2022 International Conference on 3D Vision (3DV) | 978-1-6654-5670-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/3DV57658.2022.00060

SDA-SNE: Spatial Discontinuity-Aware Surface Normal Estimation via Multi-Directional Dynamic Programming

Nan Ming, Yi Feng, Rui Fan*
MIAS Group, Robotics & Artificial Intelligence Laboratory,
Department of Control Science & Engineering,
Tongji University, Shanghai 201804, China
{nan.ming, fengyi, rui.fan}@ieee.org

Abstract

The state-of-the-art (SoTA) surface normal estimators (SNEs) generally translate depth images into surface normal maps in an end-to-end fashion. Although such SNEs have greatly minimized the trade-off between efficiency and accuracy, their performance on spatial discontinuities, e.g., edges and ridges, is still unsatisfactory. To address this issue, this paper first introduces a novel multi-directional dynamic programming strategy to adaptively determine inliers (co-planar 3D points) by minimizing a (path) smoothness energy. The depth gradients can then be refined iteratively using a novel recursive polynomial interpolation algorithm, which helps yield more reasonable surface normals. Our introduced spatial discontinuity-aware (SDA) depth gradient refinement strategy is compatible with any depth-to-normal SNEs. Our proposed SDA-SNE achieves much greater performance than all other SoTA approaches, especially near/on spatial discontinuities. We further evaluate the performance of SDA-SNE with respect to different iterations, and the results suggest that it converges fast after only a few iterations. This ensures its high efficiency in various robotics and computer vision applications requiring real-time performance. Additional experiments on the datasets with different extents of random noise further validate our SDA-SNE's robustness and environmental adaptability. Our source code, demo video, and supplementary material are publicly available at mias.group/SDA-SNE.

1. Introduction

Surface normal is an informative 3D visual feature used in various computer vision and robotics applications, such as object recognition and scene understanding [5, 9, 13]. To date, there has not been extensive research on surface normal estimation, as it is merely considered to be an auxiliary functionality for other computer vision and robotics applications [7]. As such applications are typically required to perform robustly and in real time, surface normal estimators (SNEs) must be sufficiently accurate and computationally efficient [7].

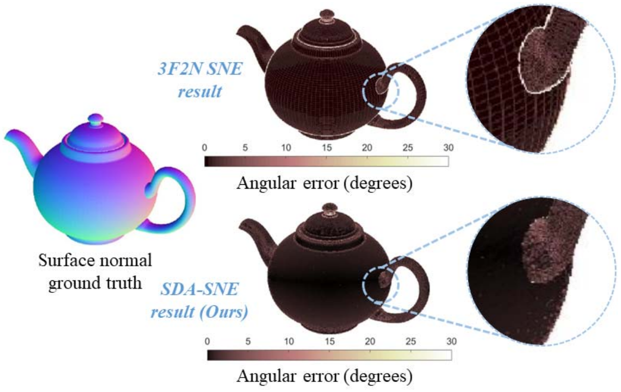

Figure 1. Comparison between 3F2N (recently published SoTA SNE) [7] and our proposed SDA-SNE. A significantly improved region is marked with a dashed blue circle.

The state-of-the-art (SoTA) SNEs [2, 4, 7, 12–14, 16, 19] generally select a set of 3D points and compute surface normals via planar fitting, geometric transformation, or statistical analysis. However, such approaches are infeasible to estimate surface normals near/on spatial discontinuities, *e.g.*, edges and ridges (see Fig. 1), as they generally introduce adjacent 3D points on different surfaces unintentionally [10]. Bormann *et al.* [3] proposed the first edge-aware SNE, capable of adaptively selecting reasonable adjacent 3D points on the same surface. It performs significantly better than prior arts near edges. Nonetheless, their method focuses only on edges and requires a manually-set discontinuity awareness threshold. This results in low adaptability to different scenarios and datasets. Hence, there is a strong

*Corresponding author

978-1-6654-5670-8/22/$31.00 ©2022 IEEE
DOI 10.1109/3DV57658.2022.00060

Authorized licensed use limited to: TONGJI UNIVERSITY. Downloaded on March 04,2023 at 09:03:44 UTC from IEEE Xplore. Restrictions apply.

motivation to develop an SNE capable of tackling all types of spatial discontinuities to achieve greater robustness and environmental adaptability.

The major contributions of this work are summarized as follows:
(1) **Spatial discontinuity-aware surface normal estimator (SDA-SNE)**, a highly accurate SNE, significantly outperforming all other SoTA SNEs, especially near/on spatial discontinuities.
(2) **Multi-directional dynamic programming (DP)** for depth gradient refinement. It iteratively introduces inliers (co-planar pixels) by minimizing a smoothness energy.
(3) **Path discontinuity (PD) norm**, a semi-norm evaluating the discontinuity of a given path. It reflects the depth gradient estimation error generated by the Taylor expansion.
(4) **Recursive polynomial interpolation (RPI)**, an ultrafast polynomial interpolation algorithm. Compared to Lagrange and Newton polynomial interpolation, it iteratively refines the first-order derivative of depth with a more efficient polynomial interpolation strategy.

2. Related Work

The existing SoTA SNEs can be categorized into four classes: (1) optimization-based [13], (2) averaging-based [14], (3) affine-correspondence-based [2], and (4) depth-to-normal [23].

The optimization-based SNEs, *e.g.*, PlaneSVD [14], PlanePCA [12], VectorSVD [14], and QuadSVD [17, 22], compute surface normals by fitting local planar or curved surfaces to an observed 3D point cloud, using either singular value decomposition (SVD) or principal component analysis (PCA). The averaging-based SNEs, *e.g.*, AreaWeighted [13] and AngleWeighted [14], estimate surface normals by computing the weighted average of the normal vectors of the triangles formed by each given 3D point and its neighbors. However, these two categories of SNEs are highly computationally intensive and unsuitable for online robotics and computer vision applications [7]. The affine-correspondence-based SNEs exploit the relationship between affinities and surface normals [2, 4, 19]. Nevertheless, such SNEs are developed specifically for stereo or multi-view cases and cannot generalize to other cases where monocular depth images are used.

Recently, enormous progress has been made in end-to-end depth-to-normal translation [6, 7, 16]. Such SNEs have demonstrated superior performance in terms of both speed and accuracy. Fan *et al.* [7] proposed a fast and accurate SNE referred to as 3F2N, which can infer surface normal information directly from depth or disparity images, with two gradient filters and one central tendency measurement filter (a mean or median filtering operation), as follows:

$$\begin{aligned} & n_x = \partial d/\partial u, \qquad n_y = \partial d/\partial v, \\ & n_z = -\Phi\left\{\frac{(x_i - x)n_x + (y_i - y)n_y}{z_i - z}\right\},\ i = 1, \dots, m, \end{aligned} \tag{1}$$

where $\mathbf{n} = [n_x, n_y, n_z]^\top$ is the surface normal of a given 3D point $\mathbf{p}^C = [x, y, z]^\top$ in the camera coordinate system, $\mathbf{p}^C$ is projected to a 2D pixel $\mathbf{p} = [u, v]^\top$ via $z[\mathbf{p}^\top, 1]^\top = \mathbf{K}\mathbf{p}^C$ ($\mathbf{K}$ is the camera intrinsic matrix), $\mathbf{p}_i^C = [x_i, y_i, z_i]^\top$ is one of the m adjacent pixels of $\mathbf{p}^C$, d is disparity (or inverse depth), and $\Phi\{\cdot\}$ represents the central tendency measurement filtering operation for n_z estimation. Nakagawa *et al.* [16] presented an SNE based on cross-product of two tangent vectors (CP2TV): $\mathbf{n}(u, v) = \boldsymbol{t}_u \times \boldsymbol{t}_v$, where

