ISSN 1213-6972 Journal of WSCG http://www.wscg.eu/ Vol.26, 2018, No.2

Mesh-based Multi-view Normal Integration with Energy Minimization Using Surface Reflectance Properties

Wichayut Eaksarayut
Department of Computer Engineering,
Faculty of Engineering,
Chulalongkorn University
Bangkok, 10330, Thailand
wichayut.ea@student.chula.ac.th

Borom Tunwattanapong
Ratchathani University
Ubon Ratchathani, 34000, Thailand
borom@rtu.ac.th

Pitchaya Sitthi-amorn
Department of Computer Engineering,
Faculty of Engineering,
Chulalongkorn University
Bangkok, 10330, Thailand
pitchaya@cp.eng.chula.ac.th

Nuttapong Chentanez
Department of Computer Engineering,
Faculty of Engineering,
Chulalongkorn University
Bangkok, 10330, Thailand
nuttapong26@gmail.com

ABSTRACT

We propose a technique to reconstruct a general 3D object using surface reflectance information from multiple viewpoints. Our core optimization framework uses multi-view normal integration, which can recovers water-tight surface of the object iteratively in a coarse to fine manner. The integration requires normal vector field from multiple viewpoints, which we can derive from surface reflectance. We then handle the topological changes if self-intersection occurs from the optimization. We also employ the idea of multi-resolution and weighted data heuristic which helps dealing with noisy data and improves both accuracy and optimization time. Our experiment shows that the framework is able to robustly recover 3D surface well with both synthetic and real data.

Keywords

3D Reconstruction, Multi-view normal integration, Multi-view vision, Triangle mesh-based surface

1 INTRODUCTION

3D reconstruction has been widely focused in the field of computer graphics and visions with various applications in today's life, such as medical, engineering, advertisement, and entertainment. This influences researchers to develop new techniques to solve this problem more efficiently and with higher accuracy. Existing state-of-the-art algorithms can reconstruct 3D objects with great accuracy, however they typically cannot handle surface that consist if both highly diffuse and highly specular parts. We leverage recent acquisition techniques that can accurately capture surface normal vector and specular reflection vector [17], and focus our 3D reconstruction algorithm base on normal integration.

There has been a considerable amount of researches that studied the multi-view normal integration problem [5, 15, 18]. Chang et al. [5] is the first to proposed the energy functional for multi-view normal integration that is derived from the classical single view shape-from-shading problem [13] and variational framework has been used in most researches to solve this error functional. Techniques above used implicit functions to represent the surface which gives an advantage on topology adaptation while performing mesh deformation. However, accurately representing a 3D object using implicit functions typically require large memory consumption and computation time, as it requires three-dimensional voxels to represent all the surface. [19] proposed an optimization framework which used triangular-mesh to represent the surface. However, they convert their mesh to an implicit surface in order to handle topological changes. This causes the edge length of the mesh to be up to the size of voxels in which fine details can be lost from converting to implicit surface.

https://doi.org/10.24132/JWSCG.2018.26.2.3

Our technique aims to use multi-view normal integration to reconstruct an arbitrary 3D object using normal and reflectance map from multiple viewpoints. We implemented multi-resolution optimization scheme in our framework which helps the overall optimization converges faster. We applied gradient descent to the error functional and perform all operations directly on the 3D triangle-based mesh. This enables us to control the resolution of the mesh during optimization. However, using this explicit surface representation has its drawbacks. Topology cannot be trivially change and self-intersection may occurs during optimization. We employ the method from [20] to remove self-intersection and handle topological change.

The rest of this paper is organized as follows: we review the related works on Section 2. We define our problem in Section 3. We then explain our proposed method in Section 4, and Section 5 to 6 will be our results and conclusion respectively.

Our main contributions are

- Mesh base optimization scheme that can handle topological change and self-intersection without conversion to implicit representation.
- Multi-resolution optimization.
- Optimization schedule that interleaves matching cost optimization with normal integration.
- Target normal calculation that takes visibility and multi-view information into account and can handle missing data.

2 RELATED WORK

3D reconstruction has gain a lot of attention in computer graphics and computer visions fields. In this section, we will focus on reviewing 3D reconstruction techniques that takes photometric and normal information as their inputs from multiple viewpoints. We refer the reading to an excellent survey for other 3D reconstruction method by Herbort and Wöhler [12].

Early methods for recovering surface information is shape-from-shading [3, 11, 13, 22]. These conventional methods were designed for reconstructing 2.5D surface from a single view information of texture-less object with known light position. Chang et al. [5] introduced a new technique that can reconstructs 3D surface using normal vector information from multiple viewpoints. They proposed their energy functional based on the single-view variational framework for shape-from-shading problem [13]. Geometric PDE is then derived to minimize their proposed functional and level-set method is used as their optimization framework. Recently, Weinmann et al. [18] employed a similar concept of multi-view normal integration in order to reconstruct the surface of high specular object. They calculated the volumetric normal field from projected illumination patterns and then applied global optimization with octree-based min-cut framework. The benefit of using an implicit surface (i.e. level-set, voxels, and octree) as their surface representation is that it automatically handles the topological changes while deforming the surface to the optimal target solution. However, it suffers from a large amount of memory consumption with more detailed mesh and can suffers from slow convergence rate.

