Bilateral Normal Integration

XU CAO[1,a)] HIROAKI SANTO[1] BOXIN SHI[2] YASUYUKI MATSUSHITA[1]

Abstract: This paper studies the discontinuity preservation problem in the task of surface recovery by integrating a single-view surface normal map. Although various sophisticated strategies have been proposed, this challenging problem has not been well solved due to unknown discontinuity positions in the normal map. The key of our method is to model the existence of discontinuity between every two adjacent pixels by the depth differences between that two pixels. At each pixel, we approximate the left and right (resp., upper and lower) partial derivatives and relatively weight the two approximations by comparing the depth differences at the left and right (resp., upper and lower) sides. Therefore we term our method as "bilateral" normal integration. By iteratively solving for the depths and updating the discontinuity maps, we can recover the surface with discontinuities. Experiments on various challenging surfaces demonstrate our method's effectiveness. In addition, we show that discontinuities can be preserved in perspective normal maps by our method.

Keywords: Normal integration, edge preservation, photometric 3D reconstruction, shape from single-view images

1. Introduction

Photometric surface recovery aims at high-fidelity three-dimensional (3D) surface reconstruction by exploiting the shading information from single-view images. Representative methods include shape from shading [8], photometric stereo [20], and shape from polarization [10]. These methods typically estimate a surface normal map as direct output; Fig. 1(a) shows an example. To recover the surface, we need to further integrate these surface normals, the problem called normal integration [15]. Therefore, the accuracy of photometric surface recovery depends on the accuracy of normal estimation and normal integration.

Despite the necessity, the existing normal integration methods are still unsatisfactory in recovering surfaces containing discontinuities. Most normal integration methods assume a smooth surface, *i.e.*, the surface is continuous and differentiable everywhere. In practice, as the normal maps are estimated from single-view images, the occlusion may occur, causing abrupt changes of depth values at occlusion boundary, as shown in Fig. 1(b) and (c). Applying the methods based on smooth surface assumption leads to heavily distorted surfaces because the surface is wrongly connected at discontinuities, as shown in Figure 1(d).

Unfortunately, preserving the discontinuity in a surface is challenging due to the unknown discontinuity positions. Many works have put efforts in detecting the discontinuities from the normal map, using curl [14], least squares residual [1], [11], or handcrafted features [22]. Given the detection results, they then use weights to indicate the discontinuities in the optimization. The detection can be one-time [14], [22] or repeated [1]; the weights can be binary [1], [19] or the possibility [16]. Such detection, however, can be fragile due to weak correlation between depth discontinuities and surface normals.

(a) (b) (c) (d) (e)

Fig. 1 Left to right: **(a)** The RGB color-coded normal map rendered from The Thinker with a perspective camera model[*2]. **(b)** and **(c)** The corresponding rendered surface's front view and side view. Depths abruptly change at occlusion boundary. **(d)** Integration result by [3], which assumes a smooth surface. The surface is wrongly connected at occlusion boundary. **(e)** Our integration result. Depth gaps at occlusion boundary are preserved.

Recently, Mumford-Shah integrator [16] has shown promising results on surfaces with simple discontinuities. The success relies on the short discontinuity assumption and the joint optimization for the weights and depths to bypass the discontinuity detection. However, it breaks down on surfaces with complex discontinuities, and the optimized weights may indicate wrong discontinuity positions.

This paper proposes bilateral normal integration to preserve the discontinuities in the surfaces when integrating the surface normals. Figure 1(e) shows our integration result; the large depth gaps at occlusion boundary are faithfully preserved. Our key idea is to use the depth differences between two adjacent pixels to indicate the existence of discontinuities. We observe that depth is a more straightforward clue for depth discontinuities than those

[1] Osaka University, Suita, Osaka, Japan
[2] Peking University, Beijing, China
[a)] cao.xu@ist.osaka-u.ac.jp

[*2] "The Thinker by Auguste Rodin" CC BY 4.0. https://sketchfab.com/3d-models/the-thinker-by-auguste-rodin-08a1e693c9674a3292dec2298b09e0ae, last accessed on Dec. 20, 2021.

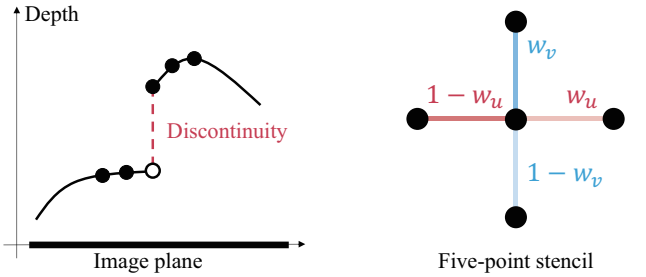

Fig. 2 Our designs on weights. **(Left)** The large depth difference between adjacent pixels is a straightforward indicator for discontinuity. Our weights have an inverse relation to depth differences. **(Right)** Bilateral weighting: the weights at two sides of a pixel are normalized to 1.

computed from normal maps, *e.g.*, curl [14]. If two adjacent pixels have a large depth difference, then discontinuity is likely to exist between that two pixels, as shown in Fig. 2 left. Therefore, our weights has an inverse relation to the depth differences. However, the depth difference between adjacent pixels can also be large at steep regions. To avoid mis-segmenting the surfaces at steep regions, we further introduce bilateral weighting. In particular, we approximate the surface gradient twice from two sides of each pixel and relatively weight the two approximations. We normalize the weights at two sides of a pixel such that they sum up to 1, as shown in Fig. 2 right. Intuitively, bilateral weighting indicates which side of each pixel is more likely to be discontinuous. By iteratively solving for the depth and updating the weights, we can then gradually preserve the discontinuities.

The most important contribution of this paper is the simple but effective strategy for discontinuity preservation in the normal integration problem. The second contribution is that we unify the normal integration problem for orthographic and perspective cases such that the proposed method applies to both without special treatment. As a result, we first realize discontinuity preservation for perspective normal maps.

We structure the remaining of the paper as follows: Section 2 prepares the basic equations for the normal integration problem. Section 3 discusses existing strategies to preserve discontinuities. Sections 4 and 5 then describe and evaluate our bilateral normal integration method, respectively. Finally, Section 6 concludes the paper.

2. Basic Normal Integration Equations

This section establishes the relation between surface normals and surface gradients as a pair of linear partial differential equations (PDEs). The normal integration problem can then be formulated based on the PDEs.