$$\begin{aligned} \boldsymbol{t}_u &= \left[\frac{\partial x}{\partial u}, \frac{\partial y}{\partial u}, \frac{\partial z}{\partial u}\right]^\top \\ \boldsymbol{t}_v &= \left[\frac{\partial x}{\partial v}, \frac{\partial y}{\partial v}, \frac{\partial z}{\partial v}\right]^\top, \end{aligned} \tag{2}$$

$$\begin{aligned} \frac{\partial x}{\partial u} &= \frac{z}{f_u} + \frac{u - c_u}{f_u}\frac{\partial z}{\partial u} \\ \frac{\partial y}{\partial u} &= \frac{v - c_v}{f_v}\frac{\partial z}{\partial u} \\ \frac{\partial x}{\partial v} &= \frac{u - c_u}{f_u}\frac{\partial z}{\partial v} \\ \frac{\partial y}{\partial v} &= \frac{z}{f_v} + \frac{v - c_v}{f_v}\frac{\partial z}{\partial v}, \end{aligned} \tag{3}$$

$\mathbf{c} = [c_u, c_v]^\top$ is the principal point (in pixels), and f_u and f_v are the camera focal lengths (in pixels) in the u and v directions, respectively. Since the depth or disparity gradient estimation has a direct impact on the surface normal quality, this paper focuses thoroughly on the strategy to improve the accuracy of depth gradient $\nabla\hat{z} = [\hat{z}_u, \hat{z}_v]^{\top}$[1]. The proposed strategy can also be applied to improve disparity gradient estimation, as it is inversely proportional to depth.

3. Methodology

As discussed in Sec. 2, surface normal estimation was formulated as a depth-to-normal translation problem in SoTA approaches, which have demonstrated superior performances over other methods in terms of both speed and accuracy [7]. The accuracy of such approaches is subject to the depth gradient quality. Since depth gradients typically jump near/on discontinuities, *e.g.*, edges and ridges, the estimated surface normals near/on such discontinuities are always significantly different from their ground truth. Therefore, we propose a multi-directional DP strategy to translate

[1] In this paper, the variables with and without hat symbols denote the estimated and theoretical values, respectively.

Authorized licensed use limited to: TONGJI UNIVERSITY. Downloaded on March 04,2023 at 09:03:44 UTC from IEEE Xplore. Restrictions apply.

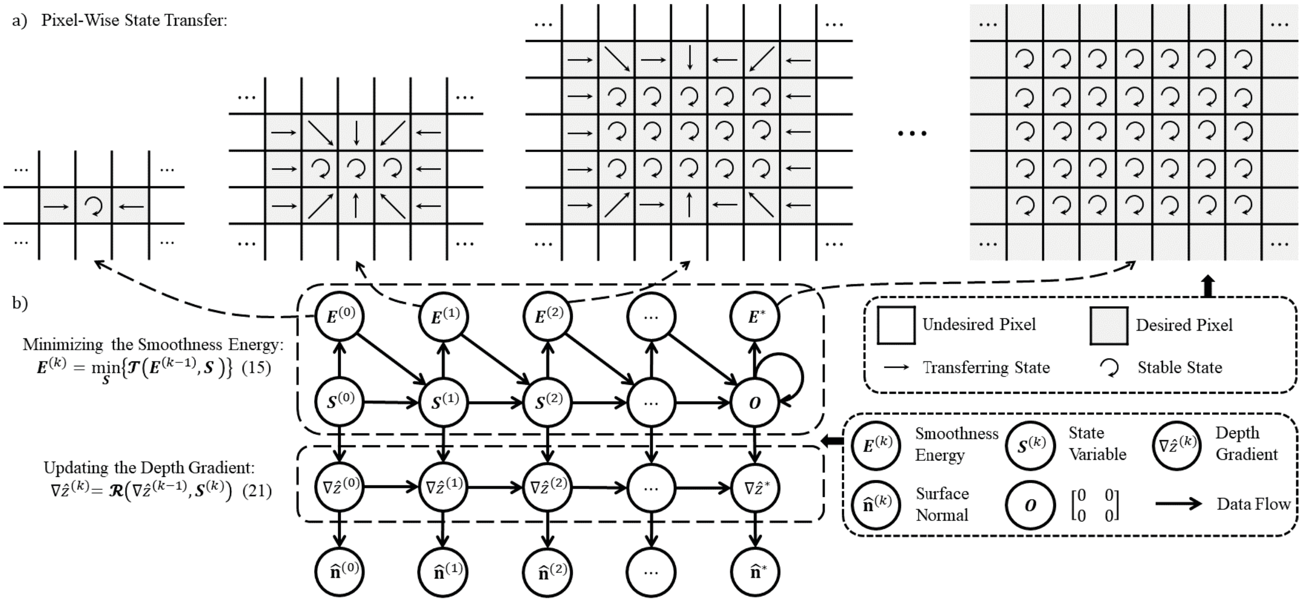

Figure 2. (a) An illustration of the energy transfer process, where the inliers (desired pixels) are gradually introduced in each iteration; (b) The relationship among smoothness energy, state variable, depth gradient, and surface normal.

a depth or disparity image into a surface normal map in a coarse-to-fine manner. It first initializes $\nabla\hat{z}$ using finite difference interpolation and then optimizes $\nabla\hat{z}$ using our proposed multi-directional DP. Accurate surface normals can then be obtained by plugging the optimum depth gradients into either the central tendency measurement filtering operation used in 3F2N (see (1)) or the cross-product operation used in CP2TV (see (2)-(3)).

3.1. Multi-Directional Dynamic Programming

3.1.1 Smoothness Energy and State Variable

As a general rule, only inliers (co-planar pixels) should be introduced when estimating the depth gradient of an observed pixel $\mathbf{p}$. Such pixels are determined by minimizing the (path) smoothness energy $\boldsymbol{E} = [E_u, E_v]$ of $\mathbf{p}$ using DP, where E_u and E_v represent the horizontal and vertical components of $\boldsymbol{E}$, respectively. To minimize the computational complexity, we recursively introduce adjacent co-planar pixels. In each iteration, two desired pixels $\mathbf{p}'_u$ and $\mathbf{p}'_v$ (corresponding to the minimum E_u and E_v, respectively) are used to extend the DP path and update the depth gradient of $\mathbf{p}$. We define a state variable $\boldsymbol{S} = [\boldsymbol{s}_u, \boldsymbol{s}_v]$, where $\boldsymbol{s}_u = \mathbf{p}'_u - \mathbf{p}$ and $\boldsymbol{s}_v = \mathbf{p}'_v - \mathbf{p}$ denote the horizontal and vertical components of $\boldsymbol{S}$, respectively[2]. In the k-th iteration, the desired pixels are determined by minimizing $\boldsymbol{E}^{(k)}$ as follows:

$$\boldsymbol{E}^{(k)} = \min_{\boldsymbol{S}} \left\{ \boldsymbol{\mathcal{T}}\left(\boldsymbol{E}^{(k-1)}, \boldsymbol{S}\right)\right\}. \tag{4}$$

The state variable $\boldsymbol{S}^{(k)}$ is transferred to

$$\boldsymbol{S}^{(k)} = \arg\min_{\boldsymbol{S}} \left\{ \boldsymbol{\mathcal{T}}\left(\boldsymbol{E}^{(k-1)}, \boldsymbol{S}\right)\right\}, \tag{5}$$

where $\boldsymbol{\mathcal{T}}$ denotes the smoothness energy transfer function. Therefore, we can update the depth gradient with the desired pixels using:

$$\nabla\hat{z}^{(k)} = \boldsymbol{\mathcal{R}}\left(\nabla\hat{z}^{(k-1)}, \boldsymbol{S}^{(k)}\right), \tag{6}$$

where $\boldsymbol{\mathcal{R}}$ denotes the depth gradient update function.