A number of previous works uses other surface representation. Esteban et al. [10] refined a visual hull by finding photometric normal consistencies and then deformed their mesh on vertex space. However, problem like self-intersection was not taken into account in their paper. Similarly, Yoshiyasu and Yamazaki [19] used a hybrid framework between intrinsic and extrinsic surface representation by optimizing their energy terms on triangular mesh and convert the mesh into an implicit surface to handles self-intersections. Though, the detail of target mesh can be washed out when converting to implicit surface. Furthermore, Tunwattanapong et al. [17] presented a technique for recovering the geometry of 3D objects by projecting spherical harmonics basis on the object to acquire its reflectance information and then used message passing algorithm on vertex space to minimize their energy functional.

Our proposed method performs optimization directly on triangle mesh similar to [10, 19]. Our energy functional is related to [5], but adding more terms in visibility function to handle inter-reflections and noisy information better. We then minimize our energy functional using gradient descent scheme applying the method from Delaunoy et al. [9] which presented a framework to optimize a triangular mesh with gradient descent scheme. We handle the topological changes by employing similar algorithm from [20, 21]. Their algorithm could fix a mesh with self-intersection without losing details on the other part of the mesh. In addition to surface normal, we used reflectance information as our inputs. This allows our framework to work when the surface is not texture-less Lambertian. Our works compatible with other research in which they measured specularity [17, 18].

3 PROBLEM STATEMENT

The goal of our framework is to recover a full watertight triangular 3D mesh with reflectance information from multiple viewpoints with known intrinsic and extrinsic camera parameters. Our mesh consists of n vertices and m triangles which we denotes our vertices as a matrix $\mathbf{V} = [\mathbf{v}_1 \cdots \mathbf{v}_n]^T$ where $\mathbf{v}_i \in \mathbb{R}^3, i \in [1,n]$ denotes a point in 3D space, and triangle $\mathbf{F} = [\mathbf{f}_1 \cdots \mathbf{f}_m]$ where $\mathbf{f}_j$, $j \in [1,m]$ is a set consists of three adjacent vertices.

Each vertex $\mathbf{v}_i$ has an outward normal $\mathbf{N}(\mathbf{p}_i)$, similarly, each triangle also has its outward normal $\mathbf{N}^{\mathrm{F}}(\mathbf{f}_j)$. Our framework also requires a set of l calibrated cameras $\mathbf{C} = \{\mathbf{c}_1, \cdots, \mathbf{c}_l\}$ located around the target object. Each camera $\mathbf{c}_k$ where $k \in [1 \cdots l]$ has its own intrinsic and extrinsic parameters which can be described as matrices $\mathbf{K}_k$ and $[\mathbf{R}_k|\mathbf{t}_k]$ respectively, where $\mathbf{t}_k$ is a translation vector in $\mathbb{R}^3$ for camera $\mathbf{c}_k$ and the projection from any point $\mathbf{v} \in \mathbb{R}^3$ to image domain of camera $\mathbf{c}_k$ can be written as $\tilde{\mathbf{v}_k} = \mathbf{K}_k[\mathbf{R}_k|\mathbf{t}_k]\mathbf{v}$,$\tilde{\mathbf{v}_k} \in \mathbb{R}^2$. For the simplicity, we will also define this projection function to be $\tilde{\mathbf{v}_k} = \pi_k(\mathbf{v})$ and a lookup function $\nu_{k,\mathbf{X}}(\tilde{\mathbf{v}_k})$ which will return information of image $\mathbf{X}$ at pixel $\tilde{\mathbf{v}_k}$.

We need some information to describe how incident light reflected the object surface which in this case, we use diffuse and specular property of the surface as it is well known and widely used in many research. These information describe how the light reflect from the object surface to the camera lens which we can then use them to optimize the target surface. Our cameras will capture (or synthetically generate) these reflection information separately in each viewpoint. Our research will use four type of reflection data which are, diffuse intensity, diffuse reflection, specular intensity, and specular reflection. These information can then be derived to surface normal and use them in the optimization process which we will elaborate them on Section 4.1.

4 PROPOSED METHOD

In this section, we explain the core algorithm in order to recover water-tight 3D mesh with reflectance information. We perform optimization directly on triangle mesh as in [5]. Therefore, we require an initial surface approximation which can be acquired from various procedures. In our work, we use shape-from-silhouette [14] to compute a visual hull and use them as an initial surface. We assumed that such information is also given as a part of the input data.

We optimize the energy functional in coarse-to-fine manner by implementing multi-resolution optimization. We schedule more optimization iterations at coarse resolution and gradually decrease the optimization iterations in finer resolution iteration. This helps the overall framework to converge faster.

After we have a visual hull, we then minimize the cost functional based on geometric and photometric normal. The concept is to deform the mesh to match the target geometric normal with observed photometric normal.

The input from cameras typically have some noises. We add **target normal blending term** in order to filter out unwanted noise and make the reconstruction more robust and visually appealing.

We minimized our energy functional using a gradient descent scheme on vertex domain (Section 4.1). This is similar to surface evolution on implicit surface framework, instead we evolve our triangular mesh towards the gradient direction directly. This may leads to unwanted self-intersection artifacts. We perform an adaptive remeshing algorithm [20, 21] on self-intersected surface. Our overall procedures is shown in Algorithm 1.