Let $\mathbf{n}(u,v) = [n_x(u,v), n_y(u,v), n_z(u,v)]^\top \in \mathcal{S}^2 \subset \mathbb{R}^3$ be a unit surface normal map and $z(u,v) \in \mathbb{R}$ be a surface, both defined on a closed, bounded, and connected subset Ω on the image plane $(u,v) \in \Omega \subset \mathbb{R}^2$. Consider the vector-valued function $\mathbf{p}(u,v) \in \mathbb{R}^3$ mapping a 2D point (u,v) on the image plane to the 3D point in camera coordinates, whose depth is $z(u,v)$. By definition, the normal vector $\mathbf{n}(u,v)$ is perpendicular to the tangent plane attached to the point $\mathbf{p}(u,v)$, or equivalently, $\mathbf{n}(u,v)$ is perpendicular to the two tangent vectors at $\mathbf{p}(u,v)$:

$$\mathbf{n}^\top \partial_u \mathbf{p} = 0 \quad \text{and} \quad \mathbf{n}^\top \partial_v \mathbf{p} = 0. \tag{1}$$

Here, ∂_u and ∂_v are partial differential operators with respect to u and v respectively, and we omit the dependence on (u,v) for brevity. Because $\mathbf{p}(u,v)$ depends on the camera projection models, we now discuss two common cases: Orthographic and perspective pinhole camera models.

Orthographic case. Under orthographic projection, $\mathbf{p}(u,v) = [u, v, z(u,v)]^\top$. Plugging

$$\partial_u \mathbf{p} = \begin{bmatrix} 1 \\ 0 \\ \partial_u z \end{bmatrix} \quad \text{and} \quad \partial_v \mathbf{p} = \begin{bmatrix} 0 \\ 1 \\ \partial_v z \end{bmatrix} \tag{2}$$

into Eq. (1) results in a pair of PDEs

$$n_z \partial_u z + n_x = 0 \quad \text{and} \quad n_z \partial_v z + n_y = 0. \tag{3}$$

Perspective case. Under perspective projection, $\mathbf{p}(u,v) = z(u,v)\,[(u-c_u)/f, (v-c_v)/f, 1]^\top$, where f is the camera's focal length, and (c_u, c_v) is the principal point on the image plane. Computing the partial derivatives yields

$$\partial_u \mathbf{p} = \begin{bmatrix} \frac{1}{f}((u-c_u)\partial_u z + z) \\ \frac{1}{f}(v-c_v)\partial_u z \\ \partial_u z \end{bmatrix} \quad \text{and} \quad \partial_v \mathbf{p} = \begin{bmatrix} \frac{1}{f}(u-c_u)\partial_v z \\ \frac{1}{f}((v-c_v)\partial_v z + z) \\ \partial_v z \end{bmatrix}. \tag{4}$$

Similar to [15], we introduce an auxiliary variable $\tilde{z}$ satisfying $z = \exp(\tilde{z})$. By chain rule, we have

$$\partial_u z = z \partial_u \tilde{z} \quad \text{and} \quad \partial_v z = z \partial_v \tilde{z}. \tag{5}$$

Plugging Eq. (5) into Eq. (4) leads to

$$\partial_u \mathbf{p} = z \begin{bmatrix} \frac{1}{f}((u-c_u)\partial_u \tilde{z} + 1) \\ \frac{1}{f}(v-c_v)\partial_u \tilde{z} \\ \partial_u \tilde{z} \end{bmatrix} \quad \text{and} \quad \partial_v \mathbf{p} = z \begin{bmatrix} \frac{1}{f}(u-c_u)\partial_v \tilde{z} \\ \frac{1}{f}((v-c_v)\partial_v \tilde{z} + 1) \\ \partial_v \tilde{z} \end{bmatrix}. \tag{6}$$

Further plugging Eq. (6) into Eq. (1) cancels out z. Rearranging the remaining terms yields

$$\begin{cases} (n_x(u-c_u) + n_y(v-c_v) + n_z f)\partial_u \tilde{z} + n_x = 0 \\ (n_x(u-c_u) + n_y(v-c_v) + n_z f)\partial_v \tilde{z} + n_x = 0 \end{cases}. \tag{7}$$

Denoting

$$\tilde{n}_z = n_x(u-c_u) + n_y(v-c_v) + n_z f, \tag{8}$$

Eq. (7) can be simplified as

$$\tilde{n}_z \partial_u \tilde{z} + n_x = 0 \quad \text{and} \quad \tilde{n}_z \partial_v \tilde{z} + n_y = 0, \tag{9}$$

which is in the same form as Eq. (3). We can pre-compute $\tilde{n}_z$ given the normal map and camera parameters. Once $\tilde{z}$ is known, we can exponentiate it to obtain z.

Equations (3) and (9) are rearrangements of the classical PDEs used by existing methods [15]

$$\partial_u z - p = 0 \quad \text{and} \quad \partial_v z - q = 0, \tag{10}$$

where $p = -\frac{n_x}{n_z}$ and $q = -\frac{n_y}{n_z}$. The rearrangements, however, are critical to our methods. We empirically find that our bilateral normal integration works well with both Eqs. (3) and (9), but not with the traditional PDEs (10).

From now on, we will not distinguish between orthographic and perspective cases. Without loss of generality, we will use the notations in Eq. (3) to describe bilateral normal integration in Section 4.

3. Related Work

There have been many works on recovering smooth surfaces from single-view normal maps since the existence of the problem [6], [12]. These works are based on minimizing the functional

$$\min_z \iint_\Omega (\partial_u z - p)^2 + (\partial_v z - q)^2 \, du \, dv. \tag{11}$$

We refer the interested reader to the survey paper [15] on more discussions about solving Eq. (11). This section will mainly discuss two types of strategies for discontinuity preservation.

Robust estimators. As the existence of discontinuities will cause large residuals of the PDEs (10), robust estimator-based methods apply robust functions to the PDEs as

$$\min_z \iint_\Omega \rho(|\partial_u z - p|) + \rho(|\partial_v z - q|) \, du \, dv. \tag{12}$$

By properly designing the ρ-function that can suppress the influence of large residuals, robust estimator-based methods may be able to preserve the discontinuities.

Durou *et al.* [5] propose to use Lorentzian function as the ρ-function, inspired by that the Lorentzian function can keep the sharp edges in the image restoration tasks. Badri *et al.* [2] introduce an intermediate surface, and use four terms in the objective function: one for the data term, and the other three regularizations for robust recovery. They assume the residual of the three regularizations are sparse, therefore their objective can handle the noise and outliers. Quéau and Durou [14] discuss the usage of isotropic total variation and L1 functionals for discontinuity preservation. Robust estimator-based approaches are however, difficult to justify whether the residuals do follow the assumed distributions.

Weighted approaches. Instead of applying robust functions, weighed approaches assign the weights to quadratic residuals of PDEs to eliminates the effects of discontinuities:

$$\min_z \iint_\Omega w_u(\partial_u z - p)^2 + w_v(\partial_v z - q)^2 \, du \, dv. \tag{13}$$

If the quadratic residuals at discontinuity positions are correctly assigned small weights, then the discontinuities are expected to be preserved. In other words, the weights w_u and w_v indicate the discontinuity positions; we will also call the weights discontinuity maps hereafter.