As shown in Fig. 2, after the initialization of depth gradient $\nabla\hat{z}$ and DP variables $\boldsymbol{E}$ and $\boldsymbol{S}$ (detailed in Sec. 3.1.2), inliers are iteratively introduced using the smoothness energy transfer function $\boldsymbol{\mathcal{T}}$ (detailed in Sec. 3.2) and the depth gradients are iteratively updated using the depth gradient update function $\boldsymbol{\mathcal{R}}$ (detailed in Sec. 3.3). The optimum depth gradients can be obtained when the smoothness energy converges to the minimum. The pseudo-code of multi-directional DP is given in Algorithm 1.

3.1.2 Initialization of Depth Gradient, Smoothness Energy, and State Variable

In the initialization stage, the coarse depth gradient $\nabla\hat{z}^{(0)}$ of a given pixel is obtained by computing the finite difference (FD) between adjacent pixels. The FDs of depth are divided into $\Delta_f z = [\Delta_f z_u, \Delta_f z_v]^\top$ and $\Delta_b z = [\Delta_b z_u, \Delta_b z_v]^\top$, where the subscripts u and v represent the horizontal and vertical directions, respectively, and Δ_f and Δ_b represent forward and backward FDs, respectively. In order to

[2] $\boldsymbol{s}_u, \boldsymbol{s}_v \in \left\{[i,j]^\top \mid i,j \in \{-1,0,1\}\right\}$ because only adjacent pixels are considered.

Authorized licensed use limited to: TONGJI UNIVERSITY. Downloaded on March 04,2023 at 09:03:44 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1 Multi-Directional DP

```
Input: Depth map $z$, Initial depth gradient $[\hat{z}_u^{(0)}, \hat{z}_v^{(0)}]$, Initial smoothness energy components $E_u^{(0)}$ and $E_v^{(0)}$, Initial state variable components $\boldsymbol{s}_u^{(0)}$ and $\boldsymbol{s}_v^{(0)}$.
Output: Optimum depth gradient $[\hat{z}_u, \hat{z}_v]$
$[\hat{z}_u, \hat{z}_v] \leftarrow [\hat{z}_u^{(0)}, \hat{z}_v^{(0)}]$,
$[E_u, E_v] \leftarrow [E_u^{(0)}, E_v^{(0)}]$,
$[\boldsymbol{s}_u, \boldsymbol{s}_v] \leftarrow [\boldsymbol{s}_u^{(0)}, \boldsymbol{s}_v^{(0)}]$
repeat
    Initialize energy set $\Omega_u \leftarrow \varnothing, \Omega_v \leftarrow \varnothing$
    for each $\mathbf{p}'$ do
        $\Omega_u[\mathbf{p}] \leftarrow \Omega_u[\mathbf{p}] \cup \mathcal{T}(E_u, E_v, \mathbf{p}, \mathbf{p}', \boldsymbol{s}_u)$
        $\Omega_v[\mathbf{p}] \leftarrow \Omega_v[\mathbf{p}] \cup \mathcal{T}(E_u, E_v, \mathbf{p}, \mathbf{p}', \boldsymbol{s}_v)$
    end for
    $\boldsymbol{s}_u[\mathbf{p}] \leftarrow \arg\min_{\mathbf{p}' \in \mathcal{N}_1(\mathbf{p})} \{\Omega_u[\mathbf{p}][\mathbf{p}']\} - \mathbf{p}$
    $\boldsymbol{s}_v[\mathbf{p}] \leftarrow \arg\min_{\mathbf{p}' \in \mathcal{N}_1(\mathbf{p})} \{\Omega_v[\mathbf{p}][\mathbf{p}']\} - \mathbf{p}$
    $E_u[\mathbf{p}] \leftarrow \Omega_u[\mathbf{p}][\mathbf{p} + \boldsymbol{s}_u]$
    $E_v[\mathbf{p}] \leftarrow \Omega_v[\mathbf{p}][\mathbf{p} + \boldsymbol{s}_v]$
    $[\hat{z}_u, \hat{z}_v] \leftarrow [\mathcal{R}(\hat{z}_u, \hat{z}_v, \boldsymbol{s}_u), \mathcal{R}(\hat{z}_u, \hat{z}_v, \boldsymbol{s}_v)]$
until all $(\boldsymbol{s}_u[:] = \mathbf{0})$ and all $(\boldsymbol{s}_v[:] = \mathbf{0})$
return $[\hat{z}_u, \hat{z}_v]$
```

achieve the discontinuity awareness ability, we set different weights to these FD operators by comparing the local smoothness measured by $\hat{z}_{uu}$ and $\hat{z}_{vv}$, the second-order partial derivatives of adjacent pixels, as follows:

$$\boldsymbol{\eta} = \big[\eta_u, \eta_v\big] = \arg\min_{[i,j]} \{|\hat{z}_{uu}(u+i, v)| + |\hat{z}_{vv}(u, v+j)|\}, \tag{7}$$

where $i, j \in \{-1, 0, 1\}$. We can then use the smoothest pixels to linearly interpolate $\Delta_f z$ and $\Delta_b z$:

$$\nabla \hat{z}^{(0)} = \frac{1}{2}(\mathbf{1} + \boldsymbol{\eta})^\top \circ \Delta_f z + \frac{1}{2}(\mathbf{1} - \boldsymbol{\eta})^\top \circ \Delta_b z, \tag{8}$$

where $\circ$ denotes the Hadamard product and $\mathbf{1} = [1, 1]^\top$. Furthermore, we initialize the smoothness energy $\boldsymbol{E}^{(0)}$ as $\big[\min\{|\hat{z}_{uu}(u+i, v)|\}, \min\{|\hat{z}_{vv}(u, v+j)|\}\big]$ and the state variable $\boldsymbol{S}^{(0)}$ as $\mathrm{diag}(\eta_u, \eta_v)$.

3.2. Smoothness Energy Transfer Function

This section discusses the formulation of the smoothness energy transfer function $\boldsymbol{\mathcal{T}}$. Energy transfer can be divided into two categories: 1) collinear transfer ($\boldsymbol{s}_u \times \mathbf{e}_1 = \mathbf{0}$ or $\boldsymbol{s}_v \times \mathbf{e}_2 = \mathbf{0}$, where $\mathbf{e}_1 = [1, 0]^\top$ and $\mathbf{e}_2 = [0, 1]^\top$ are the unit orthogonal base of 2D Euclidean space); 2) non-collinear transfer ($\boldsymbol{s}_u \times \mathbf{e}_1 \neq \mathbf{0}$ or $\boldsymbol{s}_v \times \mathbf{e}_2 \neq \mathbf{0}$). As a general rule, the collinear pixels should be introduced to adapt to the convex surface if they are smooth enough; otherwise, non-collinear pixels should be introduced to deal with discontinuities.