Algorithm 1 Reconstruction Pipeline

```
(V,F) ← Shape-from-silhouette                ▷ Initial shape
for each resolution iteration do
    if mesh is coarse then
        (V,F) ← matchingcost-optimization(V,F)
    for each optimization iteration do
        Find target normal of each V and F
        Calculate ∇E of each V and F
        repeat
            α ← arg min_α E(deform(V,F,∇E,α))
            (V,F) ← deform(V,F,∇E,α)
            (V,F) ← fix-self-intersections(V,F)
        until E(V,F) is converges
    (V,F) ← resample(V,F)
return (V,F)
```

4.1 Multi-view Reflectance Integration

As in prior research about multi-view normal integration [5, 15, 18], we employ an error functional minimization framework based on the conventional shape-from-shading approach [13]. We minimize the cost functional with variational methods by minimizing the disparity of geometric and observed normal fields on the surface domain. In the other words, we evolve the surface so that its geometric normal field matched the observed normal field. However, a normal vector at a point on the given surface can be ambiguous with noisy data which should be taken into account.

We adjusted the multi-view normal field integration functional proposed by Chang et al. [5] which required an initial approximation of surface to integrate with. In this research, we acquired an initial shape approximation using shape-from-silhouette [14] as it is good enough for our algorithm.

From a given initial approximation surface, we refine them by displacing every vertices such that its geometric normal field of both from vertices and triangles matches the observed one. Thus, we define our cost functional of a given vertices $\mathbf{V}$ and triangles $\mathbf{F}$ as follows:

$$E(\mathbf{V},\mathbf{F}) = \sum_{\mathbf{v}\in\mathbf{V}} \omega(\mathbf{v})[1 - (\mathbf{N}_t(\mathbf{v}) \cdot \mathbf{N}_g(\mathbf{v}))] + \sum_{\mathbf{f}\in\mathbf{F}} \omega^{\mathrm{F}}(\mathbf{f})[1 - (\mathbf{N}_t^{\mathrm{F}}(\mathbf{f}) \cdot \mathbf{N}_g^{\mathrm{F}}(\mathbf{f}))] \quad (1)$$

where, $\mathbf{N}_t(\mathbf{v})$ and $\mathbf{N}_t^{\mathrm{F}}(\mathbf{f})$ are observed target normal at vertex $\mathbf{v}$ and triangle $\mathbf{f}$ respectively, $\mathbf{N}_g(\mathbf{v})$ and $\mathbf{N}_g^{\mathrm{F}}(\mathbf{f})$ are geometric normal at vertex $\mathbf{v}$ and triangle $\mathbf{f}$, $\omega(\mathbf{v})$ and $\omega^{\mathrm{F}}(\mathbf{f})$ are a weighting function of vertex $\mathbf{v}$ and triangle $\mathbf{f}$, based on the surface area.

Our reflectance information can be derived to normal vector so that it is consistent with our proposed cost functional. For diffuse component, we can derive them with the following equation:

$$\tilde{\mathbf{N}}_{k,\mathrm{diff}}(\mathbf{p}) = \mathrm{Normalize}\left(\alpha_r \nu_{k,\mathrm{diff}}(\tilde{\mathbf{p}}_k) - \mathbf{p}\right) \tag{2}$$

where $\nu_{k,\mathrm{diff}}(\tilde{\mathbf{p}}_k)$ is a lookup function for diffuse reflection of k$^{\mathrm{th}}$ camera at pixel $\tilde{\mathbf{p}}_k$, $\tilde{N}_{k,\mathrm{diff}}(\mathbf{p})$ is calculated diffuse normal at point $\mathbf{p}$ from camera k, α_r is a radius constant of a projection sphere where the incident light reflected to, and for specular component:

$$\tilde{\mathbf{N}}_{k,\mathrm{spec}}(\mathbf{p}) = \mathrm{Normalize}\left(\frac{\alpha_r \nu_{k,\mathrm{spec}}(\tilde{\mathbf{p}}_k) - \mathbf{p}}{|\alpha_r \nu_{k,\mathrm{spec}}(\tilde{\mathbf{p}}_k) - \mathbf{p}|} + \mathbf{p} - \bar{\mathbf{C}}_k\right) \tag{3}$$

Similarly, where $\nu_{k,\mathrm{spec}}(\tilde{\mathbf{p}}_k)$ is a lookup function for specular reflection, $\tilde{\mathbf{N}}_{k,\mathrm{spec}}(\mathbf{p})$ is calculated specular normal ,and $\bar{\mathbf{C}}_k$ is position vector of the k$^{\mathrm{th}}$ camera.

4.2 Target Normal Calculation

According to (1), there are both $\mathbf{N}_t(\mathbf{v})$ and $\mathbf{N}_t^{\mathrm{F}}(\mathbf{f})$ terms which we need to obtain by observing normal vectors from the photometric information provided. At a vertex $\mathbf{v}$, we calculate the normal vectors from diffuse and specular component separately and blend them with weighting constants as follows:

$$\mathbf{N}_t(\mathbf{v}) = \mathrm{Normalize}(w_{\mathrm{diff}}\mathbf{N}_{t,\mathrm{diff}}(\mathbf{v}) + w_{\mathrm{spec}}\mathbf{N}_{t,\mathrm{spec}}(\mathbf{v})) \tag{4}$$

where w_{diff} and w_{spec} are the weight for diffuse and specular component which can be calculated as follows:

$$w_{\mathrm{diff}} = \sum_{k\in\mathbf{C}} \alpha_{\theta,k}(\mathbf{v})\psi_k(\mathbf{v})\nu_{k,\mathrm{diffalbedo}}(\tilde{\mathbf{v}}_k) \tag{5}$$

$$w_{\mathrm{spec}} = \sum_{k\in\mathbf{C}} \alpha_{\theta,k}(\mathbf{v})\psi_k(\mathbf{v})\nu_{k,\mathrm{specconf}}(\tilde{\mathbf{v}}_k) \tag{6}$$

