으로 depth를 추정하기 때문에 depth의 scale이 달라지거나, 이웃한 영상끼리 중복된 영역을 여러 번 계산하며 계산량이 늘어나는 문제를 가지고 있다. 이러한 depth-based method의 단점을 해결하기 위해 NeuralRecon[5]은 depth를 추정하지 않고 영상



<그림 9> NeuralRecon의 구조



<그림 10> NeuralRecon의 세부 과정

과 카메라 파라미터를 이용해 TSDF(Truncated Signed Distance Function) volume을 추정하고 marching cube 알고리즘[10]을 이용해 mesh를 생성하는 End-to-End 모델을 발표하였다.

NeuralRecon[5]은 Apple의 ARkit을 이용하여 동영상을 촬영하고, 영상의 각 frame의 카메라 파라미터를 구

$$r_t = \sigma\left(SparseConv(H_{t-1}^l, W_{hr}) + SparseConv\left(G_t^l, W_{gr}\right)\right) \quad (4)$$

$$z_t = \sigma\left(SparseConv(H_{t-1}^l, W_{hz}) + SparseConv\left(G_t^l, W_{gz}\right)\right) \quad (5)$$

$$\widetilde{H_t^l} = \tanh\left(\text{SparseConv}(r_t \odot H_{t-1}^l, W_{hh}) + \text{SparseConv}\left(G_t^l, W_{gh}\right)\right) \quad (6)$$

해 이를 입력으로 사용한다. 동영상의 frame 중 카메라의 각도나 이동 거리가 일정값 이상인 frame을 keyframe으로 선정한다. 선정한 keyframe들과 카메라 파라미터들을 9개씩 구성하여 하나의 fragment를 구성하고, Feature Pyramid Network[11]를 적용한 MnasNet[12]에 keyframe들을 입력하여 각자 다른 크기를 가진 feature map을 추출한다. 추출한 feature map은 unprojection 과정을 통해 3D voxel 값을 채우고 이를 GRU[13]를 이용해 이전 상태와 fusion하고 TSDF의 SDF값과 occupancy 값을 얻어 Sparse TSDF를 구한다. 이때 보다 정확하고 세밀한 추정을 위해 Coarse to Fine 디자인을 사용한다.

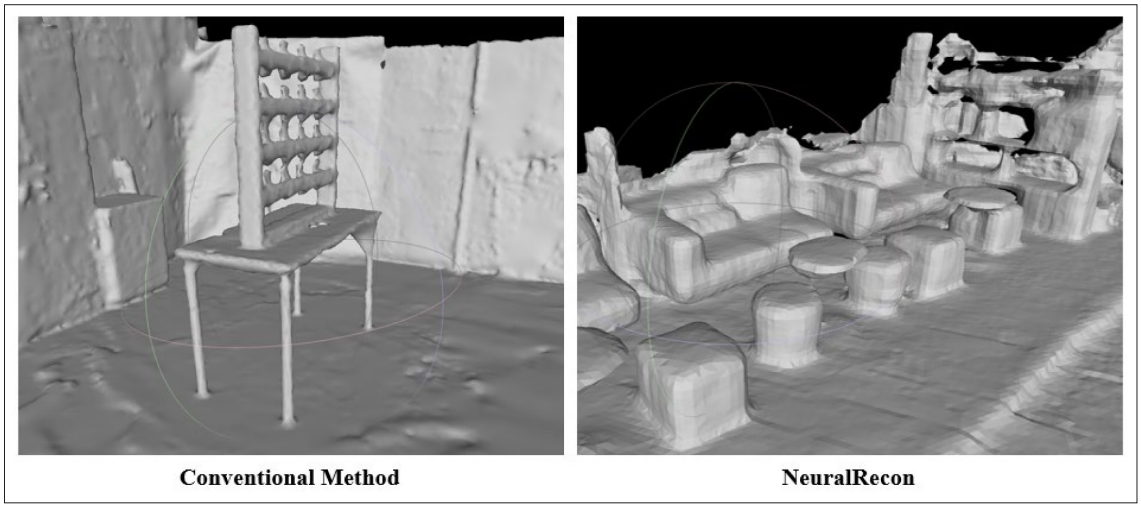
이 과정을 세부적으로 보면, 우선 fragment에 속한 각 카메라마다 voxel grid에 unprojection하여 그 값들을 누적하여 더한 뒤 visibility weight에 따라 평균을 구한다. 이렇게 평균을 내어 구한 volume은 feature volume이라고 하고 F_t^l로 표기한다. F_t^l는 3차원 공간에 2D feature를 투사한 데이터이기 때문에 voxel의 값이 0인 영역이 많다. 따라서 계산의 효율성을 위해 sparse convolution을 통해 0이 아닌 구간에서만 convolution을 하고 그 결과를 3D geometric feature G_t^l이라고 한다.