3.2.1 Depth Gradient Calculation

Based on the gradient theorem [25], the relationship among z, z_u, and z_v can be written as follows:

$$z(\mathbf{p}') - z(\mathbf{p}) = \int_{\mathcal{L}} z_u \, du + z_v \, dv, \tag{9}$$

where $\mathbf{p} = [u_0, v_0]^\top$ is the given pixel, $\mathbf{p}' = [u_1, v_1]^\top$ is its adjacent pixel to be introduced (satisfying $|u_0 - u_1| \leq 1, |v_0 - v_1| \leq 1$), and $\mathcal{L}$ is the path from $\mathbf{p}$ to $\mathbf{p}'$. We can therefore obtain (i) the collinear transfer as follows:

$$\begin{aligned} z(\mathbf{p}') - z(\mathbf{p}) &= \int_{u_0}^{u_1} z_u \, du, \text{ or} \\ z(\mathbf{p}') - z(\mathbf{p}) &= \int_{v_0}^{v_1} z_v \, dv, \end{aligned} \tag{10}$$

and (ii) the non-collinear transfer as follows:

$$z(\mathbf{p}') - z(\mathbf{p}) = \int_{\mathcal{L}_1} z_u \, du + z_v \, dv = \int_{\mathcal{L}_2} z_u \, du + z_v \, dv, \tag{11}$$

where $\mathcal{L}_1$ and $\mathcal{L}_2$ are two different paths from $\mathbf{p}$ to $\mathbf{p}'$.

In practice, however, the discrete $\hat{z}_u$ and $\hat{z}_v$ can result in errors in (10) and (11). The errors of $\hat{z}_u$ and $\hat{z}_v$ can be estimated using the path integral of z_{uu} and z_{vv} based on the Taylor expansion (more details are provided in the supplement). Hence, in order to measure these errors in a more convenient way, we propose the PD norm, which denotes the path integral (sum) of a series of $|z_{uu}|$ and $|z_{vv}|$.

3.2.2 Path Discontinuity Norm

Let $z(u, v)$ be a function in $\mathbb{R}^2$ defined on an open set Ω and $\mathcal{L}$ be a specific path contained in Ω. The discontinuity extent of the path can be measured using our proposed PD norm as follows:

$$||z||_{\mathrm{PD}} = \int_{\mathcal{L}} |z_{uu} \, du| + |z_{vv} \, dv|. \tag{12}$$

After computing the PD norm, we can judge whether an adjacent pixel $\mathbf{p}'$ should be introduced. Therefore, we define the smoothness energy component as the PD norm with respect to a given path. In each iteration, we introduce two adjacent pixels which respectively minimize the smoothness energy components as follows:

$$\begin{gathered} \boldsymbol{s}_u, \boldsymbol{s}_v = \arg\min_{\mathbf{p}' \in \mathcal{N}_1(\mathbf{p})} \{||z||_{\mathrm{PD}}\} - \mathbf{p}, \\ \mathcal{L} : \begin{cases} \mathbf{p} \to \mathbf{p}' & \text{, if } \boldsymbol{s}_u \times \mathbf{e}_1 = \mathbf{0} \text{ (or } \boldsymbol{s}_v \times \mathbf{e}_2 = \mathbf{0}\text{)} \\ \mathbf{p} \to \mathbf{p}' \to \mathbf{p} & \text{, if } \boldsymbol{s}_u \times \mathbf{e}_1 \neq \mathbf{0} \text{ (or } \boldsymbol{s}_v \times \mathbf{e}_2 \neq \mathbf{0}\text{)}, \end{cases} \end{gathered} \tag{13}$$

Authorized licensed use limited to: TONGJI UNIVERSITY. Downloaded on March 04,2023 at 09:03:44 UTC from IEEE Xplore. Restrictions apply.

where $\mathcal{N}_1(\mathbf{p}) = \{\mathbf{p}' : ||\mathbf{p}' - \mathbf{p}||_\infty \leq 1\}$, and the smoothness energy components are minimized accordingly:

$$
\begin{aligned}
&E_u(\mathbf{p}), E_v(\mathbf{p}) = \min_{\mathbf{p}' \in \mathcal{N}_1(\mathbf{p})} \{||z||_{\text{PD}}\} \\
&\mathcal{L} : \begin{cases} \mathbf{p} \to \mathbf{p}' & \text{, if } \boldsymbol{s}_u \times \mathbf{e}_1 = \mathbf{0} \text{ (or } \boldsymbol{s}_v \times \mathbf{e}_2 = \mathbf{0}) \\ \mathbf{p} \to \mathbf{p}' \to \mathbf{p} & \text{, if } \boldsymbol{s}_u \times \mathbf{e}_1 \neq \mathbf{0} \text{ (or } \boldsymbol{s}_v \times \mathbf{e}_2 \neq \mathbf{0}). \end{cases}
\end{aligned} \tag{14}
$$

The determination of co-planar pixels is, therefore, converted into an energy minimization problem.

3.2.3 Energy Minimization Strategy

Since depth is discrete, we formulate the energy transfer function $\boldsymbol{\mathcal{T}}$ in (4) as follows:

$$
\begin{aligned}
&E_u^{(k)}(\mathbf{p}), E_v^{(k)}(\mathbf{p}) = \min_{\mathbf{p}' \in \mathcal{N}_1(\mathbf{p})} \Big\{ E_p^{(k-1)}(\mathbf{p}), \\
&|\hat{z}_{pp}(\mathbf{p}'_p)| + \mathbb{I}(\mathbf{p}, \mathbf{p}'_p), \\
&2 \cdot \left[|\hat{z}_{oo}(\mathbf{p}'_o)| + E_p^{(k-1)}(\mathbf{p}'_o) \right], \\
&|\hat{z}_{oo}(\mathbf{p}'_o)| + E_p^{(k-1)}(\mathbf{p}'_o) + |\hat{z}_{pp}(\mathbf{p}'_d)| + E_o^{(k-1)}(\mathbf{p}'_d) \Big\},
\end{aligned} \tag{15}
$$

where the subscripts p, o, and d respectively denote the variable which is parallel, orthogonal, and diagonal to the given axis (either u-axis or v-axis); the representations of E_p, E_o, $\hat{z}_{pp}$, $\hat{z}_{oo}$, $\mathbf{p}'_p$, $\mathbf{p}'_o$, and $\mathbf{p}'_d$ are given in the supplement; and the indicator function

$$
\mathbb{I}(\mathbf{p}, \mathbf{p}'_p) = \begin{cases} 0, & \text{if } (\hat{z}_{pp}(\mathbf{p} + \boldsymbol{s}_p) \cdot \hat{z}_{pp}(\mathbf{p}'_p + \boldsymbol{s}_p) > 0) \\ & \wedge (\boldsymbol{s}_p(\mathbf{p}) = \boldsymbol{s}_p^{(k-1)}(\mathbf{p}'_p)) \\ \infty, & \text{otherwise} \end{cases} \tag{16}
$$

add constraints to the monotonicity and convexity of the smoothness energy transfer function, improving the DP adaptivity to convex surfaces. The four terms in (15), in turn, denote the smoothness energy components when we introduce 1) no other pixels, 2) a parallel pixel $\mathbf{p}'_p$, 3) an orthogonal pixel $\mathbf{p}'_o$, and 4) a diagonal pixel $\mathbf{p}'_d$.

3.3. Depth Gradient Update Function

This subsection discusses the formulation of the depth gradient update function $\boldsymbol{\mathcal{R}}$. Similar to $\boldsymbol{\mathcal{T}}$, depth gradient update can be divided into two categories: 1) collinear update ($\boldsymbol{s}_u \times \mathbf{e}_1 = \mathbf{0}$ or $\boldsymbol{s}_v \times \mathbf{e}_2 = \mathbf{0}$) and 2) non-collinear update ($\boldsymbol{s}_u \times \mathbf{e}_1 \neq \mathbf{0}$ or $\boldsymbol{s}_v \times \mathbf{e}_2 \neq \mathbf{0}$). For the collinear update, we use all the collinear pixels which satisfy the constraint in (16) to estimate depth gradients, while on the other hand, for the non-collinear update, we replace the depth gradient of each given pixel with the ones on other smoother paths using (11).