$$\alpha_{\theta,k}(\mathbf{v}) = \max(0, (-\hat{l}_k \cdot \mathbf{N}_g(\mathbf{v}))) \tag{7}$$

where $\hat{l}_k$ denotes a camera direction vector, $\mathbf{N}_g(\mathbf{v})$ is geometric normal at vertex $\mathbf{v}$, $\psi_k(\mathbf{v})$ is visibility function which will determine if camera $\mathbf{c}_k$ is visible for vertex $\mathbf{v}$. $\nu_{k,\mathrm{specconf}}(\tilde{\mathbf{v}}_k)$ is a look up function for diffuse albedo at point $\mathbf{v}$, and $\nu_{k,\mathrm{specconf}}(\tilde{\mathbf{v}}_k)$ is specular reflection confidence which depends on the acquisition technique.

For each component, we project this point to a set of visible cameras $\mathbf{C}_{\mathrm{seen}}$ and look up for reflectance information. We then use weighted average function based on camera angle towards the surface to calculate for the target normal as follows:

$$\mathbf{N}_{t,\mathrm{diff}}(\mathbf{v}) = \sum_{k\in\mathbf{C}} \alpha_{\theta,k}(\mathbf{v})\psi_k(\mathbf{v})\tilde{\mathbf{N}}_{k,\mathrm{diff}}(\mathbf{v}) \tag{8}$$

$$\mathbf{N}_{t,\mathrm{spec}}(\mathbf{v}) = \sum_{k\in\mathbf{C}} \alpha_{\theta,k}(\mathbf{v})\psi_k(\mathbf{v})\tilde{\mathbf{N}}_{k,\mathrm{spec}}(\mathbf{v}) \tag{9}$$

Similarly, for the triangle case, we used its centroid as a point of projection and then obtain target normal for the triangle.

The visibility terms $\psi_k(\mathbf{v})$ can be easily calculated using ray tracing algorithm like in previous research [5, 9, 15]. However, determining whether the surface in consideration is visible by just ray-tracing might not be enough as there could be some outliers (noise and inter-reflections) which can leads to inaccurate target normal. Therefore, we need to filter such outliers out first by restricting more conditions to visibility terms as follows:

$$\psi_k(\mathbf{v}) = \kappa_m(\mathbf{v})\kappa_p(\mathbf{v})\kappa_{cg}(\mathbf{v})\kappa_{ct}(\mathbf{v})\kappa_{tg}(\mathbf{v}) \tag{10}$$

$$\kappa_m(\mathbf{v}) = \begin{cases} 1, & \text{if } \mathbf{v} \text{ is visible at camera } k \\ 0, & \text{otherwise} \end{cases} \tag{11a}$$

$$\kappa_p(\mathbf{v}) = \begin{cases} 1, & \text{is } \textbf{not} \text{ self-intersected} \\ & \text{along the reflection vector} \\ 0, & \text{otherwise} \end{cases} \tag{11b}$$

$$\kappa_{cg}(\mathbf{v}) = \begin{cases} 1, & -(\hat{l}_i \cdot \mathbf{N}_g(\mathbf{v})) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{11c}$$

$$\kappa_{ct}(\mathbf{v}) = \begin{cases} 1, & -(\hat{l}_i \cdot \nu_{k,\mathrm{spec}}(\tilde{\mathbf{v}}_k)) > 0.5 \\ 0, & \text{otherwise} \end{cases} \tag{11d}$$

$$\kappa_{tg}(\mathbf{v}) = \begin{cases} 1, & \tilde{\mathbf{N}}_{k,\mathrm{spec}}(\mathbf{v}) \cdot \mathbf{N}_g(\mathbf{v}) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{11e}$$

Like in [5, 9, 15], the first term (11a) can be determine by tracing a ray from camera to vertex, if it is not occluded by any surface then this counts as visible. Although from our observation, there are several sources that lead to incorrect target normal acquisition, such as inter-reflections. (11b) checks whether the gathered information is bad from inter-reflection by tracing a ray from position $\mathbf{v}$ along the reflection vector respected to each viewpoint. If the ray hit the mesh itself, we will discard the information and treated this pixel as invalid. The term (11c) checks boundary cases when the ray-tracer hits back-face surface. This can be occurred when tracing to a point located near the silhouette or thin surfaces. For specular component, (11d) filters out the reflection vectors that have wide angle respected to its camera direction vector as the surface that face off the camera are likely to be noisy. Lastly, the term (11e) filters out the bad photometric reflection which face backward respected to the mesh geometry. This mostly occur in the area with inter-reflection.