$$H_t^l = (1 - z_t) \odot H_{t-1}^l + z_t \odot \widetilde{H_t^l} \quad (7)$$

G_t^l는 이전 Global hidden state H_{t-1}^g에서 추출한 H_{t-1}^l와 함께 GRU gate의 입력으로 사용된다. GRU gate에서 reset gate vector r_t와 update gate vector z_t를 구하는 방법은 각각 식(4), 식(5)와 같고, output H_t^l는 식(6)과 식(7)을 통해 구한다. H_t^l를 바탕으로 Global hidden state를 업데이트하여 H_t^g를 구한다. 마지막으로 H_t^l을 MLP layer를 통과시켜 각 voxel마다 occupancy score o와 SDF value x를 추정한다. Occupancy score o는 해당 voxel이 Sparse TSDF volume S_t^l에서의 surface로부터 truncation distance λ 안에 있는지에 대한 신뢰도이다. Threshold θ보다 작을 경우 해당 공간에는 유의미한 SDF value가 없는 것으로 가정하고 모든 값을 0으로 정한다. (Sparsify) 위 과정을 마치면 nearest neighbor 방법을 사용해 S_t^l을 upsampling하고 다음 feature level의 F_t^l에 더하여 입력으로 사용한다. 3번째 level에서 최종적으로 출력된 S_t^l을 이용하여 S_{t-1}^g을 update한다. Sparse TSDF volume 형태의 S_t^l은 marching cube 알고리즘[10]을 통해 손쉽게 mesh로 나타낼 수 있다. o는 Binary Cross Entropy, SDF value는 L1 Loss를 사용하여 학습을 진행한다.

Sparse Convolution과 Sparse TSDF를 이용하여 계산의 효율성을 높이고 Coarse to Fine 디자인을 통해 mesh의 품질을 높인 결과, NeuralRecon은 이전의 depth map을 이용하는 learning based 방법들[7,8,9]이나 Atlas[14]에 비해 보다 빠르고 좋은 품질의 mesh를 얻을 수 있었다. 하지만 voxel의 크기를 작게 할수록 모델의 메모리 사용량과 계산량이 많아져 물체를 표현할 수 있는 디테일에는 다소 한계가 있다.

〈그림 11〉에서 확인할 수 있듯이 conventional method는 책상의 다리와 같은 얇은 물체도 잘 표현하지만 NeuralRecon은 책상의 다리가 표현되지 않는 것을 확인할 수 있다. 하지만 평면 표현의 경우 conventional method는 노이즈에 의해 매끈한 표면이 울퉁불퉁하게 표현되는 반면 NeuralRecon은 보다 평평하게 나타나는 것을 확인할 수 있다. 소요 시간 또한 conventional method



<그림 11> Conventional Method와 NeuralRecon 결과 비교

는 수십 분이 걸리는 반면 NeuralRecon은 수분 안에 mesh를 재구성할 수 있다.