3.3.1 Depth Gradient Collinear Update with Recursive Polynomial Interpolation

The optimum $\nabla\hat{z}$ can be obtained by interpolating the depth of n collinear pixels into an $(n-1)$-th order polynomial and subsequently computing its derivative. Lagrange or Newton polynomial interpolation has redundant computations, as each pixel is repeatedly used for polynomial interpolation. Furthermore, it is incredibly complex to yield the closed-form solution of the polynomial's derivative. To simplify the depth gradient collinear update process, we introduce a novel RPI algorithm, which can update $\nabla\hat{z}$ recursively with only adjacent pixels until the farthest pixel is accessed.

As $\hat{z}_u$ and $\hat{z}_v$ can be computed in the same way, we only provide the details on $\hat{z}_u$ computation in this paper. To compute $\hat{z}_u$ of a given pixel $\mathbf{p} = [u_0, v_0]^\top$ in the k-th iteration, we use a $(k+1)$-th order polynomial to interpolate the depth of $(k+2)$ pixels, which have the same vertical coordinates v_0. The $(k+1)$-th order polynomial $f_0^{(k+1)}(u)$ interpolated by $\{(u_0, z(u_0)), (u_0+1, z(u_0+1)), \dots, (u_0+k+1, z(u_0+k+1))\}$ can be expanded into a recursive form as follows:

$$
f_0^{(k+1)}(u) = \frac{u_0 + k + 1 - u}{k+1} f_0^{(k)} + \frac{u - u_0}{k+1} f_1^{(k)}, \tag{17}
$$

where the k-th order polynomials $f_0^{(k)}$ and $f_1^{(k)}$ are respectively interpolated by the first and the last $k+1$ pixels (the proof of the theorem is provided in the supplement).

Since $\hat{z}_u^{(k-1)}(u_0) = \frac{df_0^{(k)}}{du}(u_0)$ and $\hat{z}_u^{(k-1)}(u_0+1) = \frac{df_1^{(k)}}{du}(u_0+1)$ have been computed in the last iteration, we can update $\hat{z}_u(u_0)$ using:

$$
\begin{aligned}
\hat{z}_u^{(k)}(u_0) &= \frac{df_0^{(k+1)}}{du}(u_0) \\
&= \hat{z}_u^{(k-1)}(u_0) - \frac{1}{k+1} \cdot \left[z(u_0) - f_1^{(k)}(u_0) \right].
\end{aligned} \tag{18}
$$

Substituting $f_1^{(k)}(u_0) = z(u_0+1) - \frac{df_1^{(k)}}{du}(u_0+1) = z(u_0+1) - \hat{z}_u^{(k-1)}(u_0+1)$ into (18) yields

$$
\begin{aligned}
&\hat{z}_u^{(k)}(u_0) = \hat{z}_u^{(k-1)}(u_0) - \\
&\frac{1}{k+1} \cdot \left[z(u_0) - z(u_0+1) + \hat{z}_u^{(k-1)}(u_0+1) \right].
\end{aligned} \tag{19}
$$

3.3.2 Depth Gradient Non-Collinear Update

Replacing the integral in (11) with summation yields:

$$
\hat{z}_p(\mathbf{p}) \pm \hat{z}_o(\mathbf{p}'_p) = \hat{z}_p(\mathbf{p}'_o) \pm \hat{z}_o(\mathbf{p}). \tag{20}
$$

As the pixels $\mathbf{p}'_p$ are regarded as outliers in non-colinear update, we replace $\hat{z}_o(\mathbf{p}'_p)$ with $\hat{z}_o(\mathbf{p}'_d)$ or $\hat{z}_o(\mathbf{p})$ based on

Authorized licensed use limited to: TONGJI UNIVERSITY. Downloaded on March 04,2023 at 09:03:44 UTC from IEEE Xplore. Restrictions apply.

the state variable. The depth gradient update function $\boldsymbol{\mathcal{R}}$ in (6) can be rewritten as follows:

$$\hat{z}_u^{(k)}(\mathbf{p}), \hat{z}_v^{(k)}(\mathbf{p}) =$$
$$\begin{cases} \hat{z}_p^{(k-1)}(\mathbf{p}), & \text{if } \boldsymbol{s}_u^{(k)}, \boldsymbol{s}_v^{(k)} = \mathbf{0} \\ \hat{z}_p^{(k-1)}(\mathbf{p}) \pm \frac{1}{k+1}\left[z(\mathbf{p}'_p) - z(\mathbf{p})\right] & \\ -\frac{1}{k+1}\hat{z}_p^{(k-1)}(\mathbf{p}'_p), & \text{if } \boldsymbol{s}_u^{(k)}, \boldsymbol{s}_v^{(k)} = \mathbf{p}'_p - \mathbf{p} \\ \hat{z}_p^{(k-1)}(\mathbf{p}'_o), & \text{if } \boldsymbol{s}_u^{(k)}, \boldsymbol{s}_v^{(k)} = \mathbf{p}'_o - \mathbf{p} \\ \hat{z}_p^{(k-1)}(\mathbf{p}'_o) \pm & \\ \left[\hat{z}_o^{(k-1)}(\mathbf{p}) - \hat{z}_o^{(k-1)}(\mathbf{p}'_d)\right], & \text{if } \boldsymbol{s}_u^{(k)}, \boldsymbol{s}_v^{(k)} = \mathbf{p}'_d - \mathbf{p}. \end{cases} \tag{21}$$

The representations of $\hat{z}_p$ and $\hat{z}_o$ are given in the supplement. The four terms in (21), in turn, represent 1) the unchanged depth gradient, 2) parallel update via RPI algorithm, 3) orthogonal update, and 4) diagonal update. The orthogonal and diagonal updates are based on the gradient theorem [25].

4. Experiments

4.1. Datasets and Evaluation Metrics

In this paper, we followed [7] and conducted comprehensive experiments to qualitatively and quantitatively evaluate the performance of our proposed SDA-SNE. More details on the 3F2N datasets[3] are available in [7]. Moreover, since range sensor data are typically noisy, we add random Gaussian noise of different variances σ to the original 3F2N datasets to compare the robustness between our proposed SDA-SNE and other SoTA SNEs.

Two evaluation metrics are used to quantify the accuracy of SNEs: the average angular error (AAE) [15]:

$$e_{\mathrm{A}}(\mathcal{M}) = \frac{1}{P}\sum_{k=1}^{P}\phi_k, \quad \phi_k = \cos^{-1}\left(\frac{\langle \mathbf{n}_k, \hat{\mathbf{n}}_k \rangle}{||\mathbf{n}_k||_2 ||\hat{\mathbf{n}}_k||_2}\right), \tag{22}$$

and the proportion of good pixels (PGP) [15]:

$$e_{\mathrm{P}}(\mathcal{M}) = \frac{1}{P}\sum_{k=1}^{P}\delta\left(\phi_k, \varphi_k\right), \quad \delta = \begin{cases} 0 & (\phi_k > \varphi) \\ 1 & (\phi_k \leq \varphi) \end{cases}, \tag{23}$$

where P denotes the total number of pixels used for evaluation, φ denotes the angular error tolerance, $\mathcal{M}$ represents the given SNE, and $\mathbf{n}_k$ and $\hat{\mathbf{n}}_k$ represent the ground-truth and estimated surface normals, respectively.