4.3 Gradient Descent Optimization Scheme

With observed target normal being calculated on every vertices in $\mathbf{V}$ and triangles in $\mathbf{F}$, we then minimize our energy functional in (1) with gradient descent framework. Similar to [9], but with our proposed energy functional. Basically, we will deform our mesh by translating each vertex $\mathbf{v}_i$ along the calculated deformation direction vector $\mathbf{d}_i$ which can be written as:

$$\mathbf{v}_i' = \mathbf{v}_i + t\mathbf{d}_i \tag{12}$$

where $\mathbf{v}_i'$ denotes a deformed vertex $\mathbf{v}_i$ with scalar weight t for direction $\mathbf{d}_i$. This deformation vector can be computed by finding the gradient of energy functional in (1) and energy decreases when the surface is deformed in the opposite gradient direction. Thus, the deformation equation of the whole mesh can be written as:

$$\mathbf{V}' = \mathbf{V} - \beta \nabla E(\mathbf{V}, \mathbf{F}) \tag{13}$$

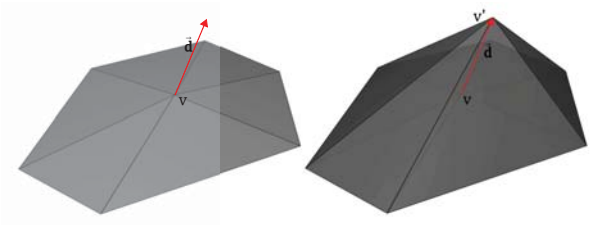

Figure 1: Vertex deformation of point $\mathbf{v}$ toward the direction vector $\mathbf{d}$ which can be calculated by finding gradient of vertex $\mathbf{v}$

Finding the gradient for each vertex $\mathbf{v}_i$ is not trivial, since our energy functional (1) is based on normal terms. Besides, we need to calculate the gradient respect to its position:

$$\nabla E(\mathbf{V}, \mathbf{F}) = \left[\frac{\delta E}{\delta x}(\mathbf{V}, \mathbf{F}), \frac{\delta E}{\delta y}(\mathbf{V}, \mathbf{F}), \frac{\delta E}{\delta z}(\mathbf{V}, \mathbf{F}) \right] \tag{14}$$

Our normal can be derived from its adjacent vertices using the following equations:

$$t_1 = \sum_{i=0}^{k-1} \cos\left(\frac{2\pi i}{k}\right) \mathrm{Adj}(\mathbf{v}, i) \tag{15a}$$

$$t_2 = \sum_{i=0}^{k-1} \sin\left(\frac{2\pi i}{k}\right) \mathrm{Adj}(\mathbf{v}, i) \tag{15b}$$

where $\mathbf{v}$ has k adjacent vertices, t_1 and t_2 are tangent vectors, and $\mathrm{Adj}(\mathbf{v}, i)$ returns the position of $\mathrm{i_{th}}$ adjacent vertex of $\mathbf{v}$. The cross product $t_1 \times t_2$ is then calculated for vertex normal. (For more in details please refer to [16]) With this we can solve for an analytic gradient of the energy with a symbolic differentiation package such as **sympy** [1].

We then perform line search algorithm to find the value β in (13) which will minimize our energy toward the current surface. Then from (13), we have:

$$\arg\min_{\beta} E(\mathbf{V} - \beta \nabla E(\mathbf{V}, \mathbf{F}), \mathbf{F}) = 0 \tag{16}$$

4.4 Target Normal Blending

Some part of the surface may not be captured with high quality information (e.g. highly concave surface) or that part of the surface is totally occluded. This could be problematic as observed target normal vector $\mathbf{N}_t(\mathbf{v})$ or $\mathbf{N}_t^{\mathrm{F}}(\mathbf{f})$ could be an undefined vector which caused by our visibility terms in (10) of every camera returns zero. This may leads to an undefined behavior for our optimization process. Therefore, our framework will need to handle this case, so that at least the surface without information can still be reconstructed with visually appealing output. We define a confidence function $\lambda(\mathbf{v})$ for our observed target normal or can be also called **normal blending weight**. This confidence value decreases as the calculated target normal become unreliable. We then use the confidence term to blend the calculated target normal with smoothed geometric normal using the following equation:

$$\mathbf{N}_t^{\mathrm{blend}}(\mathbf{v}) = \lambda(\mathbf{v})\mathbf{N}_t(\mathbf{v}) + (1 - \lambda(\mathbf{v}))\bar{\mathbf{N}}_g(\mathbf{v}) \tag{17}$$

where, $\bar{\mathbf{N}}_g(v)$ is normal vector of smoothed geometric surface at point $\mathbf{v}$. Our normal blending weight is varied to **the number of visible viewpoints** and **variance of photometric curvature of visible viewpoints**:

$$\lambda(\mathbf{p}) = \lambda_{\mathrm{H}}(\mathbf{p})\lambda_{\mathrm{C}}(\mathbf{p}) \tag{18a}$$

$$\lambda_{\mathrm{H}} = \exp\left(\min\left(0, -\frac{\sigma_{\mathbf{H}}}{2}\right)\right) \tag{18b}$$

$$\lambda_{\mathrm{C}} = \exp\left(\min\left(0, |\mathbf{C}_{\mathrm{seen}}| - \left\lceil \frac{(1-\cos\theta)}{2} \right\rceil |\mathbf{C}|\right)\right) \tag{18c}$$

where λ_{H} is photometric curvature variance term which can be calculated by looking up all normal components from visible cameras and compute its variance. That is, if the photometric normals are consistent, the calculated target normal is more likely to be reliable. Where, λ_{H} captures the photometric curvature variance. The term λ_{C} represents the vertex visibility. If the calculated target normal are computed from more viewpoints, the target normal is acceptable. We assumed that camera set $\mathbf{C}$ are uniformly located along the sphere that covers the scanning object. Then, at a particular vertex $\mathbf{v}$ on

Figure 2: Input reflectance information of **speccat** and **hammerman**. From left to right, diffuse albedo, diffuse reflection, specular albedo, specular reflection, and mask information for our optimization framework.