One type of weighted approaches detect the discontinuity positions before optimizing Eq. (13) as a preprocessing step. The major differences among these works are the clues used for discontinuity detection. Saracchini *et al.* [17] requires a user-provided discontinuity map as input. Karacali and Snyder [11] detect the discontinuity based on the residuals of the PDEs (10). Wu and Tang [21] use expectation–maximization algorithm to estimate a discontinuity map from the normal map. Wang *et al.* [19] detect a binary discontinuity map using both photometric stereo images and the normal map. Xie *et al.* [22] handcraft features from the normal map to judge the discontinuity position. These one-time detection-based methods can be fragile as there is no scheme to correct the wrong detection in the optimization afterwards.

A more robust type of weighted approach thus iteratively updates the weights. Alpha-surface method [1] first creates a minimal spanning tree from the integration domain, then iteratively add to the spanning tree the edges that are treated continuous. A threshold alpha is used at each iteration to judge whether an edge can be treated as continuous. Instead of weighting each edge, anisotropic diffusion [1], [16] applies a diffusion tensor to the gradient field to account for the noise and outliers. In [1], the diffusion tensor is computed beforehand by convolving the gradient field. Quéau *et al.* [16] instead model the diffusion tensors as functions of depths. During optimization, the diffusion tensors are iteratively updated based on the currently estimated depths.

Iteratively updating the weights still requires detecting the discontinuity positions. To bypass the detection, Mumford-Shah integrator integrator [16] jointly optimizes for the weights and depths. It assumes discontinuity positions are short curves on the image domain and penalizes the length of curves using regularization on weights. However, we will show that the short discontinuity assumption is difficult in handling complex discontinuities, and the optimized weights can indicate wrong discontinuity positions.

Our method belongs to the weighted approach. But unlike previous methods, we model the discontinuity based on the definition of discontinuity. In addition, we bilaterally weight the quadratic residuals at two sides of each pixel when discretizing the functional.

4. Bilateral Normal Integration

Our goal is to estimate the surface $z(u, v)$ given the normal map $\mathbf{n}(u, v)$ based on the PDEs (3) or (9) in the existence of discontinuity.

4.1 Discontinuity modeling

The PDEs (3) does not hold at discontinuities, as the partial derivative $\partial_u z$ or $\partial_v z$ does not exist. Based on this observation, we propose to minimize the functional

$$\min_z \iint_\Omega b_u(n_z \partial_u z + n_x)^2 + b_v(n_z \partial_v z + n_y)^2 \, du \, dv, \tag{14}$$

with

$$b_u = \begin{cases} 1 & \text{if } \partial_u z \text{ exists,} \\ 0 & \text{o.w.,} \end{cases} \quad b_v = \begin{cases} 1 & \text{if } \partial_v z \text{ exists,} \\ 0 & \text{o.w.} \end{cases} \tag{15}$$

That is, we use two binary weight maps $b_u(u, v)$ and $b_v(u, v)$ to indicate the existence of partial derivatives $\partial_u z$ and $\partial_v z$, respectively. The problem now is that how do we know where $\partial_u z$ or $\partial_v z$ does not exist, *i.e.*, the discontinuity positions. To this end, we directly design the binary weights based on the definition of discontinuity.

Definition 1 (Discontinuity) A function with two variables $z(u, v)$ is discontinuous at a point (u, v) along u-axis if

$$\lim_{h \to 0} z(u + h, v) \neq z(u, v), \tag{16}$$

and is discontinuous along v-axis if

$$\lim_{h\to 0} z(u, v+h) \neq z(u, v). \tag{17}$$

That is, a 2D function is discontinuous at a point if the limit is unequal to the function value. Therefore, the equality between the limit and the function value at a point is a straightforward indicator for discontinuity. Based on the discontinuity definition, we design the weight functions as

$$\begin{aligned} b_u &= \delta[z(u, v) - \lim_{h\to 0} z(u+h, v)] \quad \text{and} \\ b_v &= \delta[z(u, v) - \lim_{h\to 0} z(u, v+h)], \end{aligned} \tag{18}$$

where $\delta(\cdot)$ is a unit impulse function

$$\delta(x) = \begin{cases} 1 & \text{if } x = 0, \\ 0 & \text{o.w.} \end{cases} \tag{19}$$

We can verify that the weight functions Eq. (18) satisfy Eq. (15). At discontinuities, the weights become 0 and the corresponding data terms are ignored in the integral in Eq. (14).

4.2 Bilateral weighting

Based on the functional and the weight functions, we discretize them as a bilaterally weigthed optimization. At each pixel (u, v), data terms from its two sides are relatively weighted (*i.e.*, left v.s. right and upper v.s. lower). To this end, we first approximate the partial derivatives $\partial_u z$ and $\partial_v z$ twice using both forward and backward differences:

$$\begin{aligned} & \partial_u z \approx \Delta_u^+ z = z(u+1, v) - z(u, v) \quad \text{(forward)}, \\ \text{or} \quad & \partial_u z \approx \Delta_u^- z = z(u, v) - z(u-1, v) \quad \text{(backward)}, \\ & \partial_v z \approx \Delta_v^+ z = z(u, v+1) - z(u, v) \quad \text{(forward)}, \\ \text{or} \quad & \partial_v z \approx \Delta_v^- z = z(u, v) - z(u, v-1) \quad \text{(backward)}. \end{aligned} \tag{20}$$

The four approximations leads to four quadratic data terms (*e.g.*, $(n_z \Delta_u^+ z + n_x)^2$) at each pixel. We then weight each data term by the Gaussian function, which approximates the unit impulse function Eq. (19):

$$\delta(x) \approx g(x) = \exp(-kx^2), \tag{21}$$

where k is a hyperparameter controlling the sharpness of Gaussian. To approximate the differences between the function limit and the function value Eq. (18), we again use the forward and backward differences:

$$\begin{aligned} z(u, v) - \lim_{h\to 0} z(u+h, v) &\approx n_z \Delta_u^+ z \quad \text{(forward)}, \\ \text{or} \quad &\approx n_z \Delta_u^- z \quad \text{(backward)}, \\ z(u, v) - \lim_{h\to 0} z(u, v+h) &\approx n_z \Delta_v^+ z \quad \text{(forward)}, \\ \text{or} \quad &\approx n_z \Delta_v^- z \quad \text{(backward)}, \end{aligned} \tag{22}$$

which leads to Gaussian weights (*e.g.* $g(n_z \Delta_u^+ z)$) at the corresponding side of each pixel. With the discretizations, we can now present the bilaterally weighted optimization as

$$\begin{aligned} \min_z \sum_{(u,v)\in\Omega} & \frac{1}{w_u}[g(n_z\Delta_u^+ z)(n_z\Delta_u^+ z + n_x)^2 \\ & + g(n_z\Delta_u^- z)(n_z\Delta_u^- z + n_x)^2] \\ + & \frac{1}{w_v}[g(n_z\Delta_v^+ z)(n_z\Delta_v^+ z + n_y)^2 \\ & + g(n_z\Delta_v^- z)(n_z\Delta_v^- z + n_y)^2]. \end{aligned} \tag{23}$$