In addition to the above-mentioned evaluation metrics, we also introduce a novel evaluation metric, referred to as cross accuracy ratio (CAR), to depict the performance comparison between two SNEs as follows:

$$r_{\mathrm{A}}(\mathcal{M}_1, \mathcal{M}_2) = \frac{e_{\mathrm{A}}(\mathcal{M}_1)}{e_{\mathrm{A}}(\mathcal{M}_2)}. \tag{24}$$

$\mathcal{M}_2$ outperforms $\mathcal{M}_1$ in terms of e_{A} if $r_{\mathrm{A}} > 1$, and vice versa if $0 < r_{\mathrm{A}} < 1$.

4.2. Implementation Details

As discussed in Sec. 3, initial depth gradients can be estimated by convolving a depth image with image gradient filters, *e.g.*, Sobel [21], Scharr [11], Prewitt [18], *etc.* The second-order filters, *e.g.*, Laplace [20], can be used to estimate the local depth gradient smoothness $\hat{z}_{uu}$ and $\hat{z}_{vv}$. We utilize a finite forward difference (FFD) kernel, *i.e.*, $[0, -1, 1]$, as well as a finite backward difference (FBD) kernel, *i.e.*, $[-1, 1, 0]$ to initialize (coarse) $\hat{z}_u^{(0)}$ and $\hat{z}_v^{(0)}$, and use a finite Laplace (FL) kernel, *i.e.*, $[-1, 2, -1]$ to estimate $\hat{z}_{uu}$ and $\hat{z}_{vv}$. We also conduct experiments w.r.t. different number of iterations to quantify the performance of our proposed SDA-SNE. By implementing the optimum maximum iteration, the trade-off between the speed and accuracy of our algorithm can be significantly minimized.

4.3. Algorithm Convergence and Computational Complexity

Our proposed SDA-SNE converges when the state variable and surface normal estimation remains stable. We can obtain the lower bound of the smoothness energy components as follows:

$$E_u(\mathbf{p}), E_v(\mathbf{p}) \geq \min_{\mathbf{p}' \in \mathcal{N}_1(\mathbf{p})}\left\{|\hat{z}_{uu}(\mathbf{p}')|, |\hat{z}_{vv}(\mathbf{p}')|\right\} \geq 0. \tag{25}$$

The smoothness energy decreases monotonously after each iteration. Therefore, all state variables stop transferring after finite iterations (the smoothness energy converges to a global minimum).

The computational complexity of our proposed PRI algorithm is $\mathcal{O}(n)$ (n denotes the number of interpolated pixels), when computing the depth gradient using (21). Therefore, RPI is much more efficient than Lagrange or Newton polynomial interpolation whose computational complexity is $\mathcal{O}(n^2)$.

Furthermore, when the image resolution is $M \times N$ pixels, the computational complexity of our proposed SDA-SNE is $\mathcal{O}(lMN)$, where l denotes the number of iterations in DP. In most cases, $l \ll M, N$ because DP automatically stops when $|\hat{z}_{uu}|$ and $|\hat{z}_{vv}|$ no longer decrease. If we set a limit on l, SDA-SNE's computational complexity can be reduced to $\mathcal{O}(MN)$, which is identical to the computational complexities of 3F2N [7] and CP2TV [16].

4.4. Performance Evaluation

As our proposed multi-directional DP algorithm mainly aims at improving the performance of depth gradient estimation, it is compatible with any depth-to-normal SNEs.

[3] sites.google.com/view/3f2n/datasets

Authorized licensed use limited to: TONGJI UNIVERSITY. Downloaded on March 04,2023 at 09:03:44 UTC from IEEE Xplore. Restrictions apply.

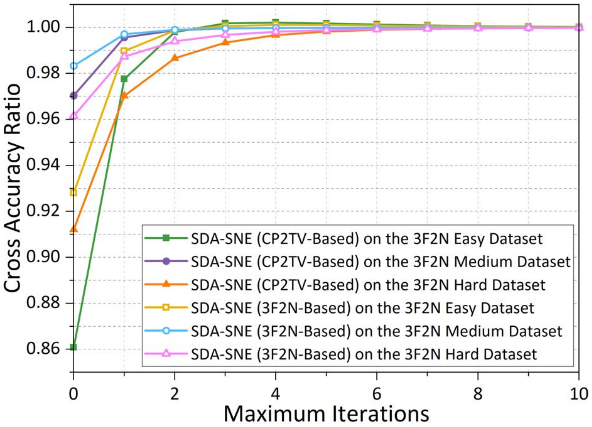

Figure 3. CAR comparison between CP2TV-based and 3F2N-based SDA-SNEs w.r.t. different maximum iterations.

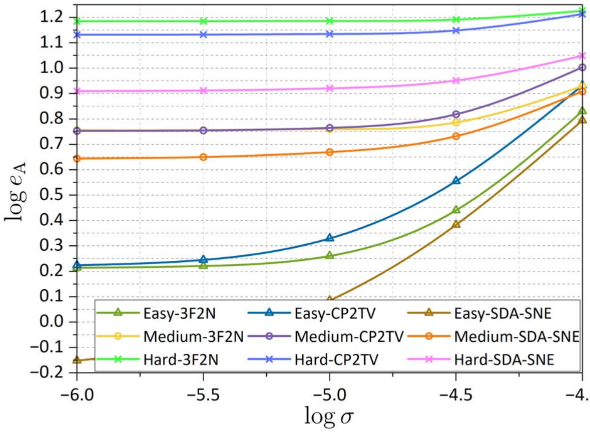

Figure 4. AAE comparison among 3F2N, CP2TV, and SDA-SNE on the 3F2N datasets with different levels of Gaussian noise added.

The AAE scores of SoTA depth-to-normal SNEs and such SNEs using our depth gradient estimation strategy are given in Table 1. It can be observed that by using our proposed depth gradient estimation strategy, the AAE scores of such depth-to-normal SNEs decrease by about 20-60%. Since SDA-SNE based on CP2TV performs better than SDA-SNE based on 3F2N, we use CP2TV to estimate n_z in the following experiments. The qualitative comparison among these SNEs is shown in Fig. 5. It can be observed that our proposed SDA-SNE significantly outperforms 3F2N and CP2TV near/on discontinuities.

Table 1. Comparison of e_{A} (degrees) between 3F2N and CP2TV w/ and w/o our depth gradient estimation strategy leveraged.

Datasets	3F2N		CP2TV	
	w/o SDA	w/ SDA	w/o SDA	w/ SDA
Easy	1.657	**0.782**	1.686	**0.677**
Medium	5.686	**4.535**	6.015	**4.379**
Hard	15.315	**9.237**	13.819	**8.098**

Additionally, we compare the performance of our proposed SDA-SNE w.r.t. a collection of maximum iterations. Fig. 3 shows the CAR scores achieved by CP2TV-based and 3F2N-based SDA-SNEs on the 3F2N easy, medium, and hard datasets. It can be observed that the performance of SDA-SNE saturates after only several iterations. The accuracy increases by less than 5% when the maximum iteration is set to infinity. Therefore, the maximum iteration of multi-directional DP can be set to 3 to minimize its trade-off between speed and accuracy. Moreover, it can be observed that our proposed SDA-SNE converges (the CAR score reaches the maximum) with more iterations on the hard dataset than on the easy and medium datasets. This is probably due to the fact that the hard dataset possesses more discontinuities than the easy and medium datasets. As a result, more distant pixels are required to be introduced to yield better depth gradients.