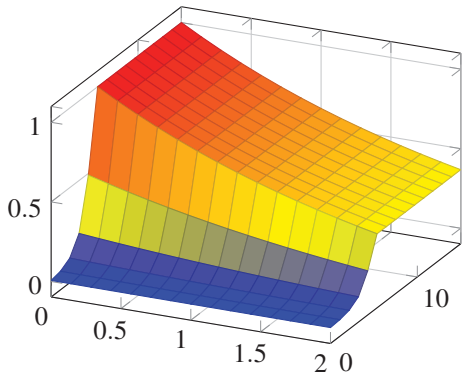

Figure 3: Blending weight function of our camera configuration, varied to $\sigma_{\mathbf{H}}$ and $|\mathbf{C}_{\text{seen}}|$ with 31 total cameras and $\theta = 45°$

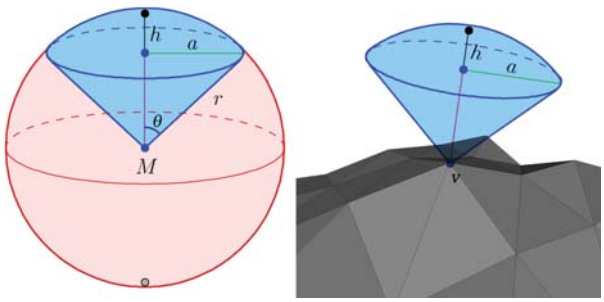

Figure 4: Cameras within an infinite radius spherical sector (hi-lighted in blue) will be marked as visible.

a surface, $\mathbf{v}$ will have a set of visible cameras $\mathbf{C}_{\text{seen}}$. The term $(1-\cos\theta)/2$ is derived from the ratio between surface of spherical sector to the whole sphere, where at a particular point $\mathbf{v}$ should be at least visible to the camera that is located on the part of spherical cap which in this research we set the θ value to be 45 degree. However, this equation is only based on our camera configuration. It could be adjusted to be suitable for other configuration as well.

4.5 Matching Cost Optimization

Normal integration has its limitation about ambiguity as stated in [2], which can result in an incorrect answer even the energy functional is converged. Especially in the concave area where the observed target normal can be inaccurate due to projection error. We can solve such problem by using similar idea to **stereo reconstruction**.

We solved this problem in a similar manner to the normal integration by defining the normal correspondence energy function based on mesh vertices and move them to the optimal solution. Given, a vertex $\mathbf{v} \in \mathbb{R}^3$ and camera set $\mathbf{C}$. We assumed that, if the vertex $\mathbf{v}$ is at the correct position, then its projected normal from each camera should be correspond to every other cameras. In order to find such correspondence, we defined our **matching cost function** to be a variance of observed normal of visible viewpoints where the number of visible viewpoint is more than 3. Otherwise we will force the matching cost to be $+\infty$

We sampled points along the vertex normal both outward and inward, then calculate the matching cost at each sampling. After that, from all computed matching cost samples, we fit a quadratic equation for the displacement from the sample with minimum cost and its adjacent samples. We then compute the location of the tip of the parabola. Finally, based on the matching cost of the optimal point, we translate the vertex along its normal with the displacement calculated earlier.

4.6 Remeshing

The drawback of using explicit surface representation like triangular mesh is that it could not automatically deal with topology changes unlike implicit surface representation. It is likely that self-intersection will occurred in our mesh due to our mesh evolution in Section 4.3.

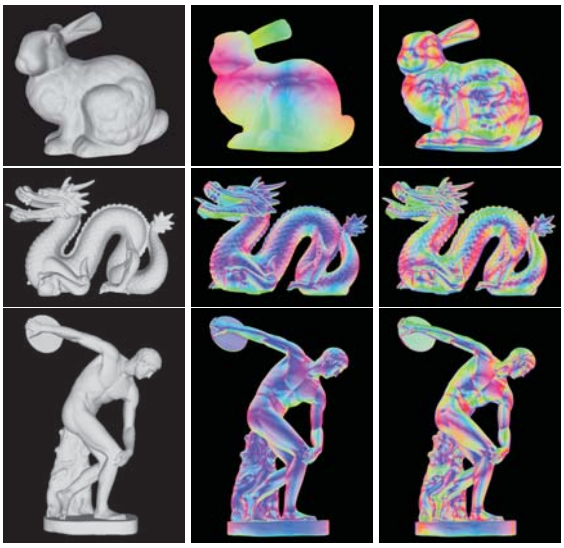

Figure 5: Synthetic data of bunny, dragon, and discolobus. From left to right, ground truth mesh, diffuse reflection, and specular reflection. We omitted diffuse and specular albedo since we set all the value into one.

The author in [20, 21] proposed a framework to efficiently solve topological changes on triangular mesh called **Transformesh**. The algorithm solves topology changes by using an intuitive geometrically driven solution which we found it suitable for our framework. We perform **Transformesh** algorithm after the whole mesh deformation process is completed in every iteration. Other self-intersection algorithm such as [4, 6, 7] can also be used in this step.

Then, after every resolution iteration, we resample our mesh to be finer with edge splitting operation and remove short edge with edge collapsing operation. We then use the mesh in the next optimization iteration.

5 RESULTS

In this section, we will discuss and evaluate the result of our reconstruction pipeline with both real and synthetic data. All procedures are executed with Intel i7-5820K 3.30GHz, with 64GB of RAM.