Here, we normalize the weights at two sides of each pixel such that they sum to 1. That is,

$$\begin{aligned} w_u &= g(n_z\Delta_u^+ z) + g(n_z\Delta_u^- z), \\ w_v &= g(n_z\Delta_v^+ z) + g(n_z\Delta_v^- z). \end{aligned} \tag{24}$$

The benefit of bilateral weighting is that the optimization performs more robust at steep regions on the surface. For example, at smooth but steep regions, $\Delta_u^+ z$ and $\Delta_u^- z$ can be both large and yield small Gaussian values. Without bilateral weighting, the steep surface can be treated as discontinuous. After the normalization, the weights at two sides of each pixel are still close to 0.5 as long as there is no discontinuity besides the pixel. In other words, the bilateral weighting judges which side of each pixel is more likely to be discontinuous. In fact, the normalization by w_u and w_v yields sigmoid weight functions. For example,

$$\begin{aligned} \frac{g(n_z\Delta_u^+ z)}{w_u} &= \sigma((n_z\Delta_u^- z)^2 - (n_z\Delta_u^+ z)^2), \\ \frac{g(n_z\Delta_u^- z)}{w_u} &= \sigma((n_z\Delta_u^+ z)^2 - (n_z\Delta_u^- z)^2), \end{aligned} \tag{25}$$

where $\sigma(x) = \frac{1}{1+\exp(-kx)}$ is the sigmoid function and k controls its sharpness. It follows from Eq. (25) that when $(n_z\Delta_u^+ z)^2$ and $(n_z\Delta_u^- z)^2$ are approximately same, the weights at the two sides of a pixel are both close to 0.5. Only when one of $(n_z\Delta_u^+ z)^2$ and $(n_z\Delta_u^- z)^2$ is significantly larger than the other, the weight at the larger side becomes close to 0, and that side is treated discontinuous.

4.3 Numerical solution method

With the discretization, we now describe the optimization problem in the matrix form and its solution method. For now, let us ignore the weights in Eq. (23). We can form a sparse linear system from Eq. (23) as

$$\begin{bmatrix} \mathbf{N}_z\mathbf{D}_u^+ \\ \mathbf{N}_z\mathbf{D}_u^- \\ \mathbf{N}_z\mathbf{D}_v^+ \\ \mathbf{N}_z\mathbf{D}_v^- \end{bmatrix} \mathbf{z} = \begin{bmatrix} -\mathbf{n}_x \\ -\mathbf{n}_x \\ -\mathbf{n}_y \\ -\mathbf{n}_y \end{bmatrix} \quad \text{or} \quad \mathbf{A}\mathbf{z} = \mathbf{b} \tag{26}$$

for short. Here, $\mathbf{z} \in \mathbb{R}^m$ is the vectorized depth map, $\mathbf{n}_x$, $\mathbf{n}_y$, and $\mathbf{n}_z$ are the stacking vector of $n_x(u, v)$, $n_y(u, v)$, and $n_z(u, v)$ at all pixels, respectively; $\mathbf{N}_z = \mathrm{diag}(\mathbf{n}_z) \in \mathbb{R}^{m\times m}$. The four square matrices $\mathbf{D}_u^+$, $\mathbf{D}_u^-$, $\mathbf{D}_v^+$, and $\mathbf{D}_v^-$ are $m \times m$ discrete partial derivative matrices. Each row either contains only two nonzero entries: -1 and 1, or is a zero vector; see [16] for details.

The system matrix $\mathbf{A}$ is rank 1 deficient, implying an offset ambiguity in the result. In the perspective case, the offset ambiguity becomes scale ambiguity after we exponentiating the result.

Now, we stack the weights in Eq. (23) into a diagonal matrix. Let $\mathbf{x}^{\circ 2}$ be the elementwise square of a vector $\mathbf{x}$, *i.e.*,

$\mathbf{x}^{\circ 2} = (\mathbf{x})\mathbf{x}$, and $\sigma[\mathbf{x}; k]$ be an elementwise Sigmoid function with a hyper-parameter k. We can stack the weights at all pixels into two $m \times m$ diagonal matrices as

$$\begin{aligned}\mathbf{W}_u(\mathbf{z}) &= \mathrm{diag}(\sigma[(\mathbf{N}_z\mathbf{D}_u^-\mathbf{z})^{\circ 2} - (\mathbf{N}_z\mathbf{D}_u^+\mathbf{z})^{\circ 2}]),\\ \mathbf{W}_v(\mathbf{z}) &= \mathrm{diag}(\sigma[(\mathbf{N}_z\mathbf{D}_v^-\mathbf{z})^{\circ 2} - (\mathbf{N}_z\mathbf{D}_v^+\mathbf{z})^{\circ 2}]).\end{aligned} \tag{27}$$

Further stacking all weights into a $4m \times 4m$ diagonal matrix yields

$$\mathbf{W}(\mathbf{z}) = \begin{bmatrix} \mathbf{W}_u(\mathbf{z}) & & & \\ & \mathbf{I}-\mathbf{W}_u(\mathbf{z}) & & \\ & & \mathbf{W}_v(\mathbf{z}) & \\ & & & \mathbf{I}-\mathbf{W}_v(\mathbf{z}) \end{bmatrix}, \tag{28}$$

where $\mathbf{I}$ is an $m \times m$ identity matrix. The matrix form of our optimization problem Eq. (23) is then

$$\min_{\mathbf{z}} (\mathbf{A}\mathbf{z} - \mathbf{b})^\top \mathbf{W}(\mathbf{z})(\mathbf{A}\mathbf{z} - \mathbf{b}). \tag{29}$$

Solution method. Equation (29) is a non-convex problem, and we use iteratively re-weighted least squares (IRLS) to solve it. At each step, we fix the weight matrix $\mathbf{W}$ and solve for the depths $\mathbf{z}$

$$\mathbf{z}^{(t+1)} = \operatorname*{argmin}_{\mathbf{z}} (\mathbf{A}\mathbf{z} - \mathbf{b})^\top \mathbf{W}(\mathbf{z}^{(t)})(\mathbf{A}\mathbf{z} - \mathbf{b}). \tag{30}$$

When $\mathbf{W}$ is fixed, Eq. (30) boils down to a convex weighted least squares problem. We can find $\mathbf{z}^{(t+1)}$ by solving its normal equation

$$\mathbf{A}^\top \mathbf{W}(\mathbf{z}^{(t)})\mathbf{A}\mathbf{z}^{(t+1)} = \mathbf{A}^\top \mathbf{W}(\mathbf{z}^{(t)})\mathbf{b}. \tag{31}$$

In our implementation, we use conjugate gradient method to solve Eq. (31).