Moreover, we compare our proposed SDA-SNE with all other SoTA SNEs presented in Sec. 2. e_{A} and e_{P} of all SNEs on the 3F2N easy, medium, and hard datasets are given in Table 2. The e_{A} scores achieved by SDA-SNE are less than 1° (easy), 5° (medium), and 9° (hard), respectively. The e_{P} scores (tolerance: 10°) achieved by SDA-SNE are about 100% (easy), 90% (medium), and 80% (hard), respectively. These results suggest that our proposed SDA-SNE performs significantly better than all other SoTA SNEs, no matter whether an iteration limit is added or not.

Furthermore, we compare the robustness (to random Gaussian noise) of our proposed SDA-SNE with two SoTA depth-to-normal SNEs (3F2N [7] and CP2TV [16]), as shown in Fig. 4. The logarithms of AAE scores and Gaussian variances are used to include the results of different datasets into a single figure. It is evident that (1) the e_{A} scores achieved by all SNEs increase monotonically with the increasing noise level, and (2) our proposed SDA-SNE outperforms 3F2N and CP2TV at different noise levels. In addition, the compared SNEs' performance on the easy dataset degrades more dramatically than the medium and hard datasets, as the original easy dataset contains much fewer discontinuities. Although the performance of these depth-to-normal SNEs becomes remarkably similar, with the increase in noise level, our proposed SDA-SNE consistently outperforms 3F2N and CP2TV. This further demonstrates the superior robustness of our algorithm over others.

5. Conclusion

This paper presented a highly accurate discontinuity-aware surface normal estimator, referred to as SDA-SNE. Our approach computes surface normals from a depth image by iteratively introducing adjacent co-planar pixels using a novel multi-directional dynamic programming algorithm. To refine the depth gradient in each iteration, we introduced a novel recursive polynomial interpolation al-

Authorized licensed use limited to: TONGJI UNIVERSITY. Downloaded on March 04,2023 at 09:03:44 UTC from IEEE Xplore. Restrictions apply.

Table 2. Comparison of e_{A} and e_{P} (with respect to different φ) among SoTA SNEs on the 3F2N datasets [7].

Method	e_{A} (degrees) $\downarrow$			e_{p} $\uparrow$								
	Easy	Medium	Hard	Easy			Medium			Hard		
				φ=10°	φ=20°	φ=30°	φ=10°	φ=20°	φ=30°	φ=10°	φ=20°	φ=30°
PlaneSVD [14]	2.07	6.07	17.59	0.9648	0.9792	0.9855	0.8621	0.9531	0.9718	0.6202	0.7394	0.7914
PlanePCA [12]	2.07	6.07	17.59	0.9648	0.9792	0.9855	0.8621	0.9531	0.9718	0.6202	0.7394	0.7914
VectorSVD [13]	2.13	6.27	18.01	0.9643	0.9777	0.9846	0.8601	0.9495	0.9683	0.6187	0.7346	0.7848
AreaWeighted [13]	2.20	6.27	17.03	0.9636	0.9753	0.9819	0.8634	0.9504	0.9665	0.6248	0.7448	0.7977
AngleWeighted [13]	1.79	5.67	13.26	0.9762	0.9862	0.9893	0.8814	0.9711	0.9809	0.6625	0.8075	0.8651
FALS [1]	2.26	6.14	17.34	0.9654	0.9794	0.9857	0.8621	0.9547	0.9731	0.6209	0.7433	0.7961
SRI [1]	2.64	6.71	19.61	0.9499	0.9713	0.9798	0.8431	0.9403	0.9633	0.5594	0.6932	0.7605
LINE-MOD [8]	2.64	6.71	19.61	0.8542	0.9085	0.9343	0.7277	0.8803	0.9282	0.3375	0.4757	0.5636
SNE-RoadSeg [6]	2.04	6.28	16.37	0.9693	0.9810	0.9871	0.8618	0.9512	0.9725	0.6226	0.7589	0.8113
3F2N [7]	1.66	5.69	15.31	0.9723	0.9829	0.9889	0.8722	0.9600	0.9766	0.6631	0.7821	0.8289
CP2TV [16]	1.69	6.02	13.82	0.9740	0.9843	0.9899	0.8512	0.9554	0.9755	0.6840	0.8099	0.8562
SDA-SNE (iteration = ∞)	0.68	**4.38**	**8.10**	**0.9947**	0.9982	0.9991	**0.9075**	**0.9868**	**0.9939**	**0.8035**	**0.9254**	**0.9461**
SDA-SNE (iteration = 3)	**0.67**	**4.38**	8.14	**0.9947**	**0.9983**	**0.9992**	**0.9075**	0.9867	0.9938	0.8027	0.9174	0.9453

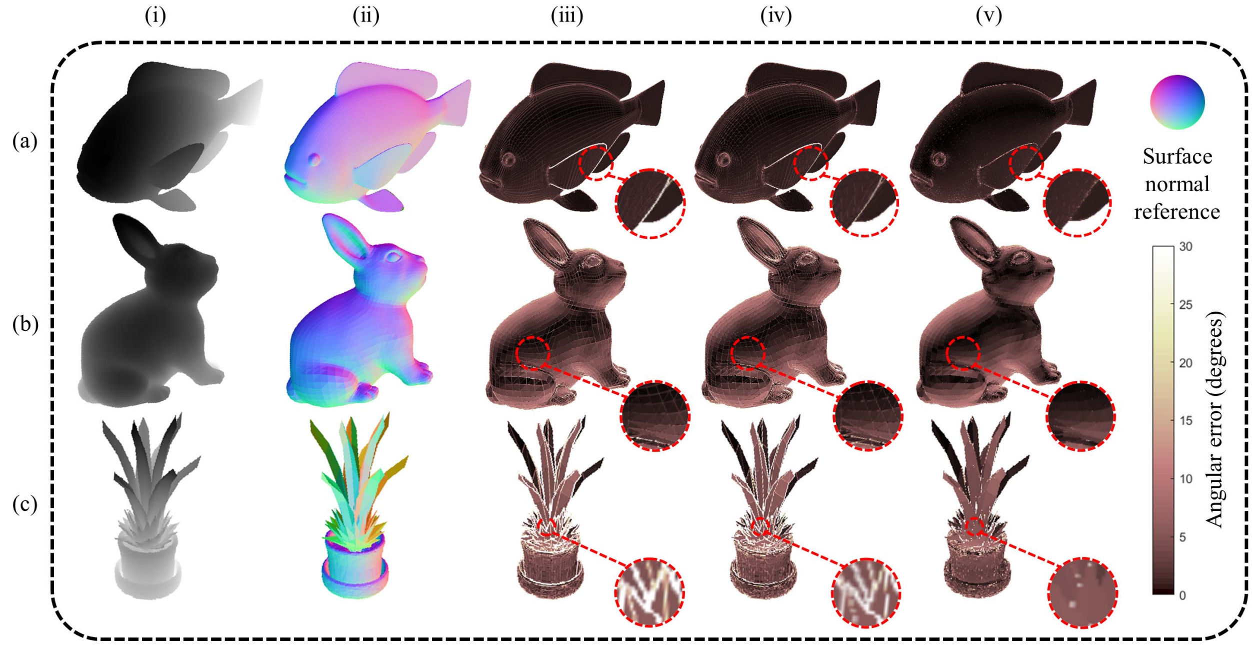

Figure 5. Examples of experimental results: Columns (i)-(v) show the depth images, surface normal ground truth, and the experimental results obtained using 3F2N, CP2TV, and our proposed SDA-SNE, respectively. Rows (a)-(c) show the results of the easy, medium, and hard datasets, respectively.

gorithm with high computational efficiency. Our proposed depth gradient estimation approach is compatible with any depth-to-normal surface normal estimator, such as 3F2N and CP2TV. To evaluate the accuracy of our proposed surface normal estimator, we conducted extensive experiments on both clean and noisy datasets. Our proposed SDA-SNE achieves the highest accuracy on clean 3F2N datasets (0.68°, 4.38°, 8.10° on the easy, medium, and hard datasets, respectively), outperforming all other SoTA surface normal estimators. It also demonstrates high robustness to different levels of random Gaussian noise. Additional experimental results suggest that our proposed SDA-SNE can achieve a similar performance when reducing the maximum iteration of multi-directional dynamic programming to 3. This ensures the high efficiency of our proposed SDA-SNE in various computer vision and robotics applications requiring real-time performance.