5.1 Real data

We performed the reconstruction on two real data (As shown in Fig.2). There are 31 viewpoints uniformly located on faces of truncated icosahedral (except one on the bottom-most) with 30 centimeters in radius which we can set our parameter α_r in (2) and (3) to be 30. All input images are captured in 4896 by 3684 pixels and camera matrices are already calibrated. Visual hull is then extracted for initial mesh with 1 millimeter in voxel edge length. We optimized more iterations in coarser resolution mesh as the coarse details will converged before refining the mesh in the finer iteration so that the optimization converges faster in overall. We scheduled 10 iterations or the coarse resolutions, then decreasing the number of optimization iterations in finer resolution iteration.

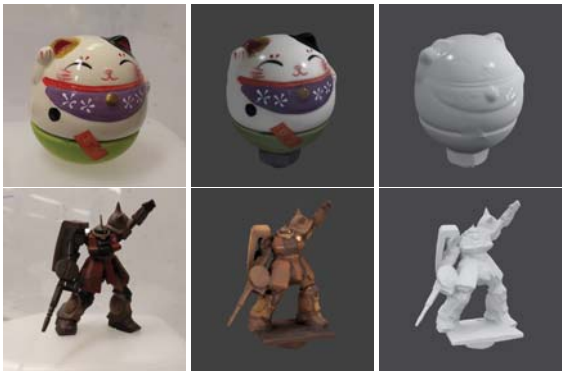

Figure 6: Reconstructed 3D models of speccat and hammerman. The real objects are shown on the left and rendered reconstructed outputs are shown on the middle and right.

Figure 6 shows the results of reconstructed mesh of **speccat** and **hammerman**. Our framework successfully recovered both data. The output of speccat has reasonable geometric features and being able to render an appealing result.

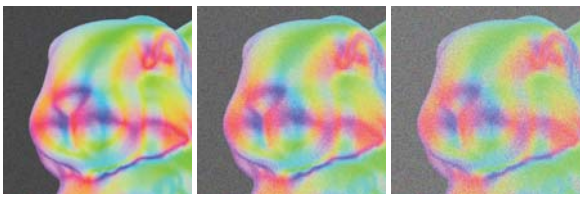

Figure 7: Additive white gaussian noise is added to bunny data with coefficient 0.1, 0.2, and 0.3

5.2 Synthetic data

We simulate the configurations from real data reconstruction from the last section, so that it is not biased or favoring to our framework. We used three ground truth meshes (As shown in Figure 5) and projecting reflectance information needed with similar camera calibration to the prior section. We replaced every pixels to be one for diffuse and specular intensity since there were no such information to project and this would not violate our framework. In addition, **additive white gaussian noise** is added to the generated images with coefficient **AWGN coeff** value set to 0.1, 0.2, and 0.3 (As shown in Fig. 7) and compared the result to evaluate the robustness of our framework.

We measure the error of our output with Hausdorff distance [8] as shown in Table 1. Note that our framework can reasonably recovered the shape even with noisy data. Only the concave area seems to far off the ground truth due to the matching cost optimization could not perfectly find the correct optimal point with noisy data.

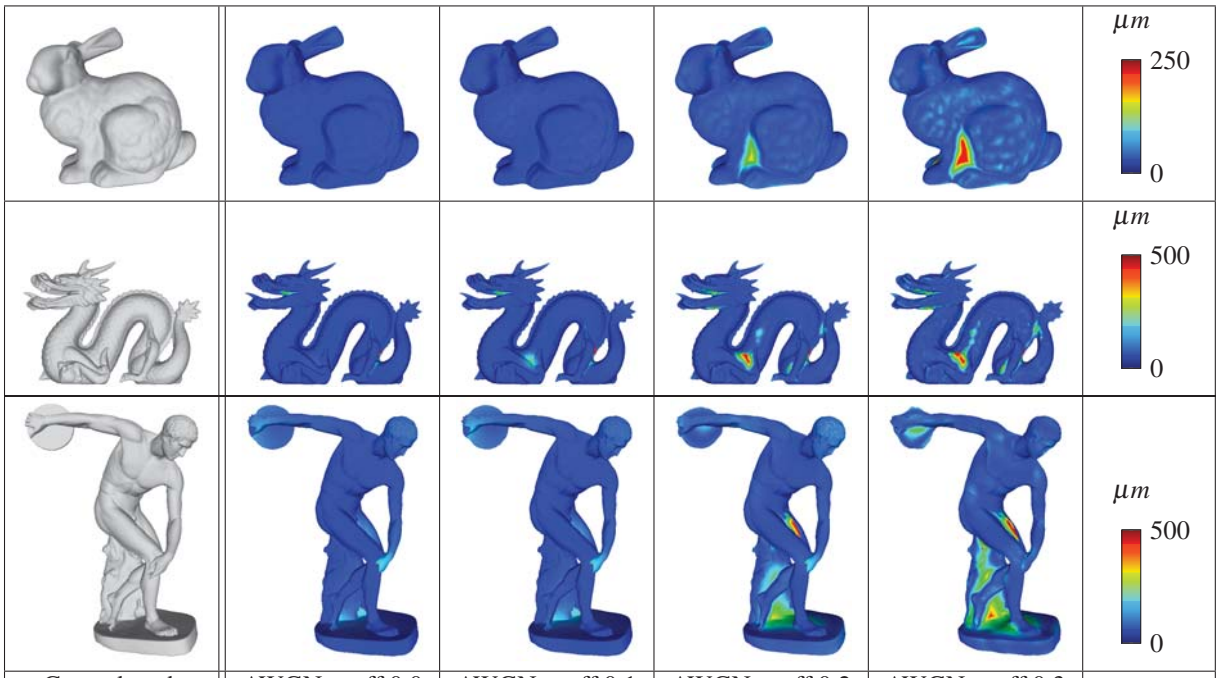

Figure 8: The Hausdorff distance of the outputs toward its ground truth.