We initialize $\mathbf{z}^{(0)}$ as a plane, or equivalently, initialize all weights as 0.5. Then $\mathbf{z}^{(1)}$ becomes the least squares approximate solution to Eq. (26). We stop the iteration if either the maximum iteration has been reached or the relative energy has been smaller than the tolerance. In our experiments, we set the maximum iteration around 100 and the stopping tolerance as 1×10^{-5}.

5. Experiments

We have presented bilateral normal integration to tackle the discontinuity preservation problem in normal integration. In this section, we will evaluate our method and compare it with existing discontinuity preserving normal integration methods.

Evaluation metrics. To quantitatively evaluate the integrated surfaces when GT surfaces are available, we show the absolute depth error maps and report the mean absolute depth error (MADE) between the integrated and GT surfaces. As there is an offset ambiguity in surfaces integrated from orthographic normal maps, we posterior shift the integrated surfaces such that the L1 norm between the shifted and the GT surfaces is minimal. We find the optimal offset by computing the median difference between the estimated and the GT depth values.

In the perspective case, there is a scale ambiguity in the integrated surfaces. Similarly, we scale the integrated surface such that the L1 norm between the scaled and the GT surfaces is minimal. We find the optimal scale by computing the median scale between the estimated and the GT depth values. As the objects can have various sizes, we also report the range of the object, *i.e.*, the difference between the maximum and the minimum depth values of the surface.

5.1 Method analysis

In this section, we analyze our method fr om different aspects. We inspect the effects of the hyper-parameter, show the evolution over the iteration, inspect the effect of bilateral weighting, and discuss the limitation of our method.

Effects of the hyperparameter. We first inspect the effect of the hyper-parameter k on the results. We test our method on two classical yet challenging toy surfaces [16] "Vase" and "Tent", whose normal maps can be analytically computed.

Figure 3 shows the integration results with different ks. It can be seen that when k is small, like $k = 0.1$, the integrated surfaces appear smooth; when k is very large, like $k > 100$, unwanted discontinuities exist. This is because k controls the sensitivity to depth differences between adjacent pixels. A smaller k gives a smoother Gaussian function; even if the squared depth differences between adjacent pixels is relatively large, the Gaussian function still takes a value close to 1. An extreme case is when $k = 0$, the Gaussian function takes the constant value 1. In this extreme case, our method degrades to Eq. (26), with which we can only recover a smooth surface. In contrast, a larger k brings a sharper Gaussian function; a tiny squared depth difference between adjacent pixels can be mapped to a value close to 0. This may yield unwanted discontinuities. Therefore we can see when k is large, the integrated surfaces appear over-segmented, especially around the discontinuities.

Figure 4 shows the surface evolution over the iteration. Our method gradually recovers the discontinuities, and the energy $\mathbf{J}^{(t)}$ steadily decreases over the iteration.

Effects of bilateral weighting. We now inspect the effects of bilateral weighting. On the positive side, bilateral weighting avoids the mis-segmentation problem. As shown in Fig. 5 top, the surfaces estimated without bilateral weighting are likely to be wrongly segmented at steep regions. Because bilateral weighting relatively judges the existence of discontinuities at two sides of a pixel, mis-segmentation is not likely to occur at steep regions.

On the negative side, bilateral weighting cannot handle the case where both sides of a pixel are discontinuous. As shown in Fig. 5 bottom, bilateral weighting cannot preserve the discontinuities when they are one pixel away. This is because bilateral weighting requires at least one side of each pixel to be smooth. Nevertheless, we still recommend bilateral weighting, as one-pixel wide discontinuities are not likely to occur in practice. In the remaining experiments, we will also use bilateral weighting.

Limitation. There is one case that our method cannot identify the discontinuity positions correctly: the partial derivatives of the surface happen to be continuous across depth discontinuities. As shown in Fig. 6, our method fails at identifying the discontinuity positions in the planar region of the surface. In such case, additional information other than the normal map is required to identify the discontinuity positions.

5.2 Comparison

To demonstrate the effectiveness of our method, this section compares our method with existing methods on orthographic and perspective normal maps. We compare our method with a smooth surface recovery method [3], the Mumford-Shah integrator [16]

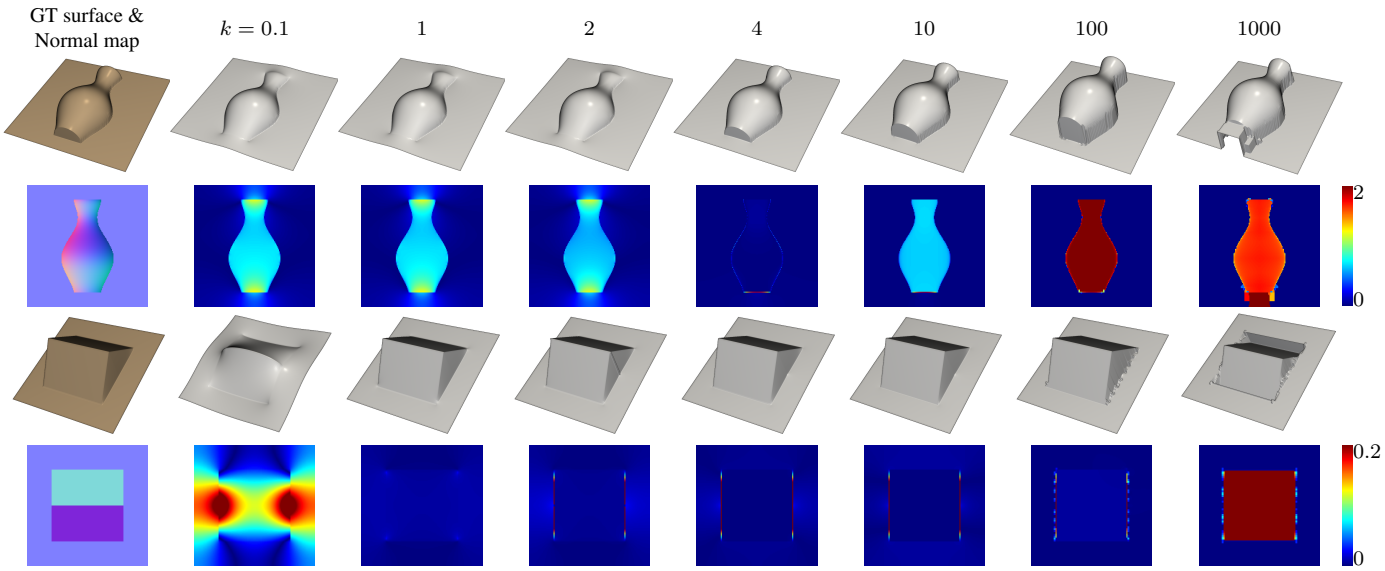

Fig. 3 The effects of the hyperparameter k. The odd and even rows display integrated surfaces and depth error maps, respectively. The hyper-parameter k controls the sharpness of the Gaussian function. Small k yields smooth surfaces; large k causes over-segmentation of the surfaces.