6. Acknowledgements

This work was supported by the National Key R&D Program of China, under grant No. 2020AAA0108100, awarded to Prof. Qijun Chen. This work was also supported by the Fundamental Research Funds for the Central Universities, under projects No. 22120220184, No. 22120220214, and No. 2022-5-YB-08, awarded to Prof. Rui Fan.

Authorized licensed use limited to: TONGJI UNIVERSITY. Downloaded on March 04,2023 at 09:03:44 UTC from IEEE Xplore. Restrictions apply.

References

[1] Hernan Badino, Daniel Huber, Yongwoon Park, and Takeo Kanade. Fast and accurate computation of surface normals from range images. In *2011 IEEE Int. Conf. on Robotics and Automation*, pages 3084–3091. IEEE, 2011. 8

[2] Dániel Baráth, Ivan Eichhardt, and Levente Hajder. Optimal multi-view surface normal estimation using affine correspondences. *IEEE Transactions on Image Processing*, 28(7):3301–3311, 2019. 1, 2

[3] Richard Bormann, Joshua Hampp, Martin Hägele, and Markus Vincze. Fast and accurate normal estimation by efficient 3d edge detection. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3930–3937, 2015. 1

[4] Barath Daniel, Jozsef Molnar, and Levente Hajder. Optimal surface normal from affine transformation. In *International Conference on Computer Vision Theory and Applications*, volume 2, pages 305–316. SciTePress, 2015. 1, 2

[5] Rui Fan, Umar Ozgunalp, Brett Hosking, Ming Liu, and Ioannis Pitas. Pothole detection based on disparity transformation and road surface modeling. *IEEE Transactions on Image Processing*, 29:897–908, 2019. 1

[6] Rui Fan, Hengli Wang, Peide Cai, and Ming Liu. SNE-RoadSeg: Incorporating surface normal information into semantic segmentation for accurate freespace detection. In *European Conference on Computer Vision*, pages 340–356. Springer, 2020. 2, 8

[7] Rui Fan, Hengli Wang, Bohuan Xue, Huaiyang Huang, Yuan Wang, Ming Liu, and Ioannis Pitas. Three-Filters-to-Normal: An accurate and ultrafast surface normal estimator. *IEEE Robotics and Automation Letters*, 6(3):5405–5412, 2021. 1, 2, 6, 7, 8, 11

[8] Stefan Hinterstoisser, Cedric Cagniart, Slobodan Ilic, Peter Sturm, Nassir Navab, Pascal Fua, and Vincent Lepetit. Gradient response maps for real-time detection of textureless objects. *IEEE Trans on Pattern Analysis and Machine Intelligence*, 34(5):876–888, 2011. 8

[9] Stefan Hinterstoisser, Cedric Cagniart, Slobodan Ilic, Peter Sturm, Nassir Navab, Pascal Fua, and Vincent Lepetit. Gradient response maps for real-time detection of textureless objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(5):876–888, 2012. 1

[10] Arvind V. Iyer and Johannes Burge. Depth variation and stereo processing tasks in natural scenes. *Journal of vision*, 18(6):4–4, 2018. 1

[11] Bernd Jähne, Hanno Scharr, Stefan Körkel, et al. Principles of filter design. *Handbook of computer vision and applications*, 2:125–151, 1999. 6

[12] Krzysztof Jordan and Philippos Mordohai. A quantitative evaluation of surface normal estimation in point clouds. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4220–4226. IEEE, 2014. 1, 2, 8

[13] Klaas Klasing, Daniel Althoff, Dirk Wollherr, and Martin Buss. Comparison of surface normal estimation methods for range sensing applications. In *2009 IEEE International Conference on Robotics and Automation*, pages 3206–3211. IEEE, 2009. 1, 2, 8

[14] Klaas Klasing, Dirk Wollherr, and Martin Buss. Realtime segmentation of range data using continuous nearest neighbors. In *2009 IEEE International Conference on Robotics and Automation*, pages 2431–2436. IEEE, 2009. 1, 2, 8

[15] Feng Lu, Xiaowu Chen, Imari Sato, and Yoichi Sato. Symps: Brdf symmetry guided photometric stereo for shape and light source estimation. *IEEE trans on PAMI*, 40(1):221–234, 2017. 6

[16] Yosuke Nakagawa, Hideaki Uchiyama, Hajime Nagahara, and Rin-Ichiro Taniguchi. Estimating surface normals with depth image gradients for fast and accurate registration. In *2015 International Conference on 3D Vision*, pages 640–647, 2015. 1, 2, 6, 7, 8, 11

[17] D. Ouyang and Feng H. Y. On the normal vector estimation for point cloud data from smooth surfaces. *Computer-Aided Design*, 37(10):1071–1079, 2005. 2

[18] Judith MS Prewitt et al. Object enhancement and extraction. *Picture processing and Psychopictorics*, 10(1):15–19, 1970. 6

[19] Carolina Raposo and Joao P Barreto. Accurate reconstruction of oriented 3d points using affine correspondences. In *European Conference on Computer Vision*, pages 545–560. Springer, 2020. 1, 2

[20] Martin Reuter, Silvia Biasotti, Daniela Giorgi, Giuseppe Patane, and Michela Spagnuolo. Discrete laplace-beltrami operators for shape analysis and segmentation, 2009. 6

[21] Irwin Sobel. An isotropic 3x3 image gradient operator. *Presentation at Stanford A.I. Project 1968*, 02 2014. 6

[22] W. Sun, C. Bradley, Y. F. Zhang, and H. T. Loh. Cloud data modelling employing a unified, non-redundant triangular mesh. *Computer-Aided Design*, 33(2):183–193, 2001. 2

[23] Hengli Wang, Rui Fan, Peide Cai, and Ming Liu. SNE-RoadSeg+: Rethinking depth-normal translation and deep supervision for freespace detection. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1140–1145. IEEE, 2021. 2

[24] Qiang Wang, Shizhen Zheng, Qingsong Yan, Fei Deng, Kaiyong Zhao, and Xiaowen Chu. Irs: A large naturalistic indoor robotics stereo dataset to train deep models for disparity and surface normal estimation. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2021. 11

[25] Richard E Williamson and Hale F Trotter. *Multivariable mathematics*. Pearson, 2004. 4, 6

[26] Qiang-Jun Xie, Wen-Biao Jin, Li Ma, and Di-Bo Hou. Fast global segmentation based on the dual formulation of tv-norm. In *2010 3rd International Congress on Image and Signal Processing*, volume 3, pages 1382–1385, 2010. 10

Authorized licensed use limited to: TONGJI UNIVERSITY. Downloaded on March 04,2023 at 09:03:44 UTC from IEEE Xplore. Restrictions apply.