Table 1: Hausdorff distance of outputs toward ground truths

Model	AWGN coeff	meand_H (mm)	RMS
Bunny	0	0.06026	0.07269
	0.1	0.06405	0.07791
	0.2	0.16838	0.27443
	0.3	0.23536	0.47291
Dragon	0	0.12251	0.27265
	0.1	0.15447	0.33639
	0.2	0.24416	0.46701
	0.3	0.30370	0.55531
Disco-lobus	0	0.21390	0.30124
	0.1	0.22146	0.30930
	0.2	0.36786	0.55528
	0.3	0.44007	0.71934

6 CONCLUSION

We have presented a novel multi-view normal integration framework using reflectance information. With our mesh-based optimization, we are able to reconstruct fine details without sacrificing unnecessary memory consumption unlike implicit surface framework. Although it can presents self-intersection, we exploit such problem by using **Transformesh** [20] which fix topology changes completely in triangular mesh domain. We also deal with those surface which only a few or none of the camera can be seen with target normal blending which will smooth out the surface without photometric information.

Our framework also has some limitations toward the area with high details and thin surface. This is due to the inability to observe high frequency target normals on the coarse iterations which result it smoothed out surface like in hammerman (Fig.6). This could be resolved with adaptive mesh w.r.t. photometric curvature and is an interesting area of future work. While our method can handle topological change during optimization, if the initial mesh is of different genus from the real mesh, our algorithm may not be able to change the genus. Hence, using a good initial mesh with matching genus may be needed. Creating a better initial mesh is hence another interesting area of future work.

7 REFERENCES

[1] Sympy. `http://www.sympy.org/`. Accessed: 2018-03-15.

[2] J. Balzer and S. Werling. Principles of shape from specular reflection. *Measurement*, 43(10):1305 – 1317, 2010.

[3] J. T. Barron and J. Malik. Shape, illumination, and reflectance from shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(8):1670–1687, Aug 2015.

[4] G. L. Bernstein and C. Wojtan. Putting holes in holey geometry: Topology change for arbitrary surfaces. *ACM Trans. Graph.*, 32(4):34:1–34:12, July 2013.

[5] J. Y. Chang, K. M. Lee, and S. U. Lee. Multiview normal field integration using level set methods. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[6] N. Chentanez, M. Müller, and M. Macklin. Gpu accelerated grid-free surface tracking. *Computers & Graphics*, 57:1–11, 2016.

[7] N. Chentanez, M. Müller, M. Macklin, and T.-Y. Kim. Fast grid-free surface tracking. *ACM Trans. Graph.*, 34(4):148:1–148:11, July 2015.

[8] P. Cignoni, C. Rocchini, and R. Scopigno. Metro: Measuring error on simplified surfaces. *Computer Graphics Forum*, 17(2):167–174, 1998.

[9] A. Delaunoy and E. Prados. Gradient flows for optimizing triangular mesh-based surfaces: Applications to 3d reconstruction problems dealing with visibility. *International Journal of Computer Vision*, 95(2):100–123, Nov 2011.

[10] C. H. Esteban, G. Vogiatzis, and R. Cipolla. Multiview photometric stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):548–554, 2008.

[11] R. T. Frankot and R. Chellappa. A method for enforcing integrability in shape from shading algorithms. *IEEE Transactions on pattern analysis and machine intelligence*, 10(4):439–451, 1988.

[12] S. Herbort and C. Wöhler. An introduction to image-based 3d surface reconstruction and a survey of photometric stereo methods. *3D Research*, 2(3):4, 2011.

[13] B. K. Horn and M. J. Brooks. The variational approach to shape from shading. *Computer Vision, Graphics, and Image Processing*, 33(2):174–208, 1986.

[14] A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2):150–162, Feb 1994.

[15] A. Osep. Multiview normal field integration using graph-cuts. In *Central European Seminar on Computer Graphics for Students*, Apr. 2012.

[16] P. Schroeder, D. Zorin, and W. Sweldens. Subdivision for modeling and animation. 1998.

[17] B. Tunwattanapong, G. Fyffe, P. Graham, J. Busch, X. Yu, A. Ghosh, and P. Debevec. Acquiring reflectance and shape from continuous spherical harmonic illumination. *ACM Transactions on graphics (TOG)*, 32(4):109, 2013.

[18] M. Weinmann, A. Osep, R. Ruiters, and R. Klein. Multi-view normal field integration for 3d reconstruction of mirroring objects. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2013.

[19] Y. Yoshiyasu and N. Yamazaki. Topology-adaptive multi-view photometric stereo. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1001–1008. IEEE, 2011.

[20] A. Zaharescu, E. Boyer, and R. Horaud. Transformesh: A topology-adaptive mesh-based approach to surface evolution. In *Proceedings of the 8th Asian Conference on Computer Vision - Volume Part II*, ACCV'07, pages 166–175, Berlin, Heidelberg, 2007. Springer-Verlag.

[21] A. Zaharescu, E. Boyer, and R. Horaud. Topology-adaptive mesh deformation for surface evolution, morphing, and multiview reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(4):823–837, 2011.

[22] R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah. Shape-from-shading: a survey. *IEEE transactions on pattern analysis and machine intelligence*, 21(8):690–706, 1999.