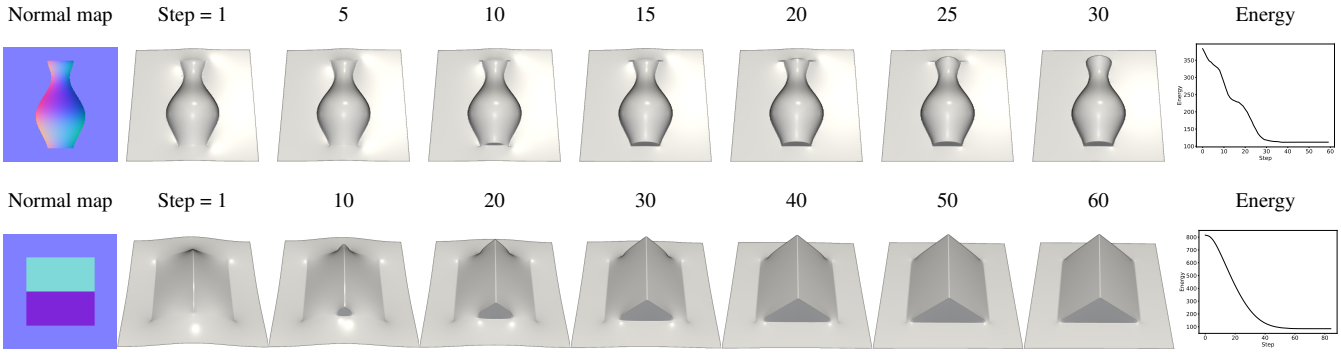

Fig. 4 Our method gradually preserves the discontinuities during the iteration. The energy of the objective function steadily decreases.

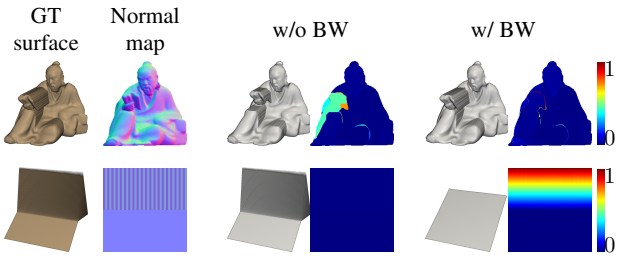

Fig. 5 Bilateral weighting (BW) has both positive and negative effects. **(Top)** Bilateral weighting avoids wrong segmentation at steep regions. **(Bottom)** A surface with multiple one-pixel wide stripes. One pixel wide discontinuities cannot be preserved as bilateral weighting requires at least one side of each pixel to be continuous.

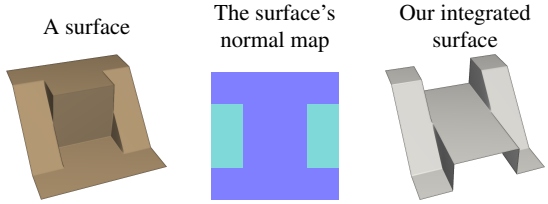

Fig. 6 An example of failure case. Our method cannot correctly identify the discontinuity positions when partial derivatives happen to be continuous across the depth discontinuities.

and the anisotropic diffusion integrator [16]. The smooth recovery method [3] does not preserve discontinuities and serves as a baseline method. Both the Mumford-Shah and the anisotropic diffusion integrator [16] aim at discontinuity preservation and have two hyperparameters in their objective functions. We follow the recommendation in [16] and set the hyperparameters in Mumford-Shah integrator as $\mu = 45$ and $\epsilon = 0.001$, and the hyperparameters in anisotropic diffusion as $\mu = 0.2$ and $\nu = 10$. For the hyperparameter of our method, we set $k = 2$.

Figure 7 shows quantitative comparisons on an analytically computed and a rendered normal map, whose ground truth surfaces are available. We rendered the normal map and the corresponding surface from the object "Reading" in DiLiGenT-MV dataset [13] by Mitsuba renderer [9] with an orthographic camera. For both surfaces, the smooth recovery method [3] cannot recover discontinuities. Mumford-Shah integrator [16] appears to be difficult at identifying the discontinuity positions correctly. Moreover, unwanted discontinuities exist. This is because Mumford-Shah integrator [16] only put regularization on the discontinuity maps but does not relate the discontinuity to the depth. Consequently, Mumford-Shah integrator [16] can recover discontinuities at wrong positions. The anisotropic diffusion integrator [16]

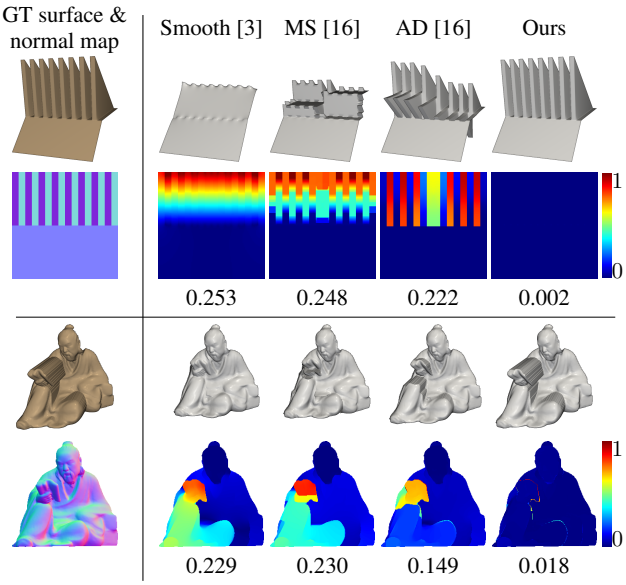

Fig. 7 Quantitative comparison on an analytically computed and a Mitsuba-rendered surface. The "Smooth" method [3] assumes a smooth surface and does not preserve discontinuities; "MS" and "AD" are short for Mumford-Shah and anisotropic diffusion integrator [16], respectively. The odd and even rows display the recovered surfaces and absolute depth error maps, respectively. Numbers underneath are mean absolute depth errors.