Ⅳ. 결 론

3D reconstruction 과정에서 표면을 복원하는 mesh reconstruction 과정은 immersive한 3D 멀티미디어 생성을 위해 중요한 과정 중 하나이다. Conventional mesh reconstruction은 depth map을 이용해 point cloud를 생성하고, Delaunay tetrahedralization을 통해 point들을 이어 사면체들을 생성하고 s-t cut을 이용해 표면으로 추정되는 면만 남긴다. 이러한 방법은 depth map의 품질에 큰 영향을 받기 때문에 딥러닝을 이용해 더 빠르고 정확한 depth map을 추정하는 연구들이 진행되어 왔다. 하지만 depth map을 이용한 방법들은 그 과정이 비효율적이고 오차를 일으키는 내용이 있어 depth map을 사용하지 않고 바로 TSDF volume을 추정하고 mesh를 생성하는 방향으로 연구가 진행되기도 하였다. 본 기고문에서는 이러한 방법의 예시로 NeuralRecon[5]의 내용을 소개하였다. NeuralRecon[5]는 기존의 방법들에 비해 속도가 빠르고 다른 딥러닝을 이용한 mesh reconstruction 방법에 비해선 mesh의 품질이 좋지만, 여전히 conventional method의 mesh 품질에 비하면 그 성능이 많이 부족하다. 따라서 향후 연구들에서는 mesh의 품질을 conventional method만큼 높이면서도 빠른 속도로 mesh를 추정할 수 있는 방향으로 연구가 진행될 것이다. 동시에 모델의 경량화를 통해 모바일 기기 수준에서도 이와 같은 mesh 추정을 할 수 있도록 하여 3D 콘텐츠의 대중화를 위한 연구도 진행될 것이라 예상한다.

참 고 문 헌

[1] P. Moulon, P. Monasse, and R. Marlet, “Adaptive structure from motion with a contrario model estimation,” In Asian Conf. Comput.Vision, pp.257-270, Springer, 2013. Available: github.com/openMVG/openMVG/.
[2] Cernea, D. OpenMVS: Open Multiple View Stereovision. Available: https://github.com/cdcseacave/openMVS/
[3] P. Labatut, J.-P. Pons, and R. Keriven, “Robust and efficient surface reconstruction from range data,” Computer Graphics Forum, vol. 28, no. 8, pp. 2275-2290, 2009.
[4] Jancosek, Michal & Pajdla, Tomas. (2014). Exploiting Visibility Information in Surface Reconstruction to Preserve Weakly Supported Surfaces. International Scholarly Research Notices. 2014. 1-20. 10.1155/2014/798595.
[5] J. Sun, Y. Xie, L. Chen, X. Zhou and H. Bao, “NeuralRecon: Real-Time Coherent 3D Reconstruction from Monocular Video,” in 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 2021 pp. 15593-15602.
[6] Enliang Zheng, Enrique Dunn, Vladimir Jojic, and JanMichael Frahm. PatchMatch Based Joint View Selection and Depthmap Estimation. In CVPR, 2014.
[7] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. MVSNet: Depth Inference for Unstructured Multi-View Stereo. In ECCV, 2018. 2
[8] Kaixuan Wang and Shaojie Shen. MVDepthNet: Real-Time Multiview Depth Estimation Neural Network. In 3DV, 2018
[9] Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, and Ping Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In CVPR, 2020.
[10] William E. Lorensen and Harvey E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. SIGGRAPH, 1987.
[11] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In CVPR, 2017.
[12] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. MnasNet: Platform-aware neural architecture search for mobile. In CVPR, 2019.
[13] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In NeurIPS 2014 Workshop on Deep Learning, 2014.
[14] Zak Murez, Tarrence van As, James Bartolozzi, Ayan Sinha, Vijay Badrinarayanan, and Andrew Rabinovich. Atlas: Endto-End 3D Scene Reconstruction from Posed Images. In ECCV, 2020.

저 자 소 개

이 승 용

- 2023년 : 세종대학교 전자정보통신공학과 공학사
- 2023년 9월 ~ 현재 : 세종대학교 정보통신공학과 석사과정
- ORCID : https://orcid.org/0009-0005-7843-1083
- 주관심분야 : 영상 신호처리, VR

한 종 기

- 1992년 : KAIST 전기및전자공학과 공학사
- 1994년 : KAIST 전기및전자공학과 공학석사
- 1999년 : KAIST 전기및전자공학과 공학박사
- 1999년 3월 ~ 2001년 8월 : 삼성전자 DM연구소 책임연구원
- 2001년 9월 ~ 현재 : 세종대학교 전자정보통신공학과 교수
- 2008년 9월 ~ 2009년 8월 : University California San Diego (UCSD) Visiting Scholar
- ORCID : https://orcid.org/0000-0002-5036-7199
- 주관심분야 : VR, 3D 영상신호 구현, 비디오 코덱, 영상 신호처리, 정보 압축, 방송 시스템