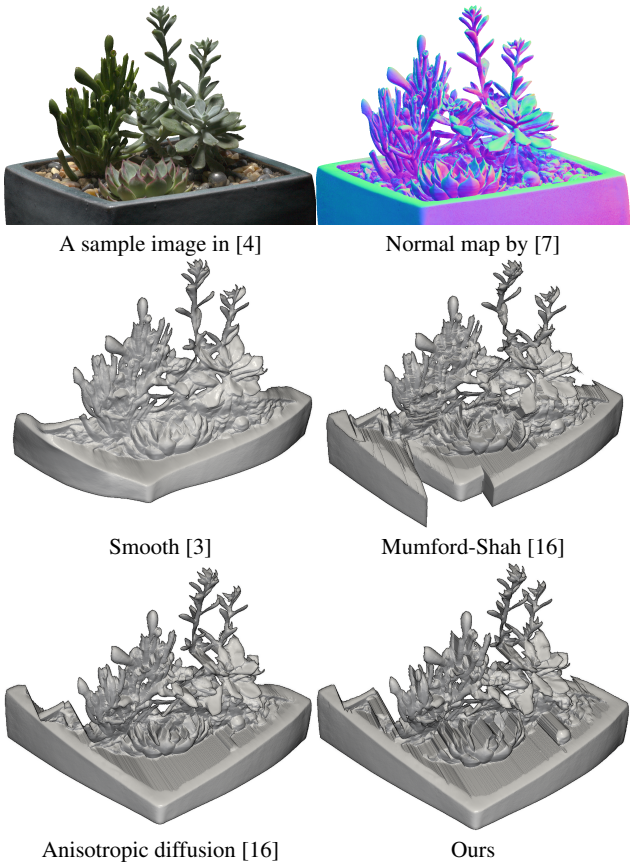

Fig. 8 Qualitative comparison of integrated surfaces from a real-world normal map. The normal map is estimated by the photometric stereo method CNN-PS [7] on real-world image observations from Light Stage Data Gallery [4]. The data gallery does not provide GT normals or surfaces.

recovers better surfaces than Mumford-Shah integrator. However, it appears difficult at recovering large depth gaps even if the discontinuity positions are correctly identified, as shown in the second object. Our method, on the other hand, can identify more accurate discontinuity positions and recover depth gaps more accurately. As a result, we achieve the smallest MADEs among all compared methods.

Figure 8 shows a qualitative comparison on a real-world normal map. The normal map is estimated by the photometric stereo method CNN-PS [7] on the real-world images from Light Stage Data Gallery [4]. It can be clearly seen that Mumford-Shah integrator [16] wrongly identifies the discontinuity positions and over-segments the surface. The anisotropic diffusion integrator [16] identifies reasonable discontinuity positions but is not good at preserving large depth gaps. On the other hand, our method identifies the discontinuity positions and preserves large depth gaps more reasonably.

Figure 9 shows integration results on the objects from DiLiGenT benchmark [18]. We use the ground truth perspective normal maps as inputs and set $k = 2$ for all objects. We find that existing methods (*e.g.*, Mumford-Shah and anisotropic diffusion [16]) do not preserve discontinuities for perspective normal maps. Therefore, we only compare our method with the smooth recovery method [3]. All surfaces contain discontinuities; thus, the smooth recovery method [3] yields heavily distorted surfaces, as can be seen from the depth error maps. On the other hand, our method preserves the discontinuities and thus reduces the distortions in the integrated surfaces. The MADEs are generally within 1 mm; especially, we achieve 0.07 mm MADE for the object "Cow". To our knowledge, we are the first to show working results that preserve discontinuities for perspective normal maps.

However, it can be seen from Fig. 9 that our method does not achieve small MADEs on the objects "Goblet" and "Harvest" compared to other objects. The large MADEs are caused by the intrinsic difficulty of single-view normal integration. In the two objects, the integration domain is separated by discontinuities into multiple disjoint regions. Consequently, multiple scale ambiguities arise between disjoint regions, which cannot be solved given only single-view normal maps. Nevertheless, our method is accurate up to scales between disjoint regions. The surface is barely distorted near discontinuities thanks to that our method can at least identify the discontinuity positions. This can be confirmed from the piece-wise uniform depth error map of "Goblet".

6. Conclusion

We have presented and evaluated bilateral normal integration for discontinuity preserving surface recovery from normal maps. The success of our method relies on the usage of depth differences between adjacent pixels to indicate the existence of discontinuities. Compared to the previous method, our method can handle more challenging discontinuities on a surface. In addition, we unify the problem formulation for orthographic and perspective normal maps in an appropriate way. As a result, we have first shown discontinuity preserving results on perspective normal maps.

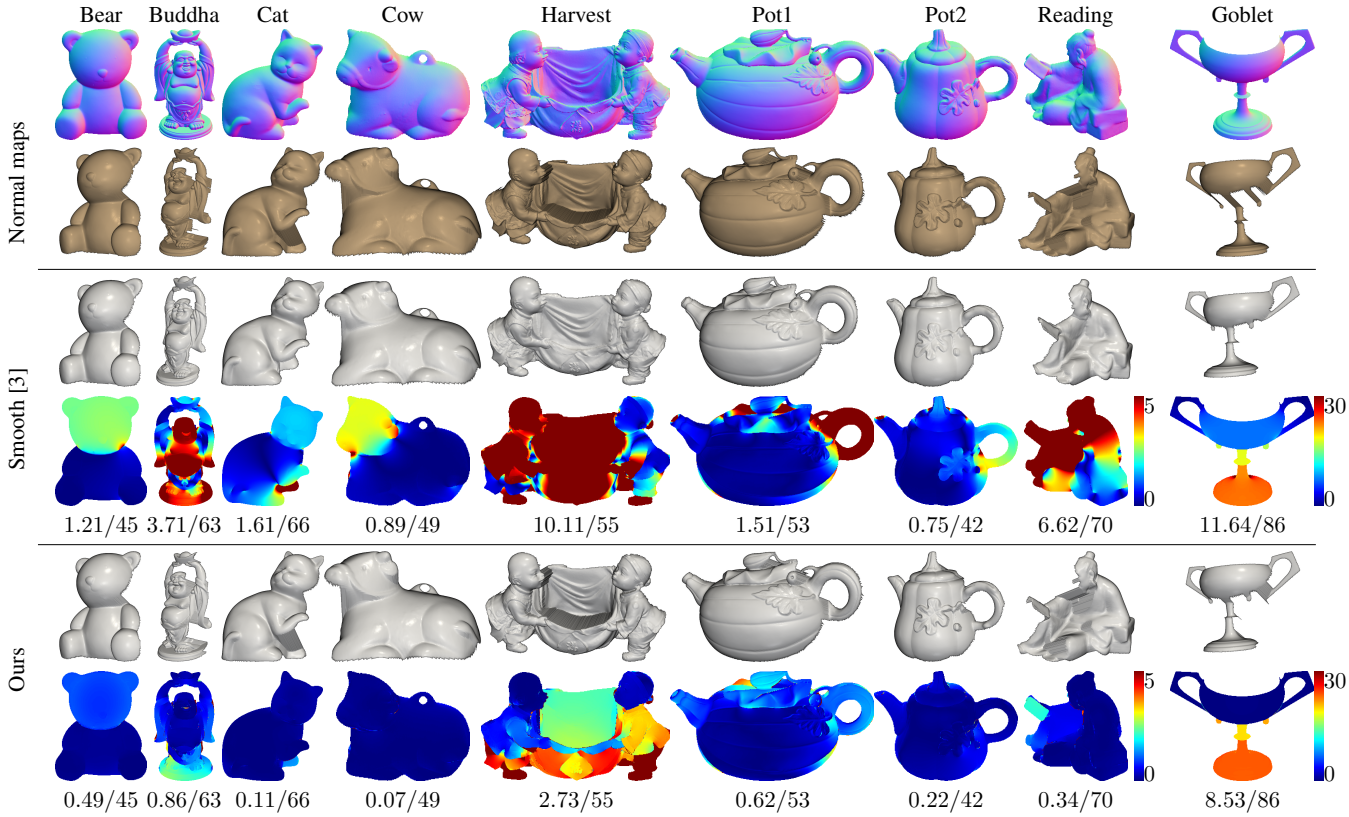

Fig. 9 Quantitative comparison on DiLiGenT benchmark [18]. **(1st & 2nd rows)** The input perspective normal maps and the ground truth surfaces. Viewpoints are adjusted to emphasize discontinuities. **(3rd & 4th rows)** Surfaces integrated by the smooth recovery method [3] and corresponding depth error maps. The colormap scale is the same for the first eight objects. Numbers underneath are mean absolute depth errors / ranges of the objects [mm]. **(5th & 6th rows)** Surfaces integrated by our method and corresponding depth error maps.

References

[1] Agrawal, A., Raskar, R. and Chellappa, R.: What is the range of surface reconstructions from a gradient field?, *Proc. of European Conference on Computer Vision (ECCV)* (2006).
[2] Badri, H., Yahia, H. and Aboutajdine, D.: Robust surface reconstruction via triple sparsity, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2283–2290 (2014).
[3] Cao, X., Shi, B., Okura, F. and Matsushita, Y.: Normal Integration via Inverse Plane Fitting With Minimum Point-to-Plane Distance, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2382–2391 (2021).
[4] Chabert, C.-F., Einarsson, P., Jones, A., Lamond, B., Ma, W.-C., Sylwan, S., Hawkins, T. and Debevec, P.: Relighting human locomotion with flowed reflectance fields, *ACM SIGGRAPH 2006 Sketches*, pp. 76–es (2006).
[5] Durou, J.-D., Aujol, J.-F. and Courteille, F.: Integrating the normal field of a surface in the presence of discontinuities, *Workshops of the Computer Vision and Pattern Recognition (CVPRW)* (2009).
[6] Horn, B. K. and Brooks, M. J.: The variational approach to shape from shading., *Comput. Vis. Graph. Image Process.* (1986).
[7] Ikehata, S.: CNN-PS: CNN-based photometric stereo for general non-convex surfaces, *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–18 (2018).
[8] Ikeuchi, K. and Horn, B. K.: Numerical shape from shading and occluding boundaries, *Artificial intelligence* (1981).
[9] Jakob, W.: Mitsuba renderer (2010).
[10] Kadambi, A., Taamazyan, V., Shi, B. and Raskar, R.: Polarized 3D: High-quality depth sensing with polarization cues, *Proc. of International Conference on Computer Vision (ICCV)* (2015).
[11] Karacali, B. and Snyder, W.: Reconstructing discontinuous surfaces from a given gradient field using partial integrability, *Computer Vision and Image Understanding* (2003).
[12] Kovesi, P.: Shapelets correlated with surface normals produce surfaces, *Proc. of International Conference on Computer Vision (ICCV)* (2005).
[13] Li, M., Zhou, Z., Wu, Z., Shi, B., Diao, C. and Tan, P.: Multi-view photometric stereo: a robust solution and benchmark dataset for spatially varying isotropic materials, *IEEE Transactions on Image Processing* (2020).
[14] Quéau, Y. and Durou, J.-D.: Edge-Preserving Integration of a Normal Field: Weighted Least-Squares, TV and L1 Approaches, *Proc. of International Conference on Scale Space and Variational Methods in Computer Vision* (2015).
[15] Quéau, Y., Durou, J.-D. and Aujol, J.-F.: Normal integration: a survey, *Journal of Mathematical Imaging and Vision* (2018).
[16] Quéau, Y., Durou, J.-D. and Aujol, J.-F.: Variational methods for normal integration, *Journal of Mathematical Imaging and Vision* (2018).
[17] Saracchini, R. F., Stolfi, J., Leitão, H. C., Atkinson, G. A. and Smith, M. L.: A robust multi-scale integration method to obtain the depth from gradient maps, *Computer Vision and Image Understanding*, Vol. 116, No. 8, pp. 882–895 (2012).
[18] Shi, B., Mo, Z., Wu, Z., Duan, D., Yeung, S.-K. and Tan, P.: A Benchmark Dataset and Evaluation for Non-Lambertian and Uncalibrated Photometric Stereo, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* (2019).
[19] Wang, Y., Bu, J., Li, N., Song, M. and Tan, P.: Detecting discontinuities for surface reconstruction, *Proc. of International Conference on Pattern Recognition (ICPR)*, IEEE, pp. 2108–2111 (2012).
[20] Woodham, R. J.: Photometric stereo: A reflectance map technique for determining surface orientation from image intensity, *Proc. of Image Understanding Systems and Industrial Applications I* (1979).
[21] Wu, T.-P. and Tang, C.-K.: Visible surface reconstruction from normals with discontinuity consideration, *Proc. of Computer Vision and Pattern Recognition (CVPR)*, Vol. 2, IEEE, pp. 1793–1800 (2006).
[22] Xie, W., Wang, M., Wei, M., Jiang, J. and Qin, J.: Surface Reconstruction From Normals: A Robust DGP-Based Discontinuity Preservation Approach, *Proc. of Computer Vision and Pattern Recognition (CVPR)* (2019).